Dependability in Software Products

Francis Tam
Objectives

• You should be able to:
  1. Classify different types of dependable systems.
  2. Apply some basic high availability implementation techniques.
  3. Explain software quality issues associated with highly available software products.

• A trigger for potential thesis topics.
Contents

1. Introducing high availability
2. Implementing high availability
3. Example in telecommunications
4. Discussions – the software product management connection
5. Concluding remarks
INTRODUCING HIGH AVAILABILITY
Service Disruptions – A Dossier

Sonera phone network disruption over

Network problems blocked 2G and 3G Sonera network connections for more than four hours Thursday afternoon. Mobile phone service provider TeliaSonera announced on Twitter at 5 pm that the problem had been resolved.

Elisa reports problems in web traffic

Customers of one of Finland’s largest internet service providers reported widespread disruptions Wednesday afternoon and evening. The company says the glitch was due to a broken underground cable in Lohja – the company said in a statement that service had been restored early Thursday morning.

http://yle.fi/uutiset/osasto/news/phone_and_it_services_disrupted_in_central_finland/6978147
Big Picture of Concepts

Dependability

attributes

- reliability
- availability
- safety
- integrity
- maintainability

threats

- fault
- error
- failure

means

- fault prevention
- fault tolerance
- fault removal
- fault forecasting

From Toeroe and Tam, Service Availability: Principles and Practice, Wiley 2012.
The Meaning of Availability

- A common measure of availability is the expected fraction of time that a system is available to perform its functions correctly.
- Can be expressed as $t_{op} / (t_{op} + t_{repair})$
  - Operational time = $t_{op}$
  - Downtime = $t_{repair}$
- To be 99.999% (5-9s) available, maximum allowable downtime is just over 5 minutes and 15 seconds in a year.

<table>
<thead>
<tr>
<th>Years of continuous operation</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{op} + t_{repair}$ in minutes</td>
<td>525600</td>
<td>1051200</td>
<td>1576800</td>
<td>2102400</td>
<td>2628000</td>
</tr>
<tr>
<td>Availability (%)</td>
<td>99.0000</td>
<td>99.9000</td>
<td>99.9900</td>
<td>99.9990</td>
<td>99.9999</td>
</tr>
<tr>
<td>Maximum allowable downtime in minutes ($t_{repair}$)</td>
<td>5256.0000</td>
<td>10512.0000</td>
<td>15768.0000</td>
<td>21024.0000</td>
<td>26280.0000</td>
</tr>
</tbody>
</table>
IMPLEMENTING HIGH AVAILABILITY
Achieving High Availability

• Protective redundancy is key

• Dealing with failures by following the phases of fault tolerance:
  1. Error detection
  2. Damage confinement and assessment
  3. Error recovery
  4. Fault treatment and service continuation

• Upgrade matters
Redundancy Models

(a) Active-standby: normal operation

(b) Active-standby: node X failed

(c) N+1 (3+1): normal operation

(d) N+1 (3+1): node Y failed

From Tam, Service Availability Standards for Carrier-Grade Platforms: Creation and Deployment in Mobile Networks, Tampere University of Technology, Publication 800, 2009.
Rolling Upgrade

(a) Initial state

S

\[
\begin{array}{c}
\text{node X} \\
A \\
\end{array}
\]

S

\[
\begin{array}{c}
\text{node Y} \\
A \\
\end{array}
\]

S

\[
\begin{array}{c}
\text{node Z} \\
A \\
\end{array}
\]

S – service

A – application A version 1

(c) Second upgrade

S

\[
\begin{array}{c}
\text{node X} \\
A \\
\end{array}
\]

S

\[
\begin{array}{c}
\text{node Y} \\
A \\
\end{array}
\]

S

\[
\begin{array}{c}
\text{node Z} \\
A \\
\end{array}
\]

S – service

A – application A version 2

A – application A version 1

(b) First upgrade

S

\[
\begin{array}{c}
\text{node X} \\
A \\
\end{array}
\]

S

\[
\begin{array}{c}
\text{node Y} \\
A \\
\end{array}
\]

S

\[
\begin{array}{c}
\text{node Z} \\
A \\
\end{array}
\]

S – service

A – application A version 2

A – application A version 1

(d) Final upgrade

S

\[
\begin{array}{c}
\text{node X} \\
A \\
\end{array}
\]

S

\[
\begin{array}{c}
\text{node Y} \\
A \\
\end{array}
\]

S

\[
\begin{array}{c}
\text{node Z} \\
A \\
\end{array}
\]

S – service

A – application A version 2

A – application A version 1

A – application A version 1

From Tam, Service Availability Standards for Carrier-Grade Platforms: Creation and Deployment in Mobile Networks, Tampere University of Technology, Publication 800, 2009.
Split-Mode Upgrade

(a) Initial state

S1

S1

S1

S1

node W
node X
node Y
node Z

S1 – service version 1

(b) Stop half of the applications and have them upgraded

S1

S1

node W
node X
node Y
node Z

S1 – service version 1

A – application A version 2

A – application A version 1

(c) Switchover to version 2 service

S2

S2

S2

S2

node W
node X
node Y
node Z

S2 – service version 2

A – application A version 2

A – application A version 1

(d) Complete second half upgrade and restore full service

S2

S2

S2

S2

node W
node X
node Y
node Z

S2 – service version 2

A – application A version 2

From Tam, Service Availability Standards for Carrier-Grade Platforms: Creation and Deployment in Mobile Networks, Tampere University of Technology, Publication 800, 2009.
Horses for Courses

- Many variations of redundancy models and upgrade methods
- Trade-off between capabilities and required resources
- Application specific
- Product specific
EXAMPLE IN TELECOMMUNICATIONS
A Simplified Mobile Network

From Tam, Service Availability Standards for Carrier-Grade Platforms: Creation and Deployment in Mobile Networks, Tampere University of Technology, Publication 800, 2009.
Usage Context of a Carrier-Grade Platform

From Tam, Service Availability Standards for Carrier-Grade Platforms: Creation and Deployment in Mobile Networks, Tampere University of Technology, Publication 800, 2009.
What is a Carrier-Grade Platform?

• **Carrier-grade** refers to a class of systems used in public telecommunications network that deliver up to five nines or six nines (99.999% or 99.9999%) availability

• Origin comes from the fact that telecommunications company that provides the public for hire with communications transmission services is known as a carrier

• The equipment associated with providing these highly available services has traditionally been dubbed “carrier-grade”

From Tam, Service Availability Standards for Carrier-Grade Platforms: Creation and Deployment in Mobile Networks, Tampere University of Technology, Publication 800, 2009.
From Tam, Service Availability Standards for Carrier-Grade Platforms: Creation and Deployment in Mobile Networks, Tampere University of Technology, Publication 800, 2009.
Software product management issues associated with a service availability middleware
Software Product Management Connection

• Bring non-functional requirements to the fore
  – Configurable redundancy models for applications
  – The middleware itself has to be highly available

• Testing
  – Fault injection is your friend
    • From hardware to software implemented fault injection
    • Assumed fault model
    • Failure scenarios
    • Compile-time injection
    • Run time injection
    • Tools needed
  – Robustness testing
    • Carnegie Mellon University’s Ballista
    • Fuzzing or fuzz testing
Dependability Benchmarking Vision

- Problems with selecting and integrating off-the-shelf products into the carrier-grade platform
  - Over emphasis on performance – “seems to work” syndrome
  - Application dependent dependability and performance requirements

- Based on previous work – Ballista and DBench
- Preliminary results on 3 Availability Management Framework implementations:
  - OpenAIS-0.80.1, OpenAIS-trunk, SAFE4TRY
  - Fault-load – workload pair

From Tam, Service Availability Standards for Carrier-Grade Platforms: Creation and Deployment in Mobile Networks, Tampere University of Technology, Publication 800, 2009.
CONCLUDING REMARKS
Final Thoughts

• Highly available software products and software quality.
• Cloud changes some aspect of the high availability challenges.
• Tolerating software design fault is the holy grail.
• Designing dependable computer systems should be in the core of the CS/IT/IS curricula.
ETSI Network Function Virtualisation

From ETSI Network Functions Virtualisation (NFV);
Architectural Framework ETSI GS NFV 002 V1.2.1 (2014-12)
Clouding the Availability Issue

Amazon data centre fault knocks websites offline temporarily

Further Reading


• M. Toeroe and F. Tam, Service Availability: Principles and Practice, Wiley 2012