Calculating Total System Availability

KLM ICT Infrastructure

Hoda Rohani, Azad Kamali Roosta
Project Supervisors: Betty Gommans, Leon Gommans

What we will see

- Availability Definition
- How to calculate availability for:
  - A single component
  - Parallel / Serial configurations
- How to calculate availability of a system

Research Project
Place in the Hierarchy

- Artificial IT Intervention Handler (AITIH)
  - To establish a framework for calculation of the availability (as a non-functional requirement) for a KLM Business Application

Availability is a requirement

Definitions

- Availability
  - Reliability Engineering

    A function of time, defined as the probability that system is operating correctly and is available to perform its function at the instant of time t

- Unavailability
  - 1 - Availability

Definitions

- MTBF

  The (mean) time expected between two consecutive system failures

  - High MTBF means...

- MTTR

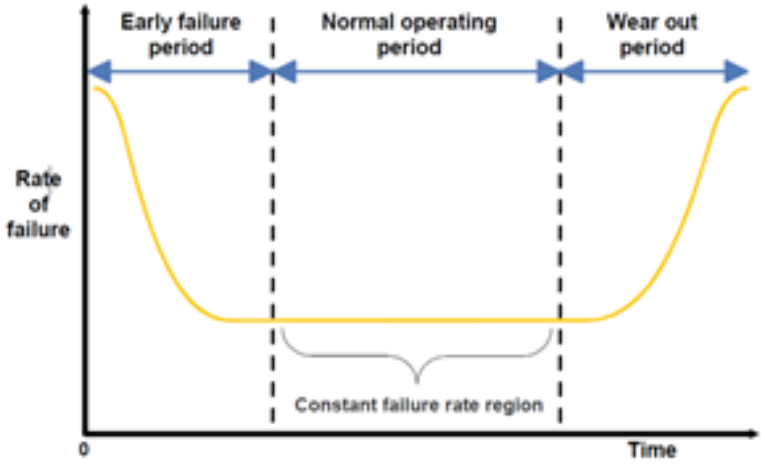
  The (mean) Time required to repair a failed system

  - This time includes …

- Represented in units of hours
- Basic measures of calculating the availability

Failure Rate

- Hardware failures
  - Design Faults,
  - Mechanical malfunction
  - Electronic Interference
- Bathtub Curve



http://www.mana-ups.com

- Software failures:
  - Complexity of software, Size of code. Team experience
  - Depth of testing before releasing the product, Percentage of code reused from a previous stable project

Basic assumption: Constant Failure Rates

How to Calculate Availability

- $A = \frac{Uptime}{Downtime + Uptime}$

- $A = \frac{MTBF}{MTBF+MTTR}$

  - The impact of MTBF and MTTR

Many Factors in Availability Calculation

Designing and implementing a high available network:

- Hardware
  - Hardware failures like I/O errors, hard disk failures, memory parity errors, network hardware failures
- Software
  - Software errors like bugs in source codes, system overload, resource exhausting
- Environmental Faults
- Human Errors
  - Mostly occur as a result of changes

HW/SW factors in Availability Calculation of a Component

Calculating Hardware Availability:

- MTBF
  - Can be obtained by the vendor for the off-the-shelf components or the hardware team for the in-house component
- MTTR
  - Service contract response time

Calculating Software Availability:

- MTBF
  - Multiplying the defect rate by the size of program executed per second
- MTTR
  - Mean time taken to reboot or debugging

Human Errors and Environmental Factor

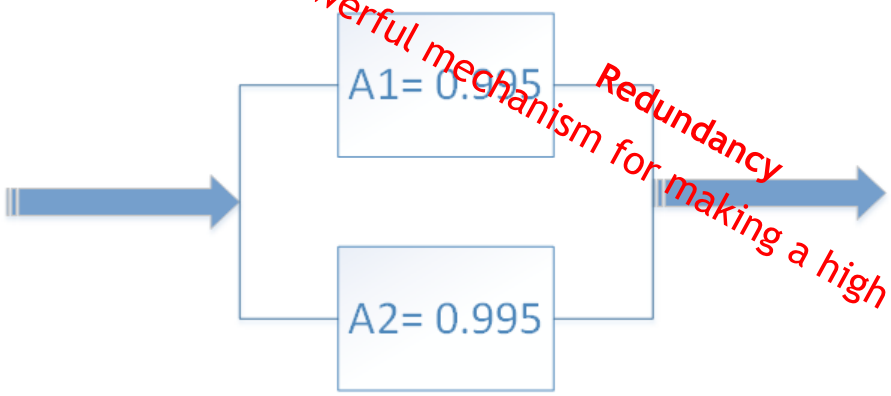
- Environment
  - 29 minutes down time for power loss per year, get the availability of 0.999945
  - Can be increased by backup power devices
- Human Errors
  - experienced
  - Task complexity: either it is simple or hard, routine or non-routine
  - Stress factor: how much time is available
  - If there is any procedural guidance for doing the job

Availability in a Serial System

Availability = A1 × A2 = 0.990025

What happens if A1 is high but A2 is low?

Availability in a Parallel System



A powerful mechanism for making a high available system

Redundancy

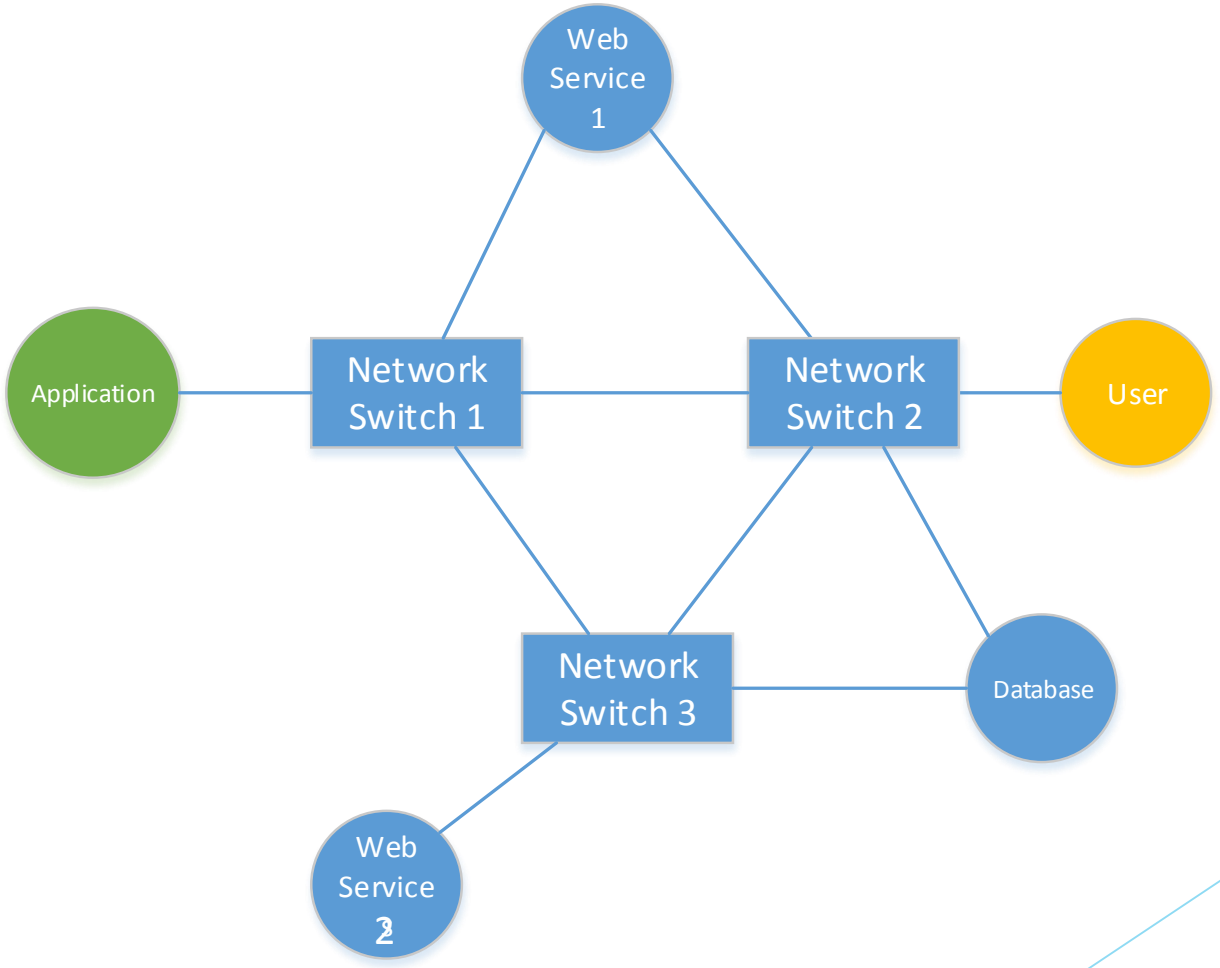
Unavailability = (1-A1) × (1-A2) = 0.000025

Availability = 1 – Unavailability = 0.999975

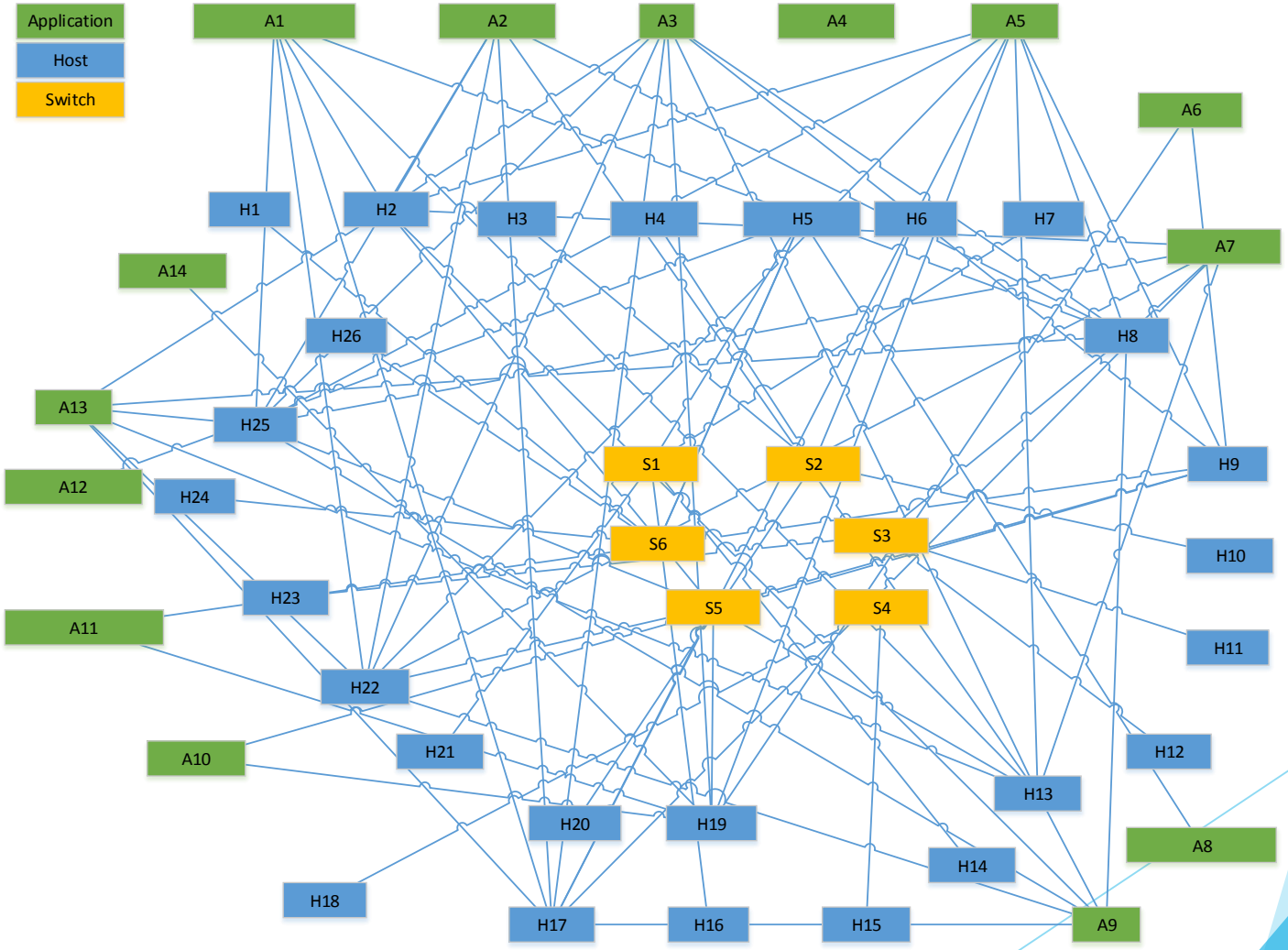
So far…

- We know what the availability is
- We can calculate the availability of a single (independent) component
- We can calculate the availability of dependent components with simple relations

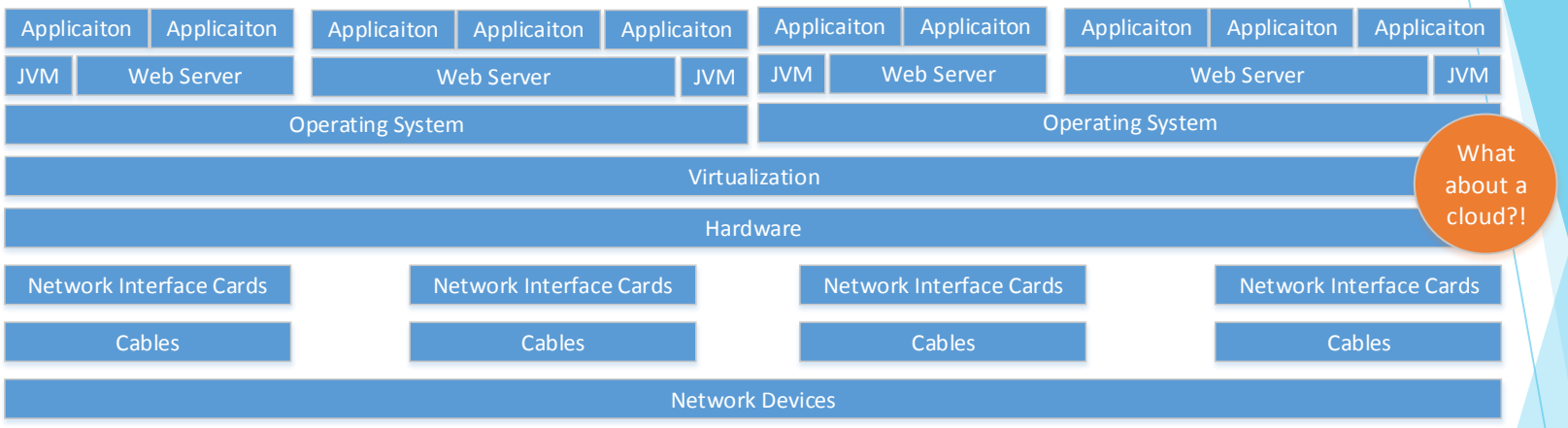
Application Dependency



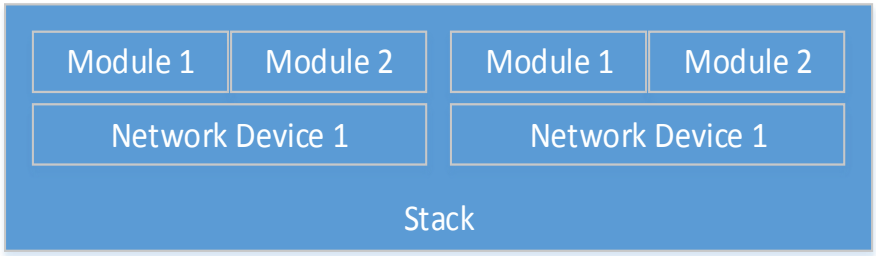
Real life example!



Different Layers



Network Device



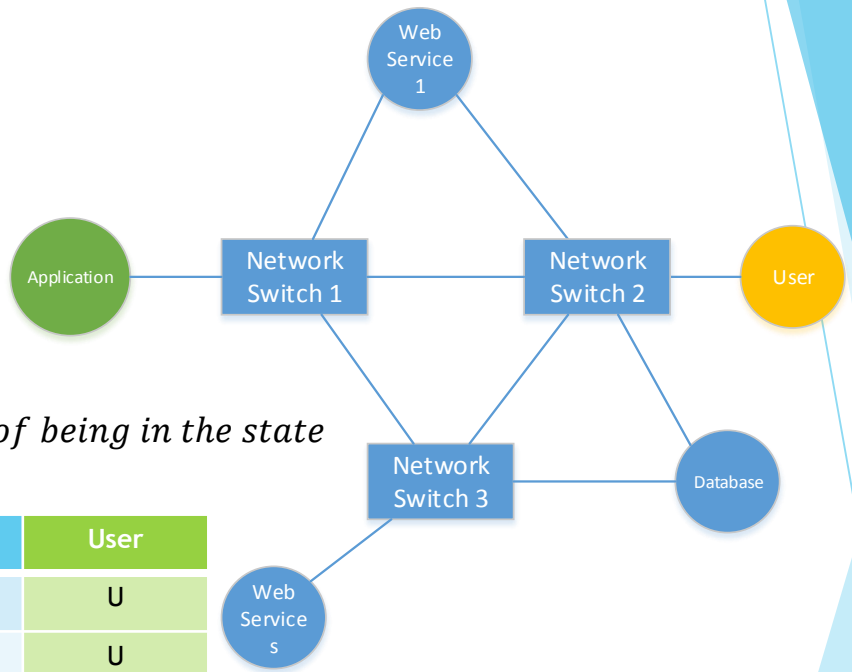
What may go wrong?

- An application may have bugs
- An application server may run out of resources
- An operating system may fail
- A hard disk may fail
- A server hardware may fail
- A network cable may get disconnected
- A switch may malfunction
- An administrator may make a mistake while configuring something
- You may have power outage
- Your cooling system may fail
- And …

- Are these happening one at a time?!

The approach

$$A(System) = 1 - \sum_{Unavailable\ States} Probability\ of\ being\ in\ the\ state$$



APP	DB	WS1	WS2	NS1	NS2	NS3	User
U	X	X	X	X	X	X	U
A	U	X	X	X	X	X	U
A	A	X	X	X	U	X	U
A	A	X	X	U	A	X	U
A	A	U	U	A	A	X	U
Otherwise							A

A	Available
U	Unavailable
X	Don't Care

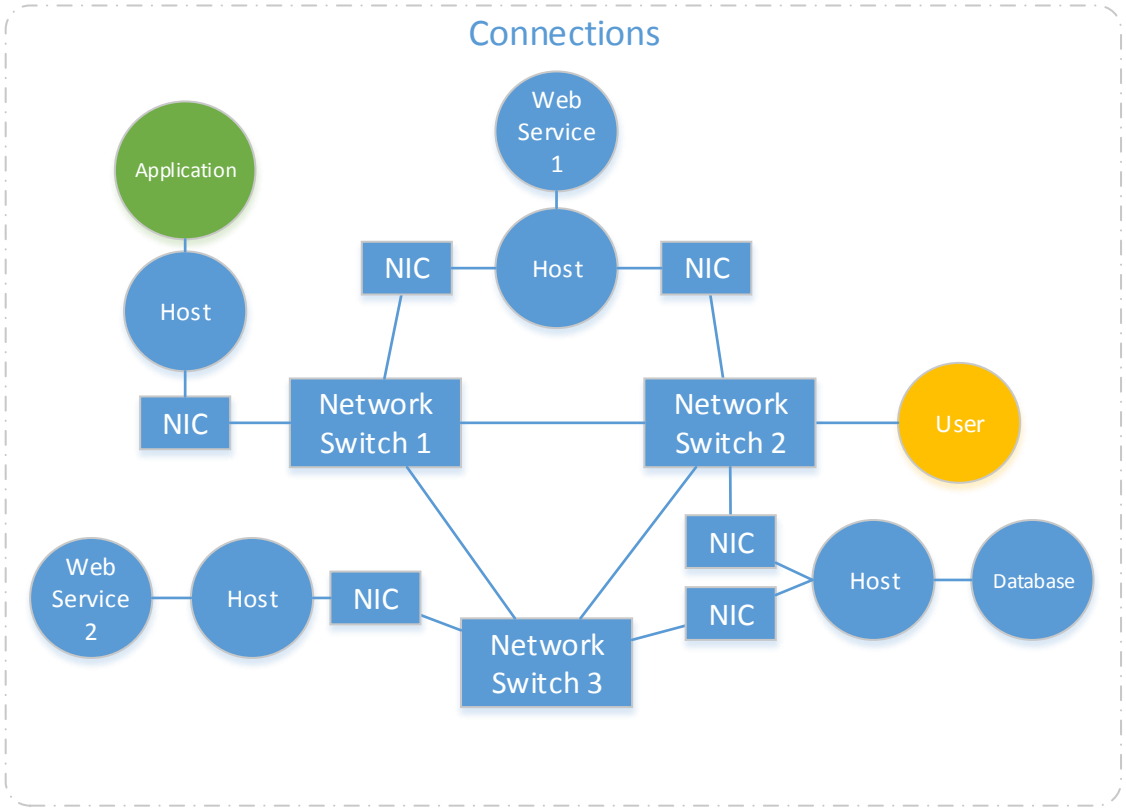
In order to find failures

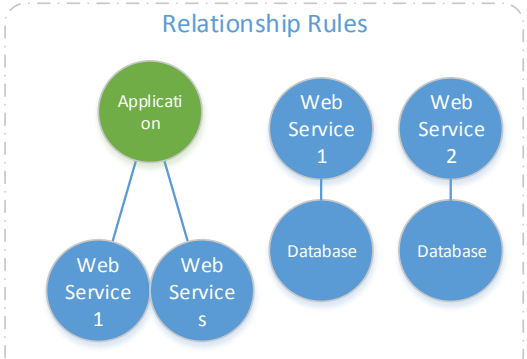
- Choose what layers you want to include in your calculations
  - You may want to skip a level or integrate it into others
- Partition those layers into two categories:
  - Network Category: All those providing network connectivity
  - End point Category: All those are not engaged in network connectivity
- Divide End Points into two subcategories:
  - Application itself
  - Containers (no dependency rule)
- And Network subcategories are:
  - Container
  - Interface

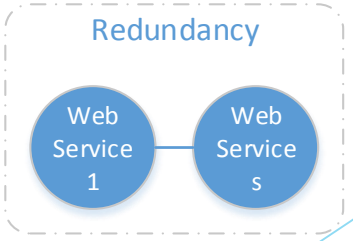
The rules are:

- A container will fail, if either of its components fails
- An application will fail if:
  - Itself fails;
  - Its container fails;
  - What it depends on had failed;
  - There is no connectivity between the application and what it depends on.
- An interface will fail if it fails!

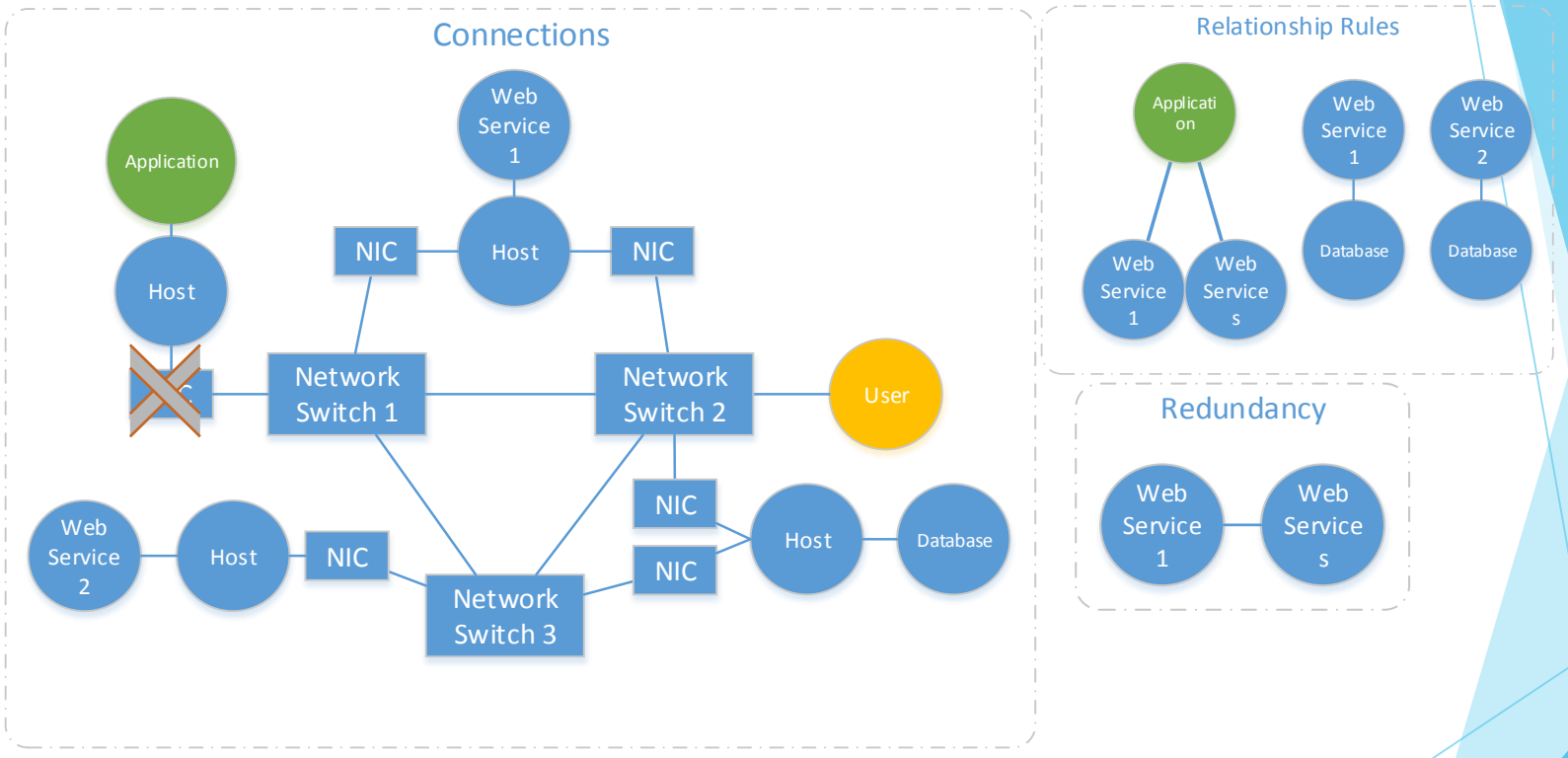
Situation Modeling







Calculation Steps



Inject Fault(s)

If not all rules are satisfied, it is a Fail State

Calculate the probability

Add to the sum

Test Case

- Getting AITIH data for a part of a business application in csv format

- appT
- appCSA
- appEUI
- appEBC
- appEDB
- appCS
- appkia

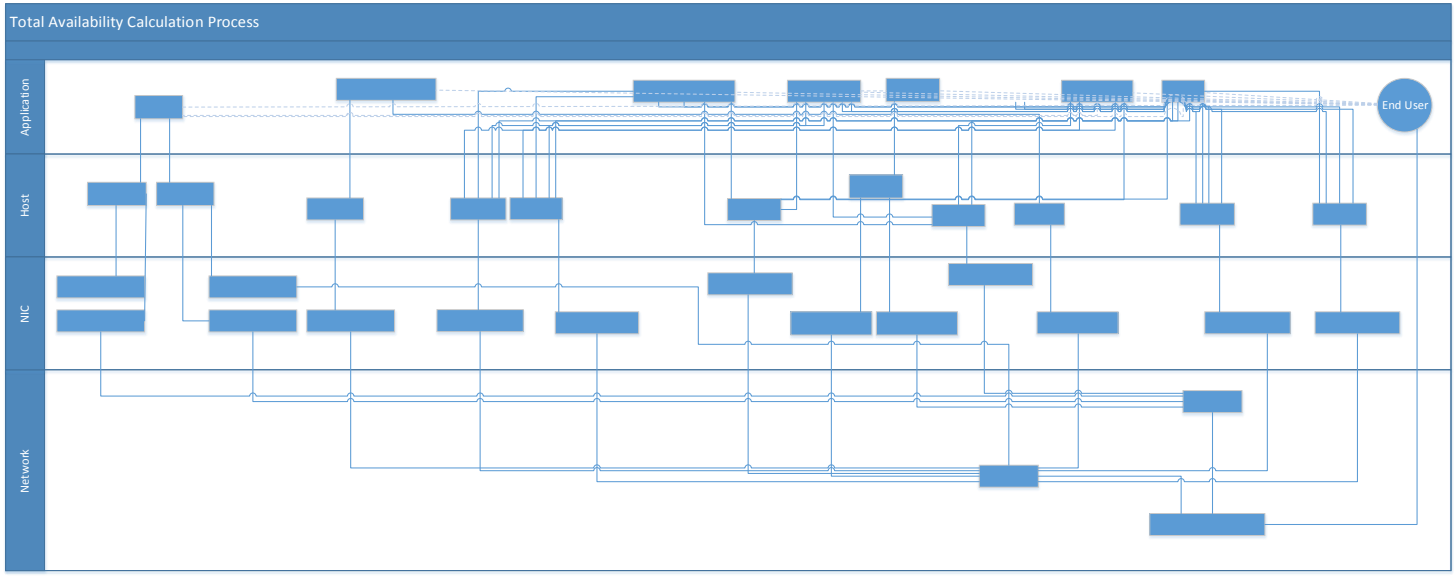
Application - Hosts

Application Name	Host	No. of Clones Running
appCSA	hst01	1
appCSA	hst02	1
appEUI	hst03	5
appEUI	hst04	5
appEUI	hst05	5
appEBC	hst06	3
appEBC	hst07	3
appEBC	hst08	3
appEBC	hst03	3
appEBC	hst04	3
appEBC	hst05	3
appCS	hst06	1
appCS	hst07	1
appCS	hst08	1
appCS	hst03	1
appCS	hst04	1
appCS	hst05	1
appkia	hst06	1
appkia	hst07	1
appkia	hst08	1
appkia	hst03	1
appkia	hst04	1
appkia	hst05	1
appT	hst09	1
appT	hst10	1
appEDB	hst11	1

Application Dependencies

Application Name	Database Service	Hosted on
appCS	appT	hst09
appkia	appT	hst10
appEBC	appEDB	hst11

All components together



The input data

- **apps.csv**

  hst01,appCSA,1

  hst02,appCSA,1

- **netnods.csv**

  Switch_1,Switch_3

  Switch_3,Switch_2,Switch_1

- **hostnicsw.csv**

  hst08,eth2,Switch_1

  hst07,eth2,Switch_1

  hst01,eth2,Switch_1

- **dep.csv**

  appCS,appT

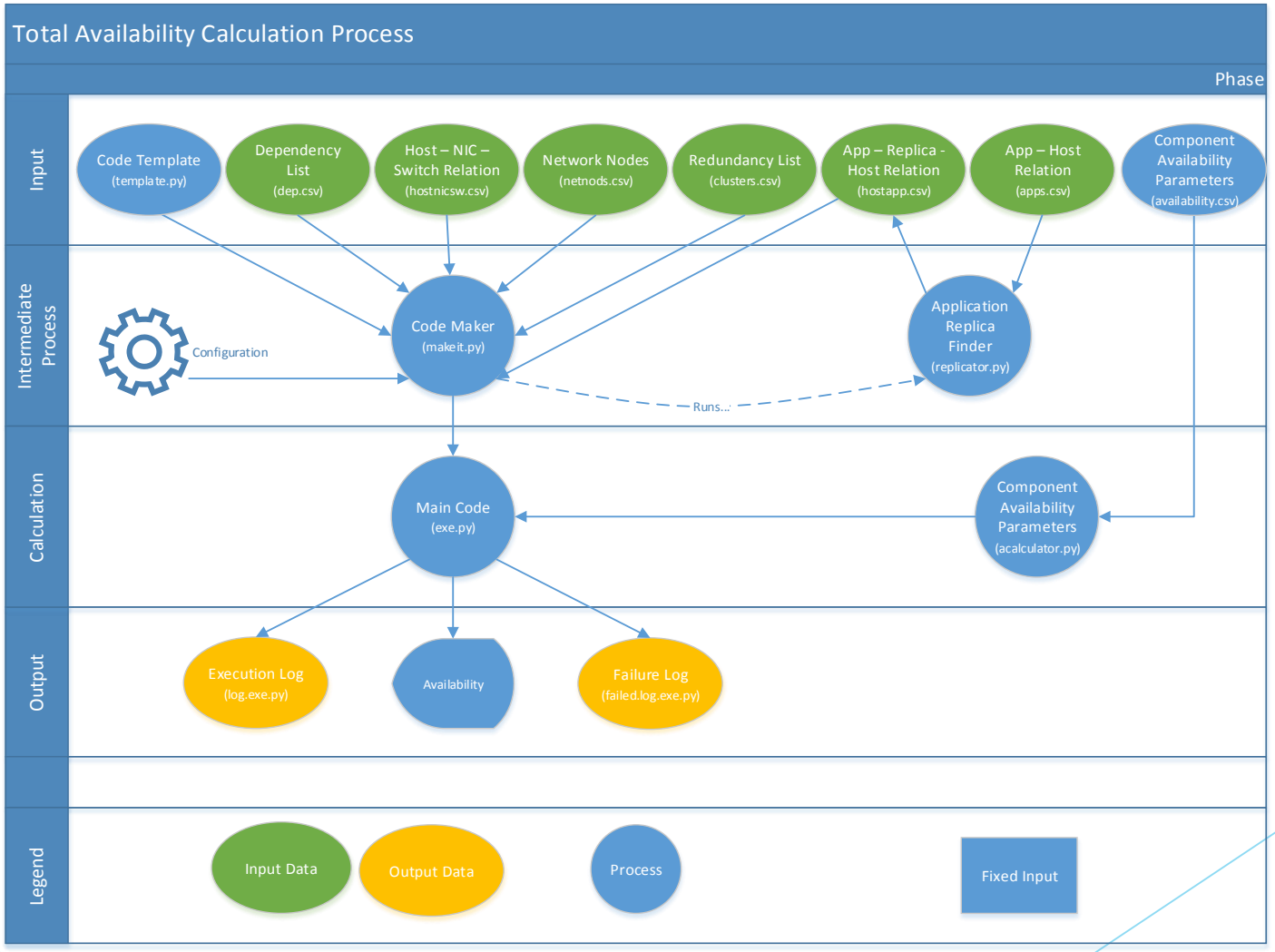
  appkia,appT

- **availability.csv** (A random number between 0.9999 and 0.999997)

  hst08->eth2,,,0.999944

  hst09,,,0.999972

The Process


Total Availability Calculation Process
Phase
Input
Intermediate Process
Calculation
Output
Legend
Code Template (template.py)
Dependency List (dep.csv)
Host – NIC – Switch Relation (hostnicsw.csv)
Network Nodes (netnods.csv)
Redundancy List (clusters.csv)
App – Replica - Host Relation (hostapp.csv)
App – Host Relation (apps.csv)
Component Availability Parameters (availability.csv)
Configuration
Code Maker (makeit.py)
Application Replica Finder (replicator.py)
Runs...
Main Code (exe.py)
Component Availability Parameters (acalculator.py)
Execution Log (log.exe.py)
Availability
Failure Log (failed.log.exe.py)
Input Data
Output Data
Process
Fixed Input

Results - Summary

Maximum Simultaneous Faults	Number of Failure Scenarios	Total Availability
1	5	99.9781476669 %
2	280	99.9780993579 %
3	8,192	99.9780993065 %
4	136,153	99.9780993064 %
5	1,769,375	99.9780993064 %
6	17,919,053	99.9780993064 %

Results - Two Simultaneous Fault Scenarios

#	Component A	Component B	Desc
1	'hst09'	'appT.REP2'	appT.REP2 on hst10
2	'hst09'	'hst10'	
3	'appT.REP2'	'appT.REP1'	appT.REP1 on hst09
4	'appT.REP1'	'hst10'	
5	'hst02'	'hst01->eth2'	
6	'hst02'	'appCSA.REP1'	.REP1 on hst01
7	'hst02'	'hst01'	
8	'hst01->eth2'	'appCSA.REP2'	.REP2 on hst02
9	'hst01->eth2'	'hst02->eth2'	
10	'appCSA.REP2'	'appCSA.REP1'	
11	'appCSA.REP2'	'hst01'	
12	'appCSA.REP1'	'hst02->eth2'	
13	'hst02->eth2'	'hst01'	
14	'Switch_2'	hst11->eth1	
15	'hst11->eth1'	'hst11->eth2'	

Result - A little analysis

- **Criticality Function:**

  CF(component) =
  Number of times it appeared as a cause of system failure
  * (1- Availability(component))

- **Most Critical Components: ['Switch_1']**

Assumptions

- Each single node's independent Availability is either pre-calculated, or its MTBF and MTTR parameters are present. If none were present, a random number between 0.9999 and 0.999997 were assigned as the availability.
- Whenever there is a physical network path between two network nodes, it illustrates a network connection between them. In other words, no network segmentation exists in upper layers.
- Physical connectors (like cables) are considered as always available.
- Network devices are seen as a single component even if they are modular.
- There is no virtualization involved.
- There is only one web application on each web server.
- Hosts include: Operating System and Host hardware (except for the NIC).
- All network cards of a server are able to take-over other cards.
- In the network layer, Redundancy is made by using separate paths. There is no Stacked Switch.
- *Environmental and Human Related Factors are rolled out for simplicity*

To Sum-up, we saw

- What availability means
- How to calculate it for a standalone (independent) component
- How to calculate it for simple dependent components
- A method of calculating availability in a complex system
- An example of such calculation