AN EFFICIENT PERFORMANCE TESTING METHODOLOGY FOR TELECOMMUNICATION SUBSYSTEMS

Objectives

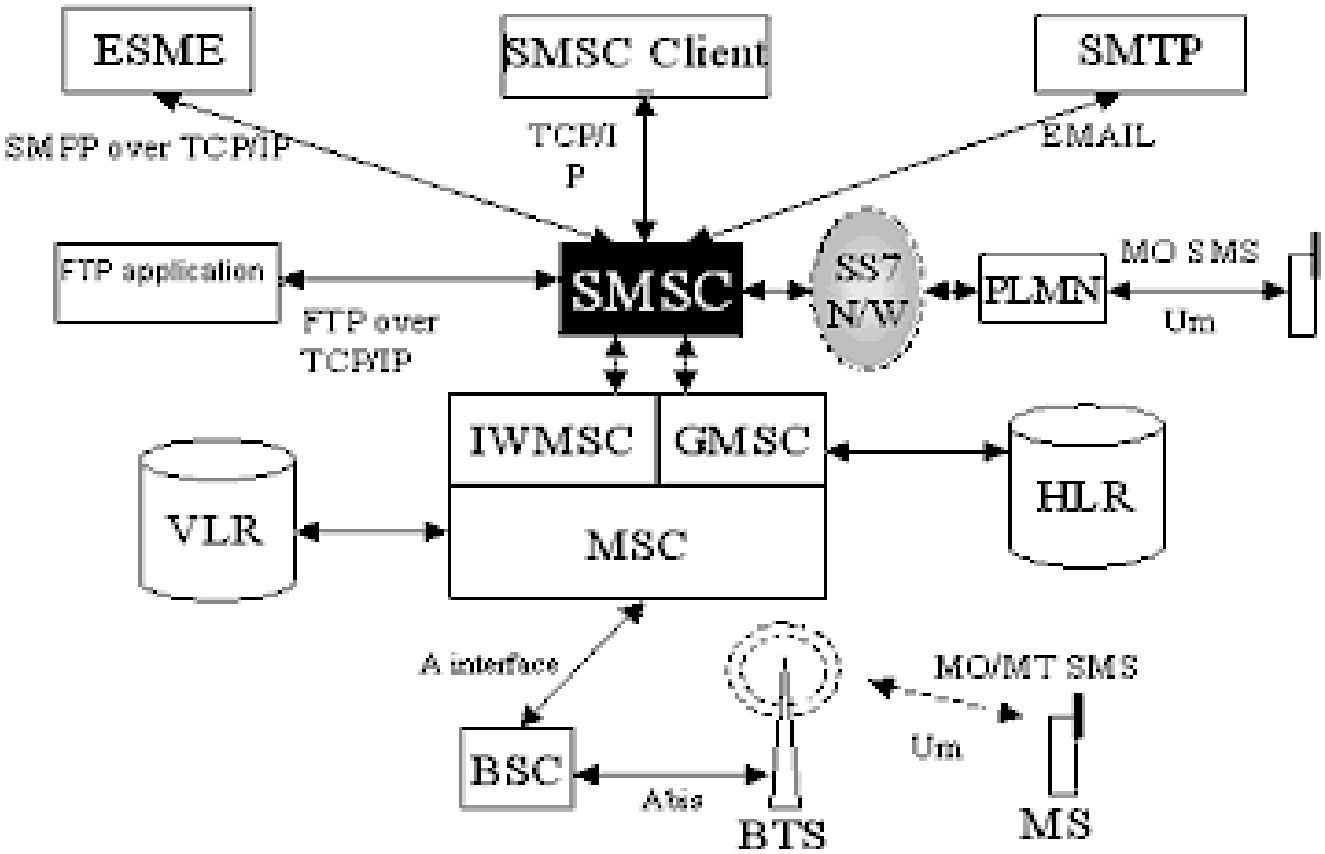
- Introduction
- Designing Performance Test suite
- Three stage methodology for Benchmark testing
- Case Study

Introduction

Large telecommunication software subsystems are characterized by requirements of very high reliability, availability

A combination of benchmark testing, peak-rest test and stress test found to be significantly effective to pin-point issues with respect to performance, reliability and scalability of such legacy system

What is SMSC


ESME
SMSC Client
SMTP
SMPP over TCP/IP
TCP/IP
EMAIL
FTP application
SMSC
SS7 N/W
PLMN
MO SMS
Um
FTP over TCP/IP
IWMSC
GMSC
HLR
VLR
MSC
A interface
MO/MT SMS
Um
BSC
Abis
BTS
MS

Designing Performance Test suite

In case of SMSC followings need to be taken account while designing a system for performance testing

- System Configuration
- Message Flow Settings
- Performance Settings
- Network factors
- Hardware Configuration
- Stress Factor

Designing Performance Test suite (cont…)

Performance test-suites should ideally explore the entire system boundary and monitors the sub-system behavior for each settings with respect to

- » Invalid Functionality
- » Throughput of the system
- » Response time of a system with peaks, bursts etc.
- » CPU utilization of the system
- » Memory utilization and leak

Three stage methodology of Benchmark testing

- Phase 1:

  System with default configuration
- Phase 2:

  System with special configuration
- Phase 3:

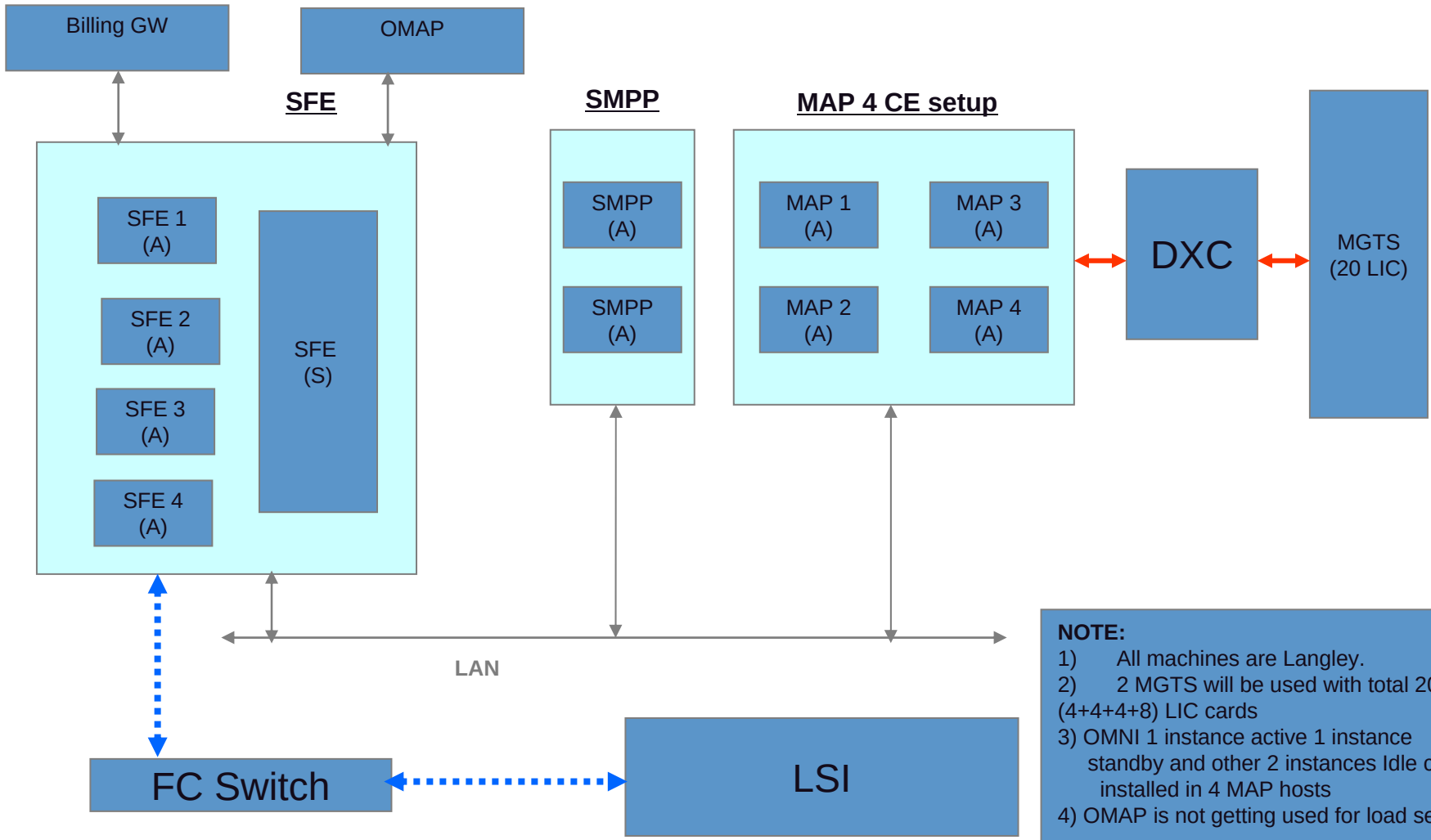
  System’s measurements

Case Study

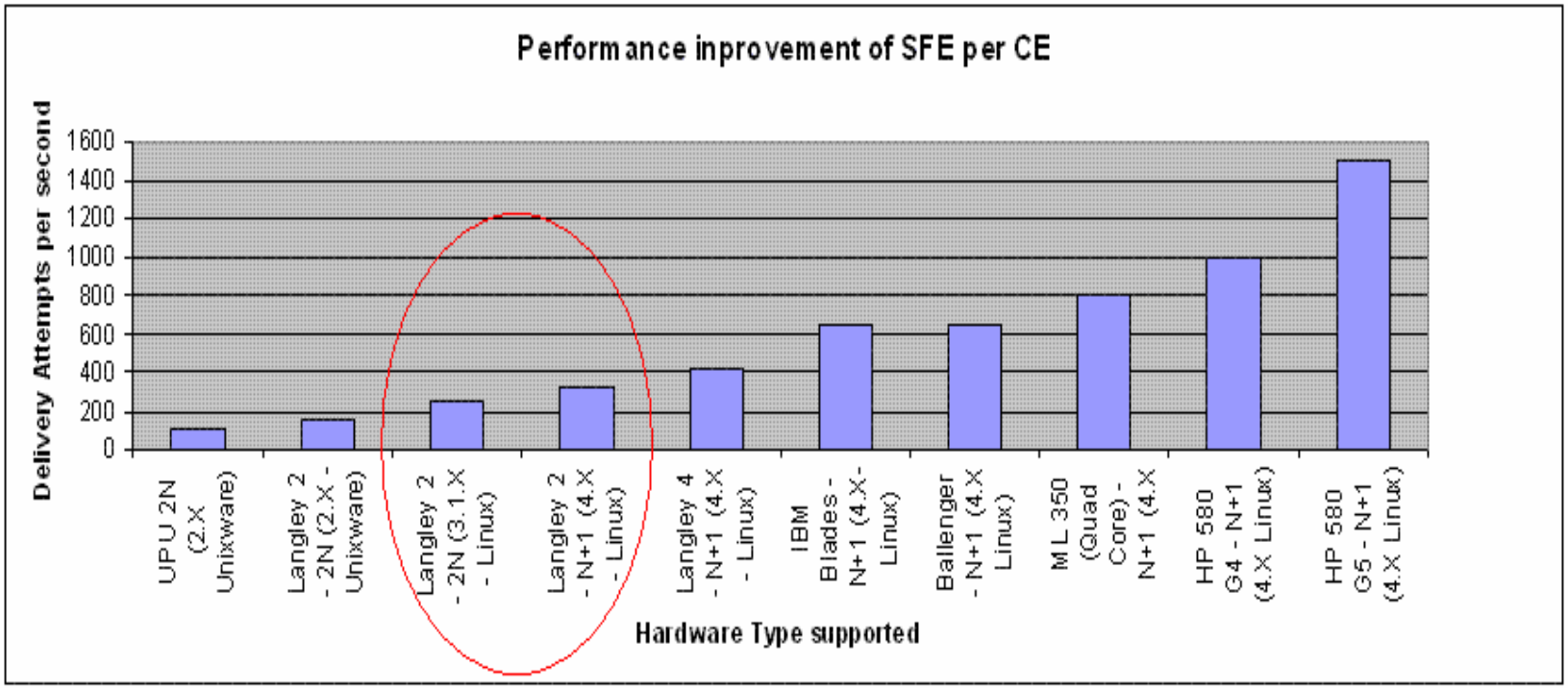
Using combination of benchmark testing along with stress and peak-rest testing, reliability and High-Availability of a fault-tolerant SMSC system can be certified as follows:

- Percentage of Idle CPU
- Error and Warning generated in the system
- Memory Usage
- Number messages holding in the system
- Number of open dialogs
- Queue Status
- Free Disk space

Performance model to support 4.5M message

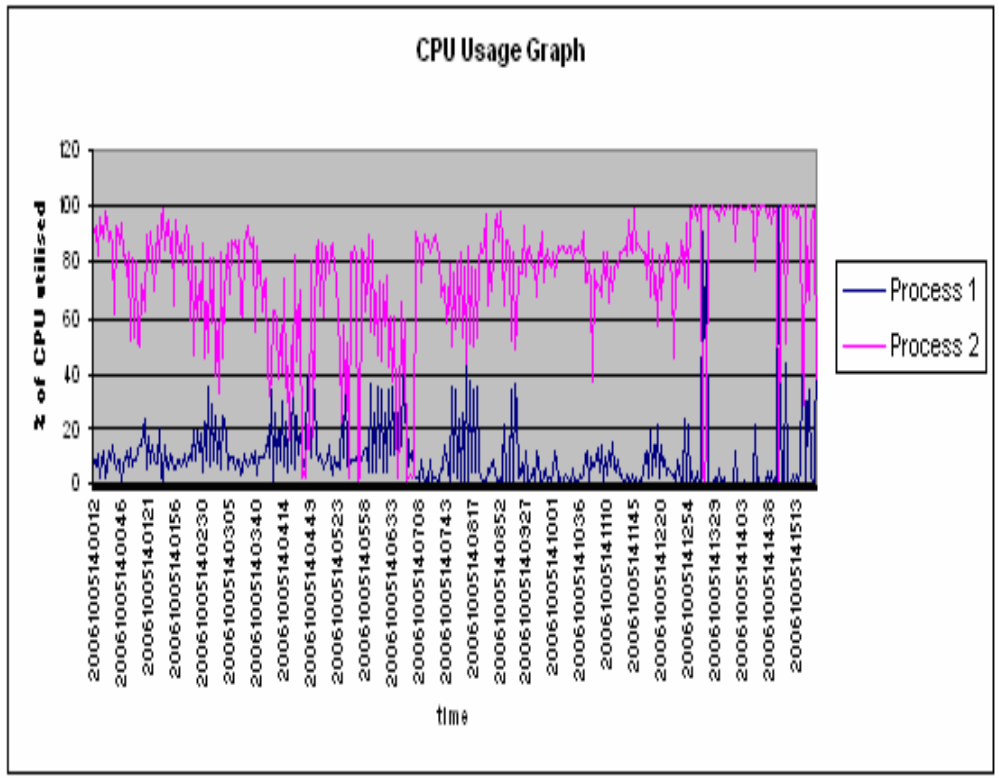


Roadmap Data of SMSC system



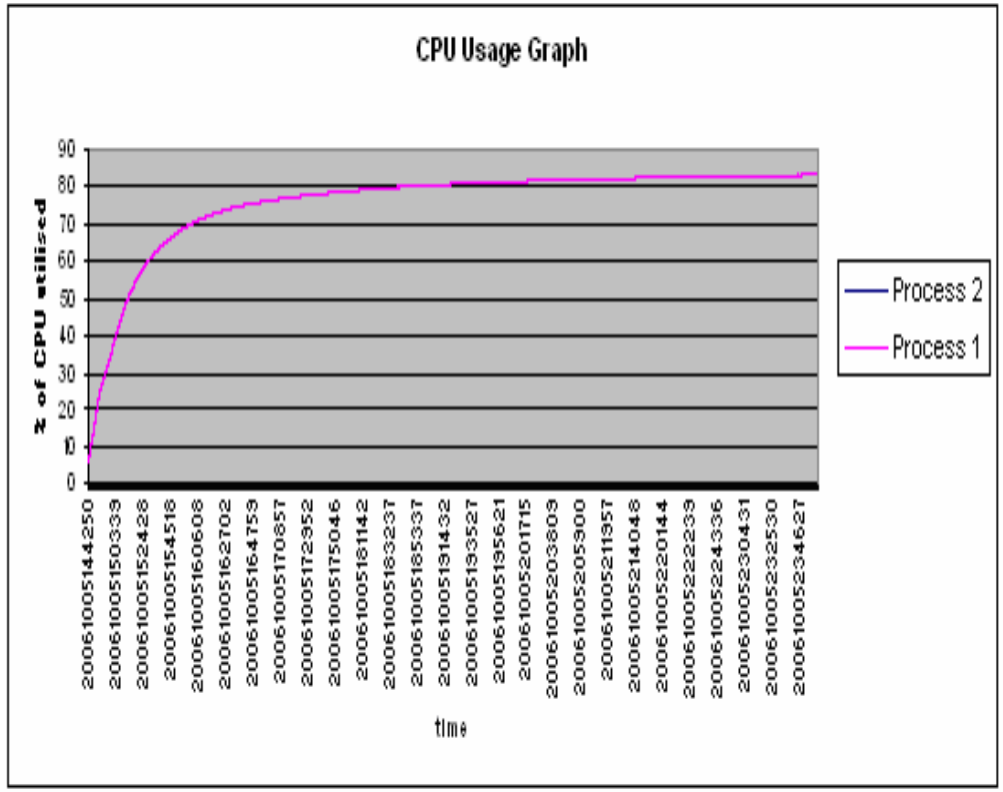
Preparation of performance model

A robust system with UPU hardware supporting UnixWare system gets overloaded and arises race condition for a load of 400 SDAPS



Preparation of performance model (cont…)

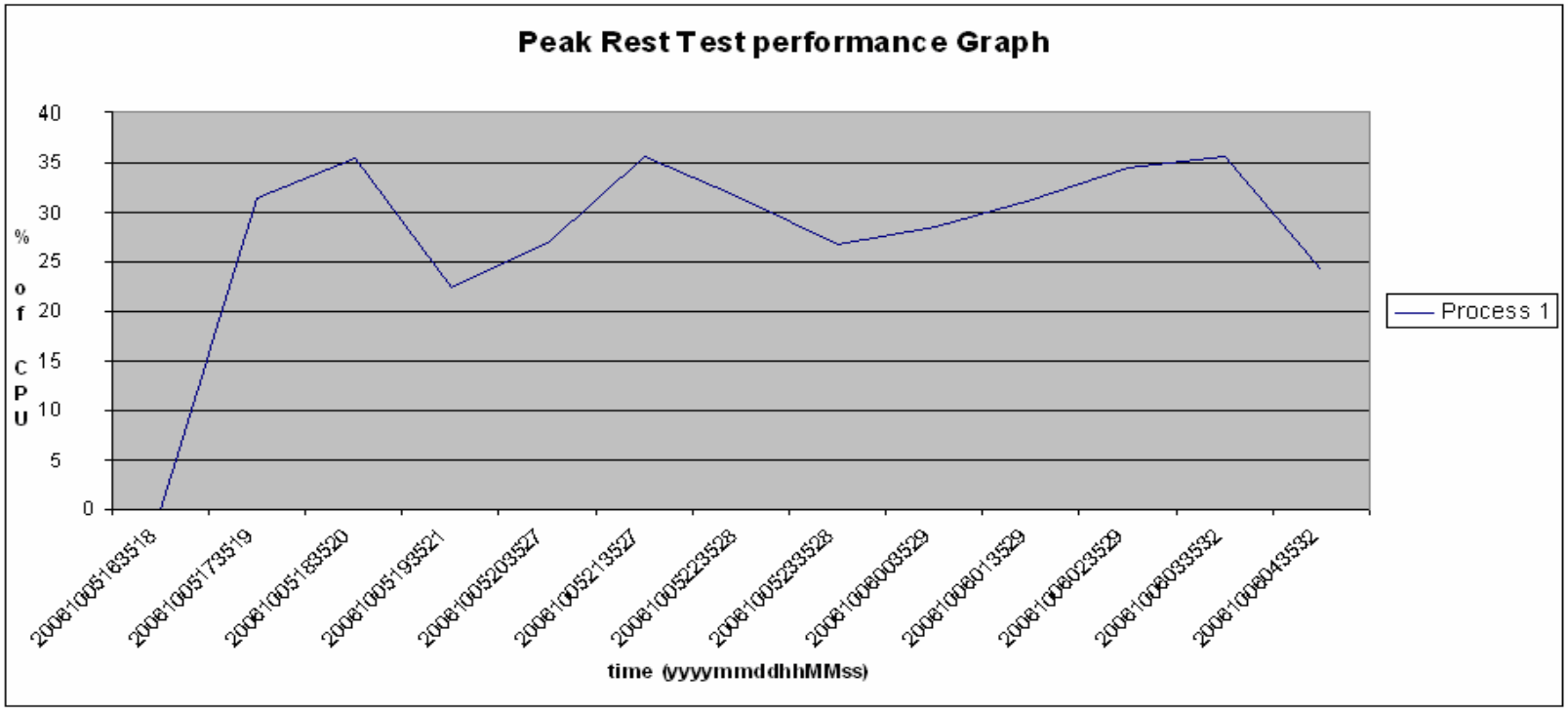
As an Initial step system architect redesigned the product and change the product platform from UnixWare to Linux and certify the system for 600 SDAPS



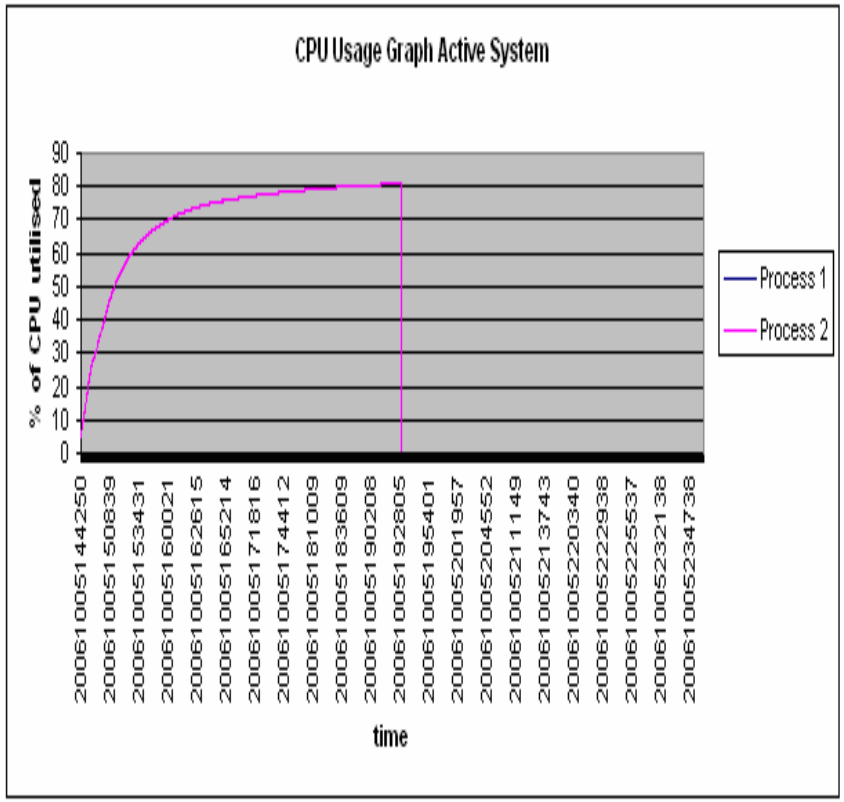
But now to achieve a figure of load 1000 SDAPS and holding capacity of message 4.5 M at peak hours system got migrated from UPU to Langley II. Initial testing shows system now can bear a load of 1000 SDAPS

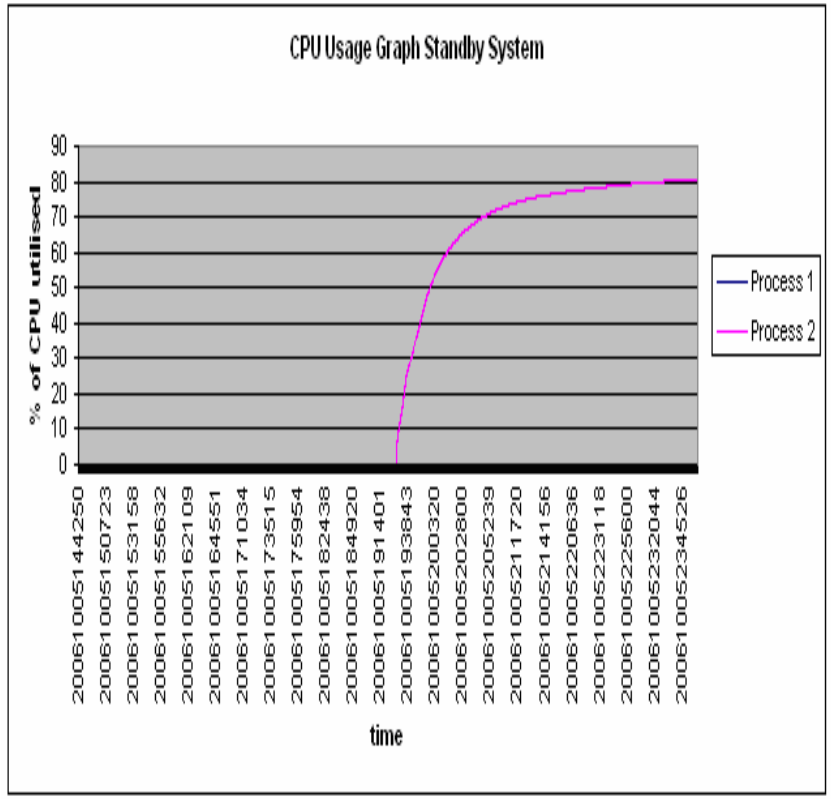
Reliability marking

Peak-rest and stress test performed together in order to mark reliability of the system



High-Availability (HA) marking





Redundancy of the system gets checked with respect to throughput in order to mark High-Availability

Observation

- **Percentage of Idle CPU –**

  ~ 20% and it is constant throughout the stabilization period

- **Error and Warning generated in the system –**

  No errors/ warnings or unexpected event noticed

- **Memory Usage –**

  There is no rise/drop in the memory once stabilization achieved

- **Queue status –**

  messages is not waiting in the queue

- Free disk space –

  ~ 40% under all circumstances

Summary

With the combined methodology outlined in this presentation, a 3X improvement in execution cycle-time was achieved for a large system (2 man-months vs 6 man-months with traditional approach), without additional hardware resources

Thanks