
# Supporting Operations Personnel Through Performance Engineering

Len Bass

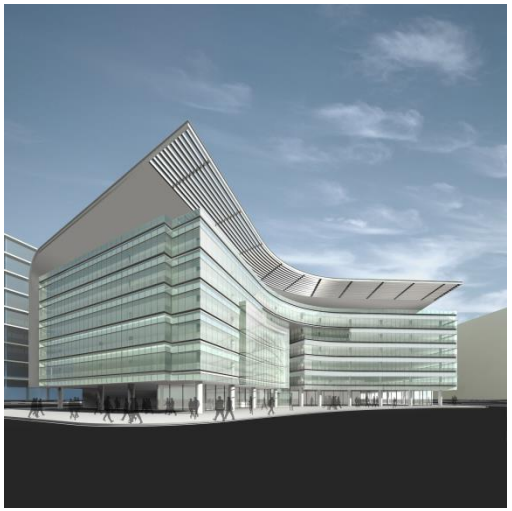
Australian Government
Department of Broadband, Communications and the Digital Economy
Australian Research Council

NICTA Funding and Supporting Members and Partners

# About NICTA

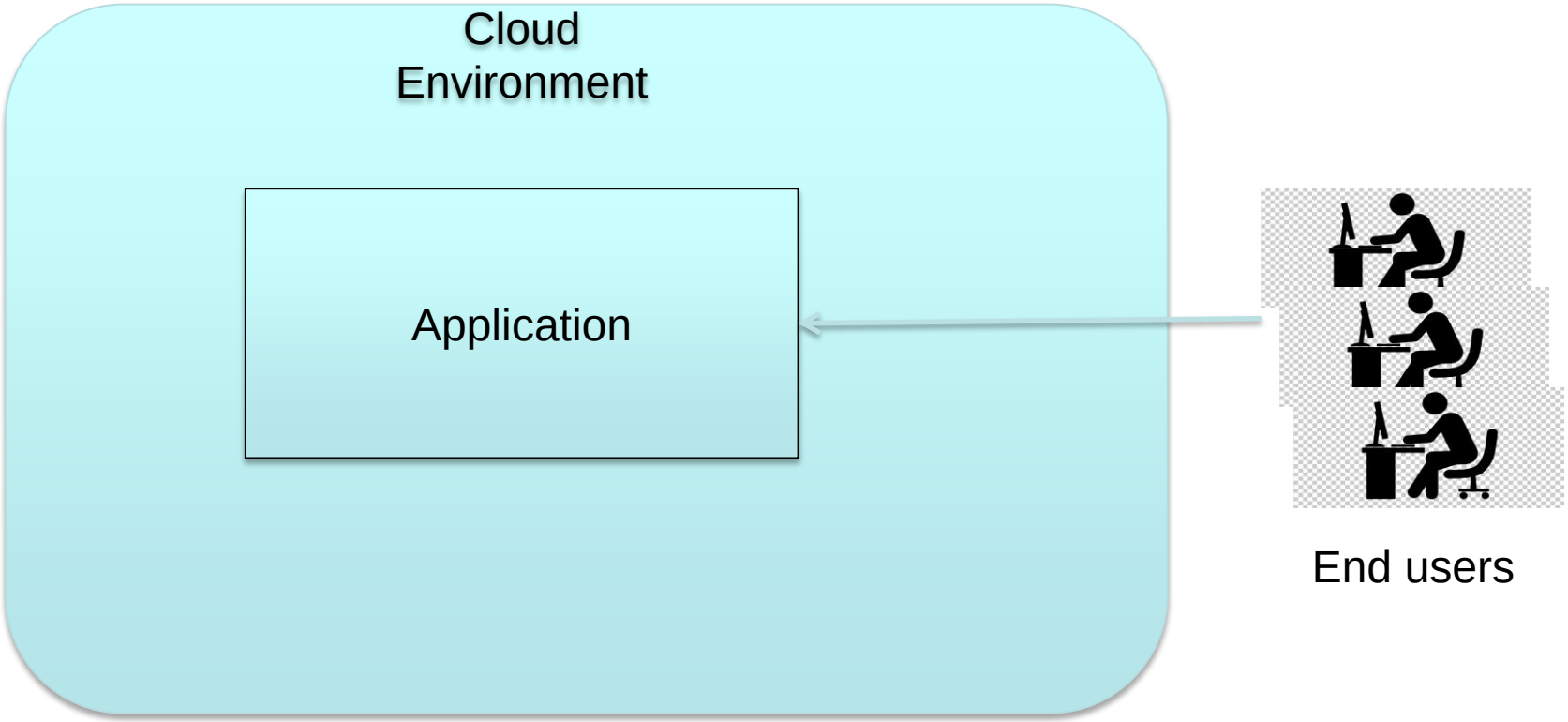
## National ICT Australia

- Federal and state funded research company established in 2002
- Largest ICT research resource in Australia
- National impact is an important success metric
- ~700 staff/students working in 5 labs across major capital cities
- 7 university partners
- Providing R&D services, knowledge transfer to Australian (and global) ICT industry



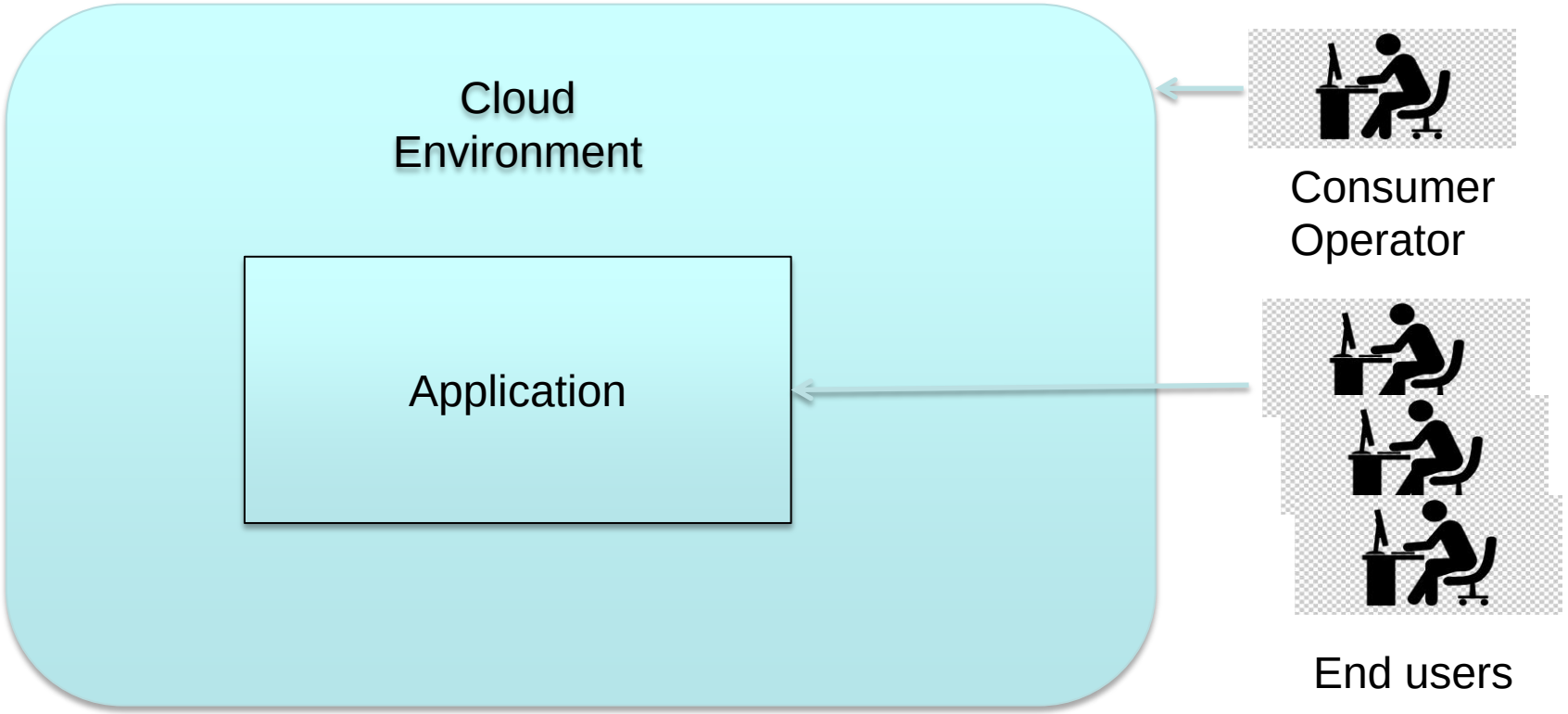
NICTA technology is in over 1 billion mobile phones

# Traditional View from Performance Engineers



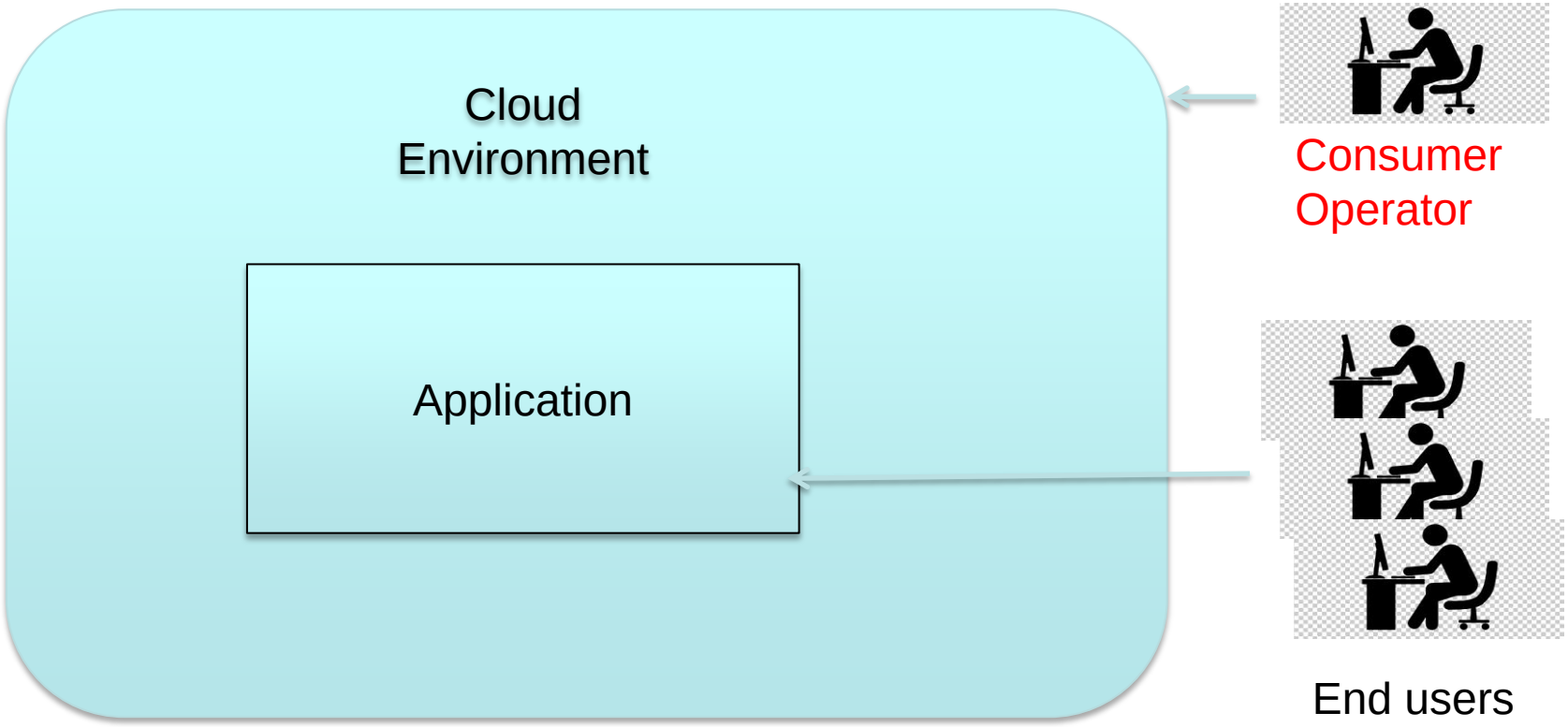

Traditionally, the performance community has viewed systems as having users and existing in an environment. The motivating question has been: How can I, in this world, improve the performance of applications?

# A Broader View




Applications are not only affected by the behavior of the end users but also by actions of operators who control the environment for a consumer's application.

# My message: Consider the operator in this picture



# Business Context

"Through 2015, 80% of outages impacting mission-critical services will be caused by people and process issues, and more than 50% of those outages will be caused by change/configuration/release integration and hand-off issues."

Change/configuration/release integration and hand off are all operations issues.

Gartner - http://www.rbiassets.com/getfile.ashx/42112626510

# Outline

- **Overview of operations space**
  - **What do operators do?**
  - **What can go wrong with what they do?**
- Some results we have achieved
- Operations through performance engineering glasses

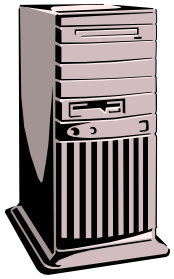
# What do operators do?

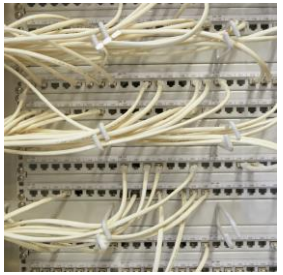
- Monitor and control data center/network/system activity
  - Install new/upgraded applications/middleware/configurations/hardware
- Support business continuity through back ups and disaster recovery

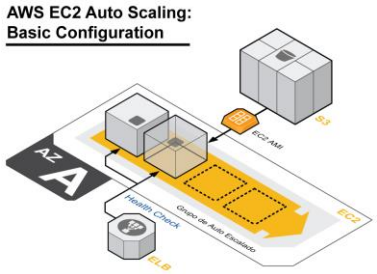

**Akamai's NOC in Cambridge, Massachusetts**

# Monitor and Control

- Data Center
  - Total number and type of resources (may be virtual)
    - Processors
    - Storage
    - Network
- Network
  - Intrusion detection
  - Routing
  - Loading
- System
  - Allocation to resources
  - Install/uninstall
  - Configure







# What can go wrong with monitor and control?

NICTA

## Everything that was on previous slide.

- Failure
  - Installations can fail
  - Resources fail and must be replaced
- Overload
  - Resources are over/under loaded and must be supplemented/removed
  - Networks get overloaded and routing must be changed
- Error
  - Routing may be incorrectly specified
  - Allocation of systems to resources may be incorrect
  - Configurations can be incorrectly specified


# Install new/upgraded applications

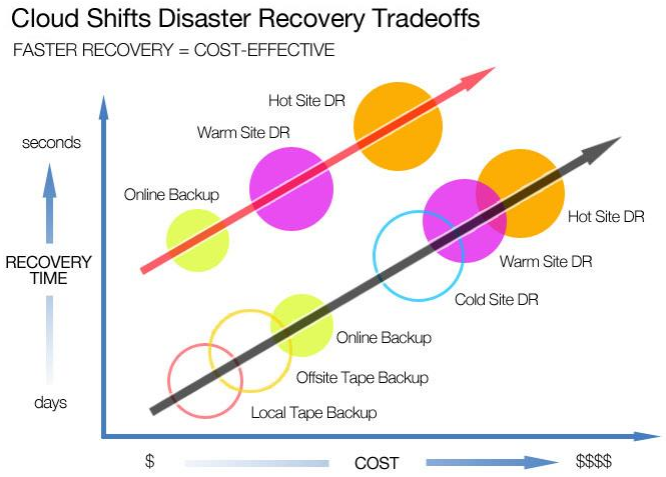
- Configuration for applications
- Synchronizing state for upgraded applications
- Testing new/upgraded applications in target environment
- Allocating resources for new version

# What can go wrong with installing apps?


- Again its everything.
  - Configuration can be misspecified
  - Cut over to new version may leave inconsistent state
  - Rolling upgrade may introduce “mixed version race condition”
  - Upgrade to level N of the stack may break software in level >N of the stack
  - Testing environment may not appropriately mirror real environment

# Supporting business continuity

- Disasters happen – natural or human causes
- Backing up data provides recovery possibility
  - Lag between last version backed up and when disaster happens
  - In the Cloud, backing up large amounts of data to different geographic regions takes time.



# Hand offs

- Problems can arise when a shift changes
  - What problems did old shift deal with?
  - What problems were totally solved?
  - What problems were partially solved?
  - What operations activities are currently ongoing?

# Operations is a target rich environment


- There are many existing tools. Operation of data centers would not work without tools
- Much room for improvement (see Gartner quote)
- Some general approaches for improvement
  - Make software systems operations process aware. E.g. make them perform operations that humans might otherwise do.
  - Model operations processes and systems using a single model. I.e. model analysis will provide opportunities for detecting trade offs between human and automated activities.I will talk about our solution to the mixed version race condition.
  - Make tools process incident aware. Eg upgrade or shift change.

# Outline

- Overview of operations space
- **Some results we have achieved**
  - Disaster Recovery product
  - Operator undo
    - Prevention of mixed version race condition
- Operations through performance engineering glasses

# Disaster Recovery

- Clouds fail – Amazon had three outages in 2011 that affected whole availability zones or regions.
- NICTA has a subsidiary (Yuruware) with a non-intrusive disaster recovery product (Bolt).
- Bolt copies data periodically to a back up region.
- Bolt utilizes sophisticated data movement techniques to reduce time required to back up
- This is an insurance policy.

# Upgrade

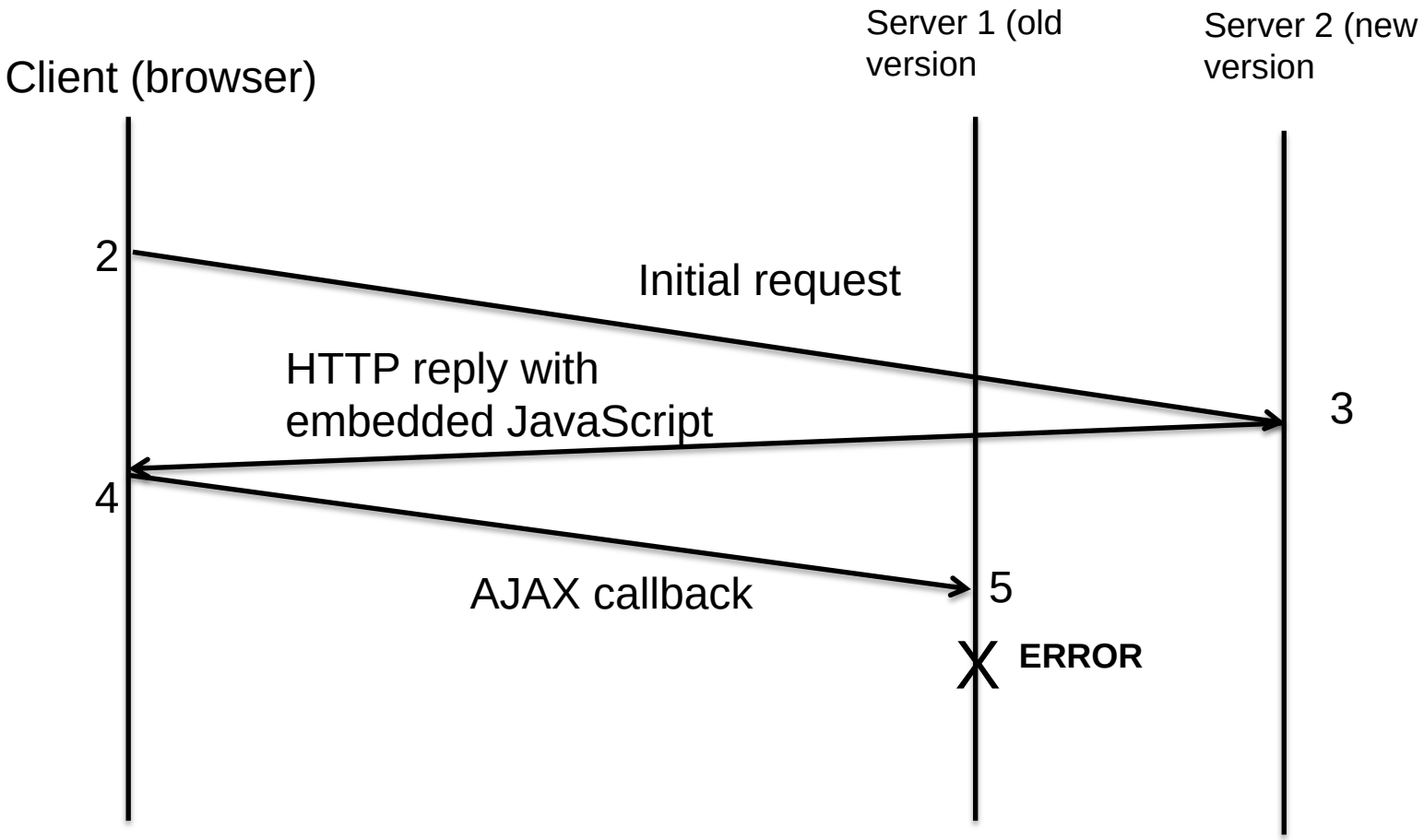
- You are installing version N+1 of an application and replacing version N.
- May take hours to install a new version on 1000s of machines
- Several different strategies

# Strategies

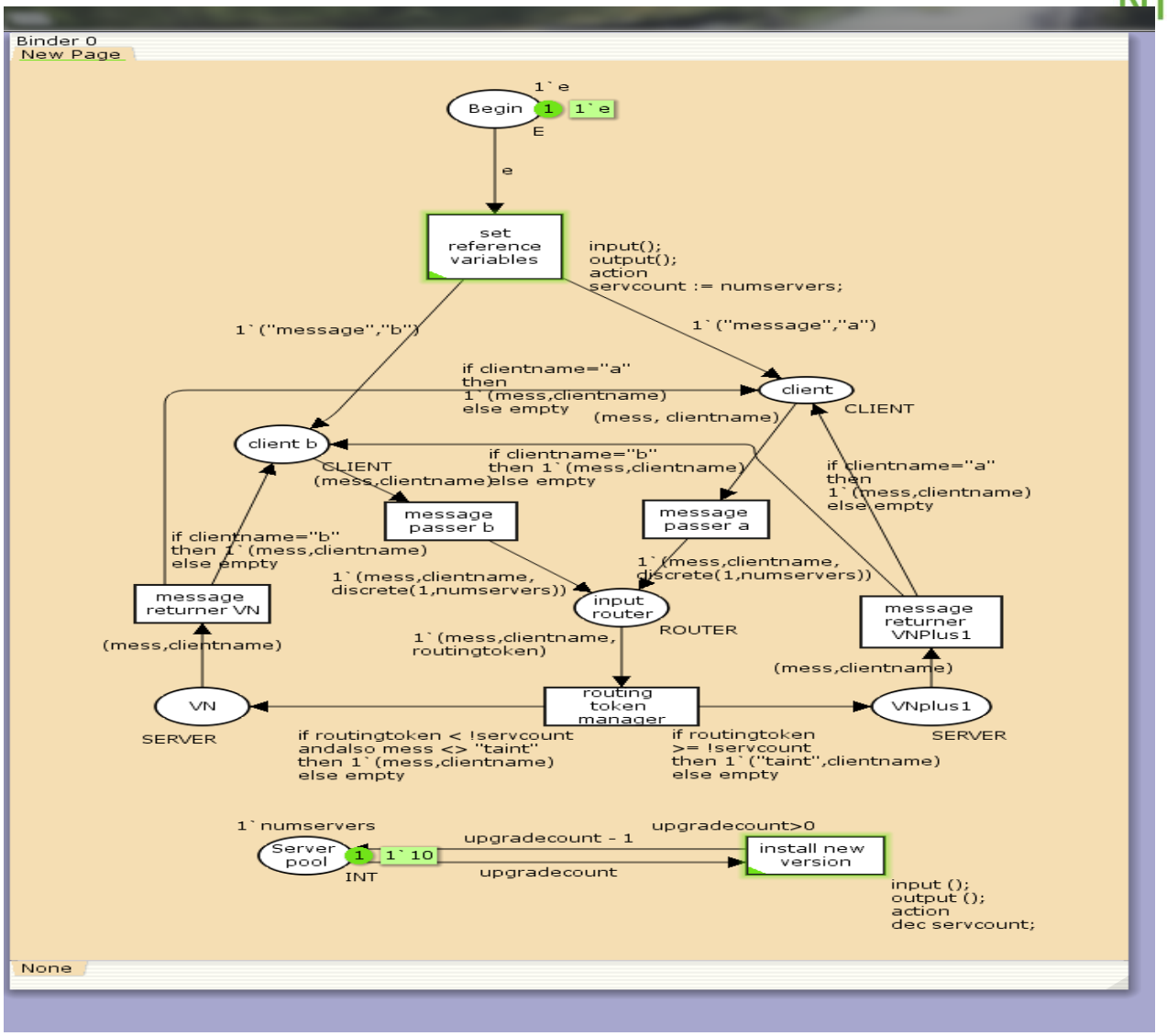

- **Big flip**– install version N+1 and keep all servers running version N active until an equivalent number of version N+1 servers are available. Switch version N off and switch version N+1 on. Requires 2 times resources required for each version.
- **Rolling upgrade.** For each server running version N, shut the server down, install version N+1, reboot. Requires minimal additional resources but means that different versions are simultaneously active. Industry standard practice. May lead to mixed version race condition.
  - One variant is to delay after first upgrade (canary) to ensure there are no problems.
  - Another variant is to stage upgrade. 100, 1000, 10000 servers, etc. More on this later

# Mixed Version Race Condition

1 Start rolling upgrade



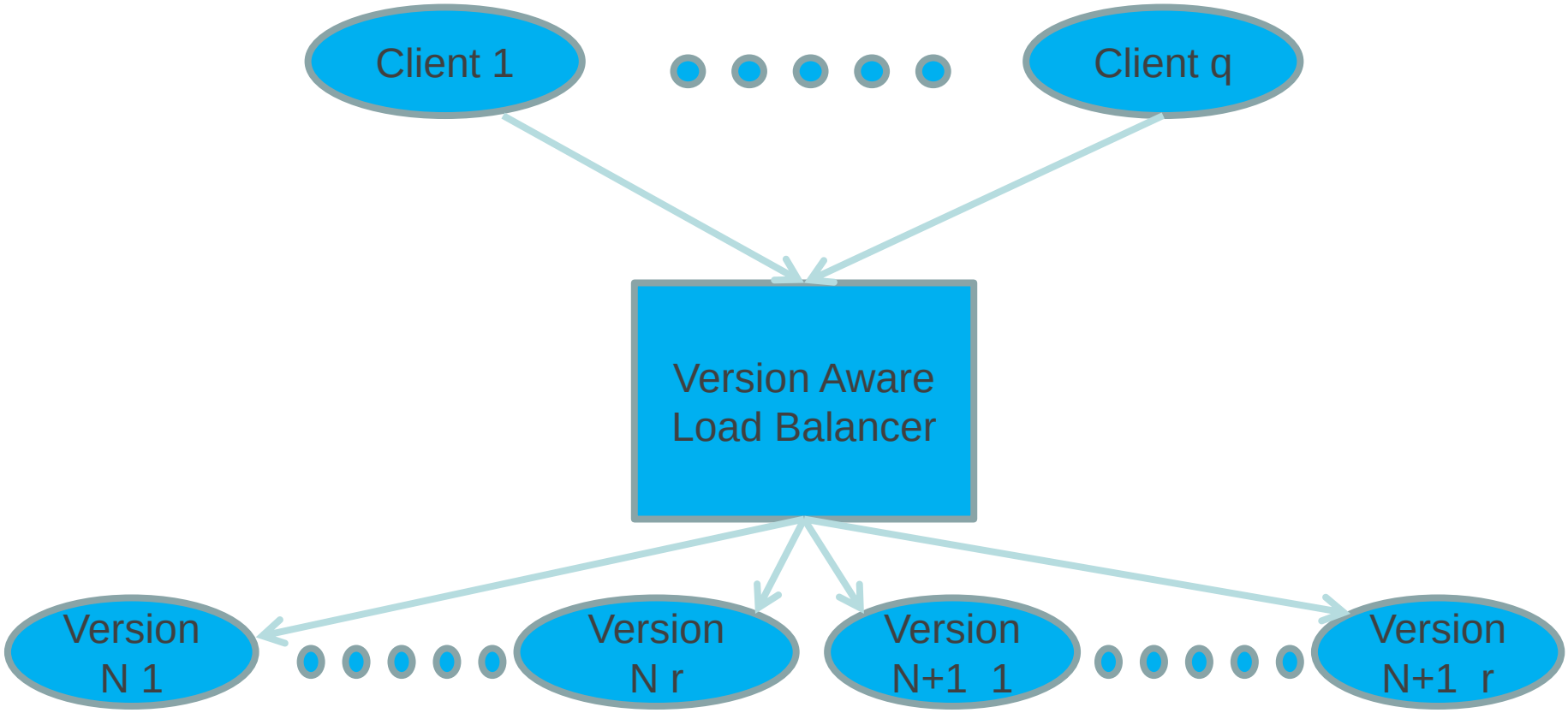
# Modelling Rolling Upgrade Using Colored Petri Net



# Preventing mixed version race condition


- Model verifies that making load balancer version aware will prevent mixed version race condition.
- Problems associated with making load balancer version aware
  - Difficulty in modifying load balancer
  - Performance impact on load balancer of necessary modifications
  - Making modified load balancer balance the load across requests and versions
  - Synchronizing solution across multiple independent clusters
- Student team has implemented a solution and solved the first two problems.

# Version Aware Load Balancer




Constraint: once a client request has been routed to a Version N+1 server, no subsequent requests from that client are routed to a Version N server

# Scheduling Version Aware Load Balancer

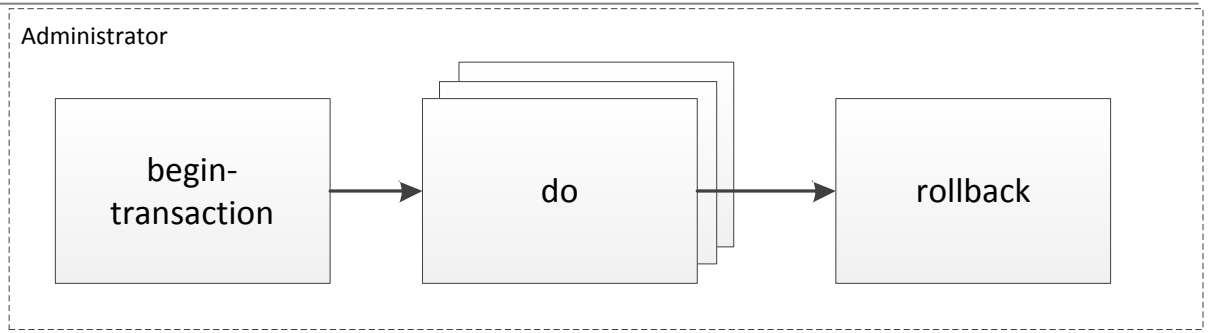
NICTA

- What is measure of goodness of load balancer scheduling algorithm?
  - Requests evenly distributed across all servers will not work in version aware context
    - Either some clients never see Version N+1 server or
    - Some servers never get requests
  - Keeping all servers within load constraints is complicated by version awareness and desire to complete upgrade quickly.
- Different variants for upgrade strategy will affect load balancer scheduling.
  - Canary strategy (small number of upgraded servers for a period) combined with version awareness means canaries must get sufficient number of requests to act as an indicator but not so many as to overwhelm them.

# Operator undo

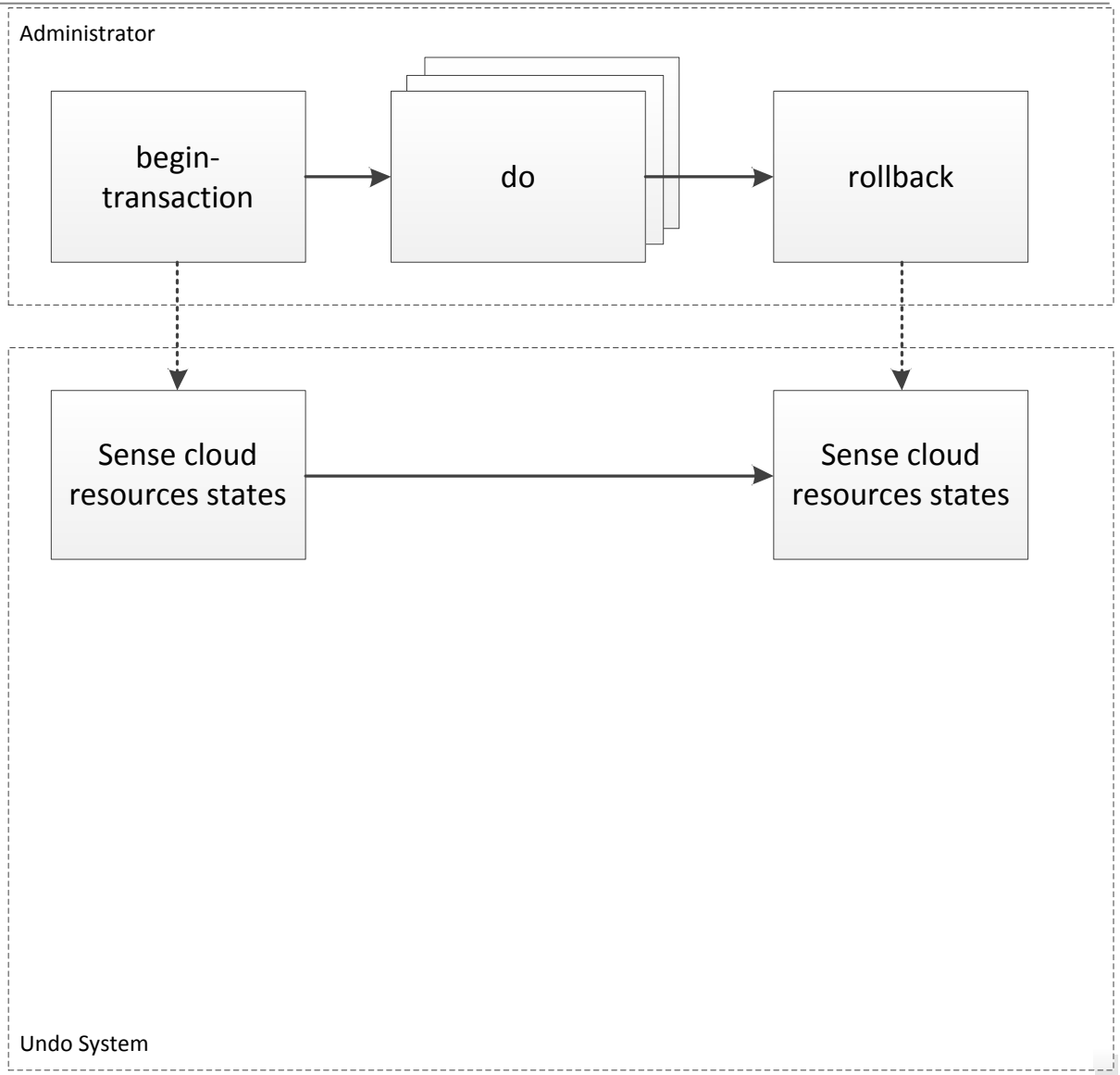
- After performing an operation in AWS, may want to go back to original state – i.e. Undo the operation
  - May be result of API failure
  - May be desire to set up testing environment
  - May be result of upgrade failure.
- Not always that straight-forward:
  - Attaching volume is no problem while the instance is running, detaching might be problematic
  - Creating / changing auto-scaling rules has effect on number of running instances
    - Cannot terminate additional instances, as the rule would create new ones!
  - Deleted / terminated / released resources are gone!

# Undo for System Operators

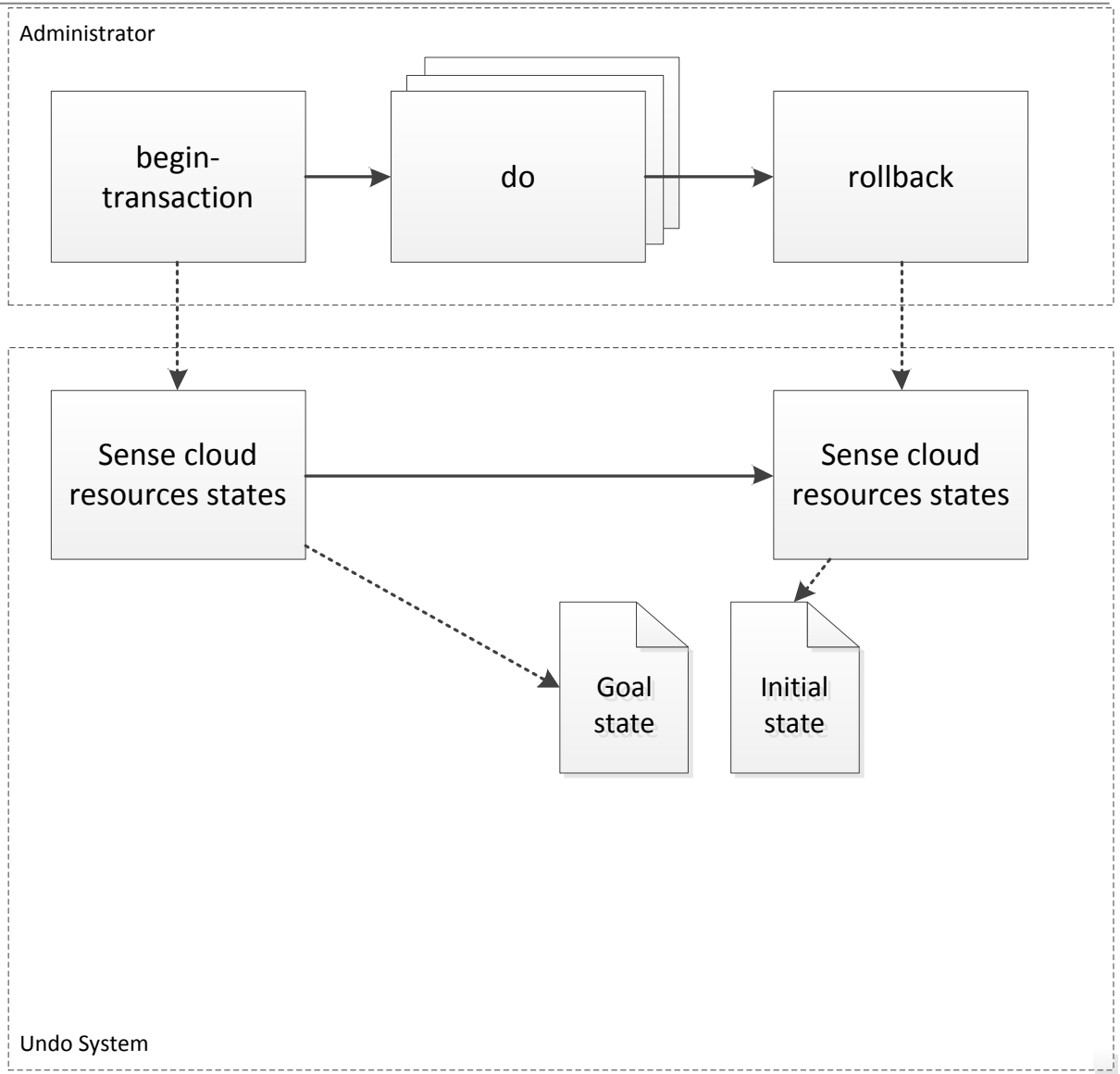



+ commit
+ pseudo-delete

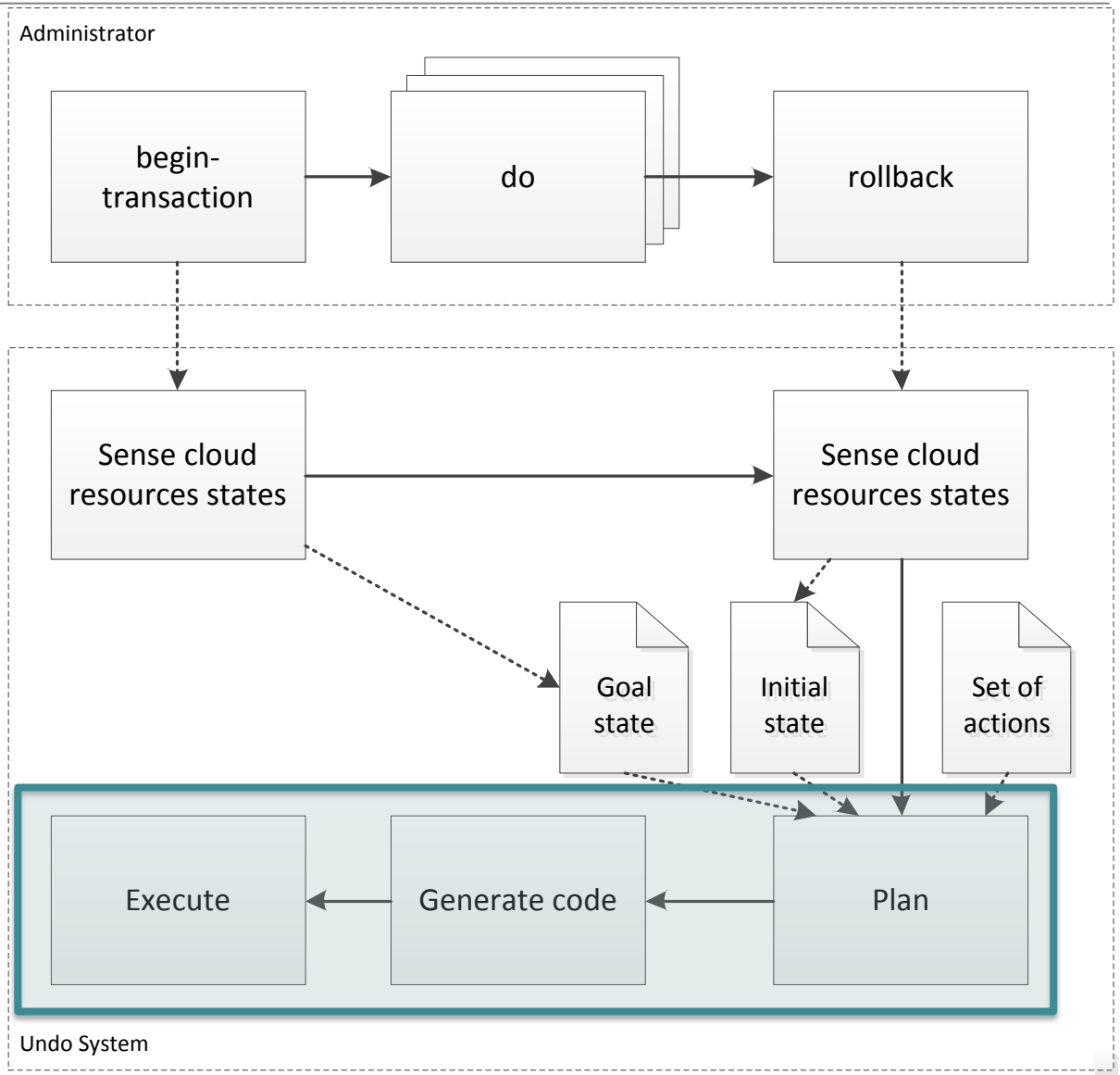
# Approach



# Approach



# Approach

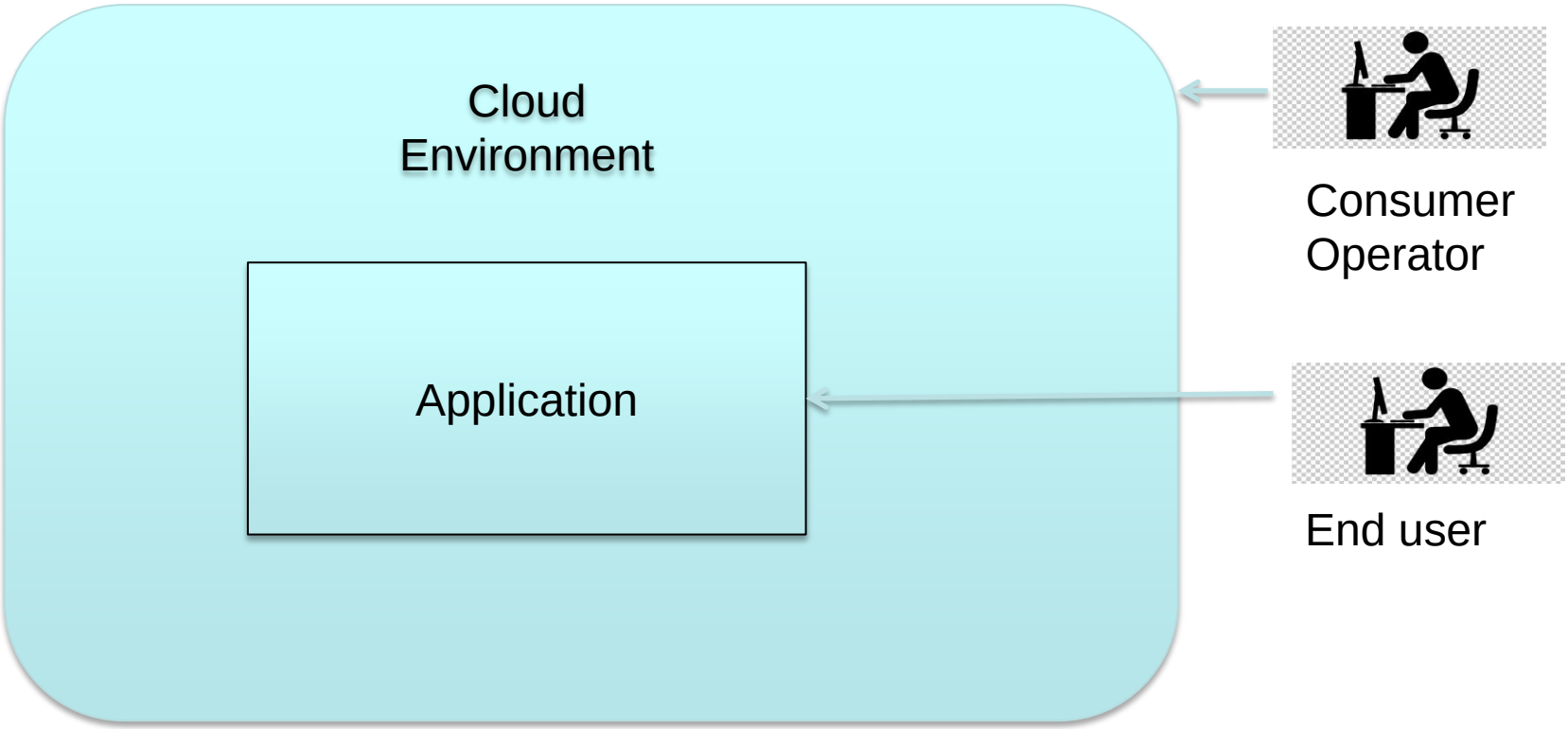


# Outline

- Overview of operations space
- Some results we have achieved
- **Operations through performance engineer glasses***
  - Technical challenges
  - Adoption challenges

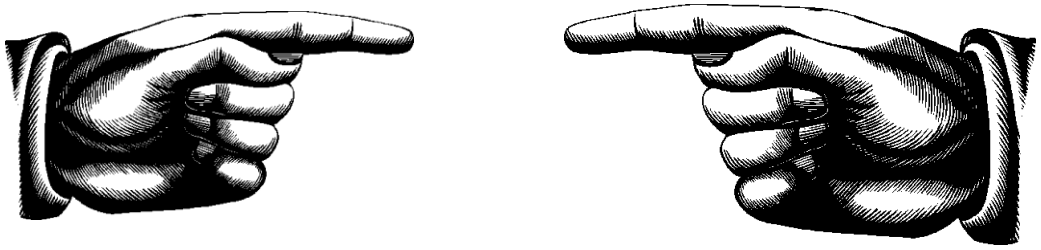
*apologies in case I am being too simplistic in this arena.

# An application and its environment



# Motivating Scenario

- You change the operating environment for an application
  - Configuration change
  - Version change
  - Hardware change
- Result is degraded performance
- When the software stack is deep with portions from different suppliers, the result is frequently:



# Technical Challenge 1

- Major internet company updates its underlying file system once a month.
- For each update
  - Week 1 – push update to 1000 servers
  - Week 2 – push update to 10,000 servers
  - Week 3 – push update to 100, 000 servers
  - Week 4 – push update to remaining servers
- Problems are reported by personal communication to individual in charge of file system.

# Why is personal communication necessary?

NICTA

- Some performance problems are only apparent when 10s of thousands of servers are active.
- 1% degradation in performance is within normal variance and not detected by existing models or tools.
- See “The Tail at Scale”, CACM, Feb 2013 for a more precise description of why this problem occurs.
- Challenge: construct model or tool that can give early warning of subtle performance problem.

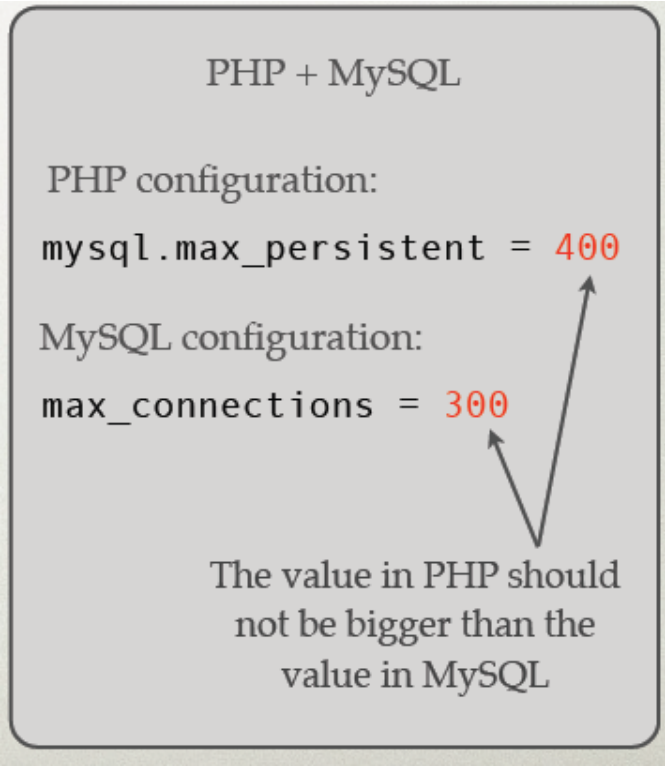
# Configuration errors*

- Configuration errors have large impact on system availability – e.g. a Facebook mis-configuration caused several hour outage
- Performance degradation resulted from 2-20% of configuration errors.
- These type of errors are particularly difficult to detect, especially when multiple systems are involved.


*Yin, et al. SOSP '11

# Technical Challenge 2

- Configuration parameters across multiple systems are particularly difficult to detect.
- How can performance engineering assist in detecting this type of error?



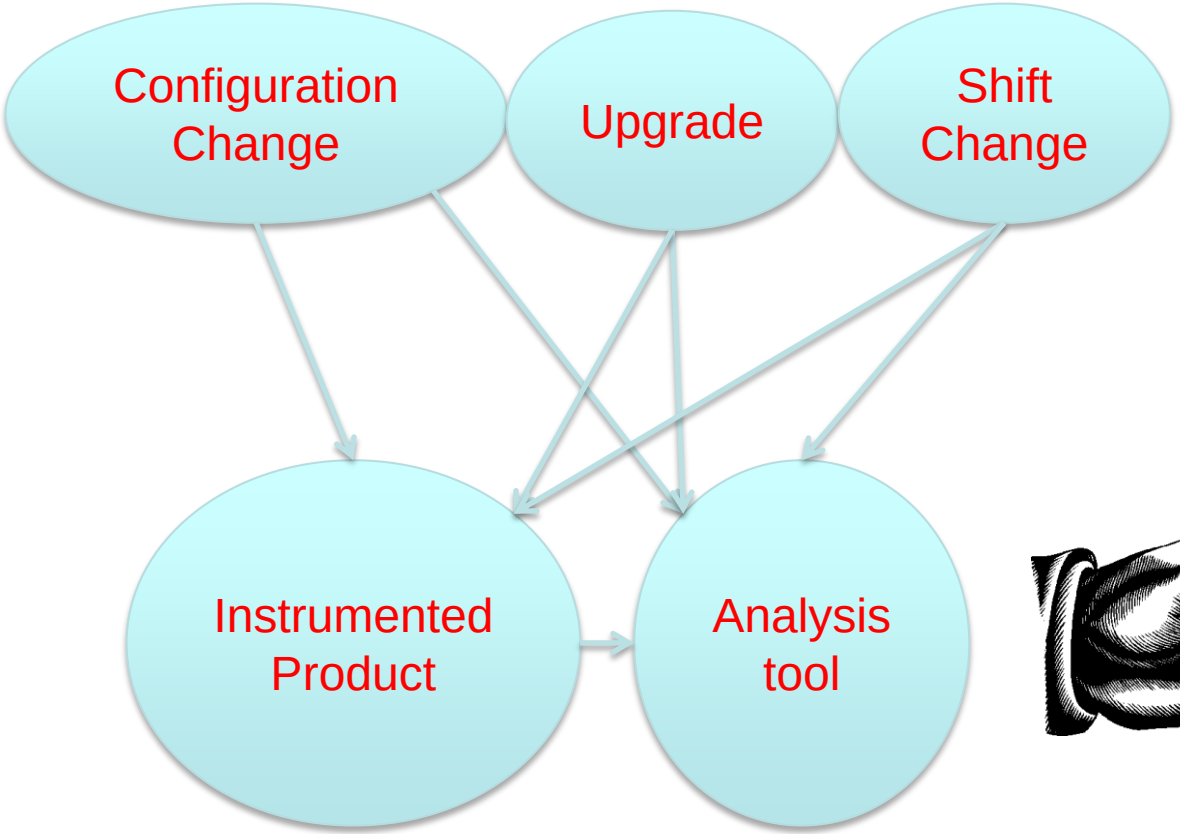
# Adoption Challenge 1

- For some systems, UDP is used instead of TCP for performance reasons.
- UDP is faster than TCP but does not provide guaranteed delivery.
- Number of packets lost is a function of buffer size.
- Trade off between performance, memory, and packet loss.
- The person making the choice may be ignorant of performance engineering. How can the results of performance engineering be packaged to enable its use.

# Adoption Challenge 2

- Database System mistakenly allocated to a slow disk/processor combination.
- How can performance engineering make this choice obvious quickly if the recipient is unsophisticated?

# Desired World




# General Question for Performance Engineers.

- How can you model effects produced by misconfiguration, user errors, or subtle effects?
- How can you make these models and their results available to people with no formal performance training?

# How do I get information to perform research?


- Every open source program requires a variety of configuration parameters.
- Every modern applications depends on a variety of middleware so cross stack examples should be readily available.
- Most organizations have extensive processes for their operations personnel. Use these processes as a framework investigating process/product interactions.

# Summary

- Operations problems will account for the majority of outages in the next several years.
- The operations space is a rich source of research problems that has been insufficiently mined.
- Many operations problems are caused by incorrect parameter setting. Performance problems are the most difficult to detect.
- Operations processes provide one setting for research.

# Questions?

Contact len.bass@nicta.com.au