SPA2007 WS1 – Strategies and Patterns for Systems Continuity

Eoin Woods
UBS Investment Bank
www.eoinwoods.info

Nick Rozanski
Marks and Spencer
www.nick.rozanski.com

Timetable

10:00 – 10:10	Introductions
10:10 – 10:30	Presentation 1: Setting the Scene
10:30 – 10:35	Exercise 1 Overview
10:35 – 10:55	Exercise 1: Identifying Threats & Risks
10:55 – 11:05	Break
11:05 – 11:15	Collate outputs of Exercise 1
11:15 – 11:35	Presentation 2: Achieving Availability
11:35 – 11:40	Exercise 2 Overview
11:40 – 12:00	Exercise 2: Applying Solutions
12:00 – 12:05	Break
12:05 – 12:20	Presentation of Exercise 2 outputs
12:20 – 12:30	Summary and Conclusions

Presentation 1 – Setting the Scene

Disaster Recovery is Important!

- a 48-hour outage would put 20% of Fortune 500 companies out of business
- only 43% of companies without a working disaster recovery plan would ever resume operations
- only 13% of companies would be in business two years later

University of Minnesota

- the average time to recover a business's systems after a disaster is 48 hours

Contingency Planning Research Inc.

Terminology

- system continuity
  - the ability of a system to protect some or all of its elements from certain types of failure and to recover those elements to operation after failures which they could not be protected against
  - Achieved via specialist solution technologies including fault tolerance, high availability & DR [more in Part II]
- IT service continuity
  - builds on and extends Systems Continuity to allow an entire IT service to continue with no / minimal interruption in the event of a localised or major failure disaster, irrespective of the underlying systems involved
- business continuity
  - extends IT Service Continuity to the ability of the business as a whole to continue operations (so includes people, processes, workplaces etc)

Assets Which Require Protection and Recovery

PEOPLE AND PROCESSES

APPLICATIONS

INFRASTRUCTURE

ENVIRONMENT

includes:

- users
- support and development
- management and operations

OUT OF SCOPE FOR TODAY

Artechra

Assets Which Require Protection and Recovery

APPLICATIONS
- software presentation layer
- systems and applications
- middleware (database, integration)
- application data

INFRASTRUCTURE
- software infrastructure (operating systems, security and identity, addressing)
- presentation devices (eg desktop, handhelds)
- processing capability (servers)
- persistent storage (disk and backup)
- network (local and wide-area communications)

ENVIRONMENT
- data centre
- electric power and cooling
- external connectivity
- physical access

impact

Scope of Failure

- component failure
  - failure of an individual component (disk, network card etc)
  - failure of an physical server, logical server or server partition
- service failure
  - failure of an application
  - failure of an application service (database, integration)
  - failure of an infrastructure service (eg DNS)
- site failure
  - complete or partial loss of a data centre
  - restricted access to data centre
- other types of failure
  - desktop or server distributed failure (eg virus infection)
  - application-level data corruption (eg bad data written to db)
- unexpected volumes / business growth
  - especially Internet expansion or new sales channels

Severity of Failure

- intermittent ("soft") failure
  - expected failures that occur routinely (if occasionally)
  - retrying an operation is the preferred recovery option
  - may escalate to a hard failure after retry is attempted
  - deadlocks, communication timeouts, log full conditions
- permanent ("hard") failure
  - unexpected failures which do not occur routinely
  - no change in behaviour is expected on a retry attempt
  - invoking a recovery procedure is the only recovery option
  - usually caused by total failure of a system component
- renewability
  - the idea that when a broken element is repaired, it is as reliable as before it failed (its MTBF is not reduced – see later)
  - components that are not renewable degrade over time

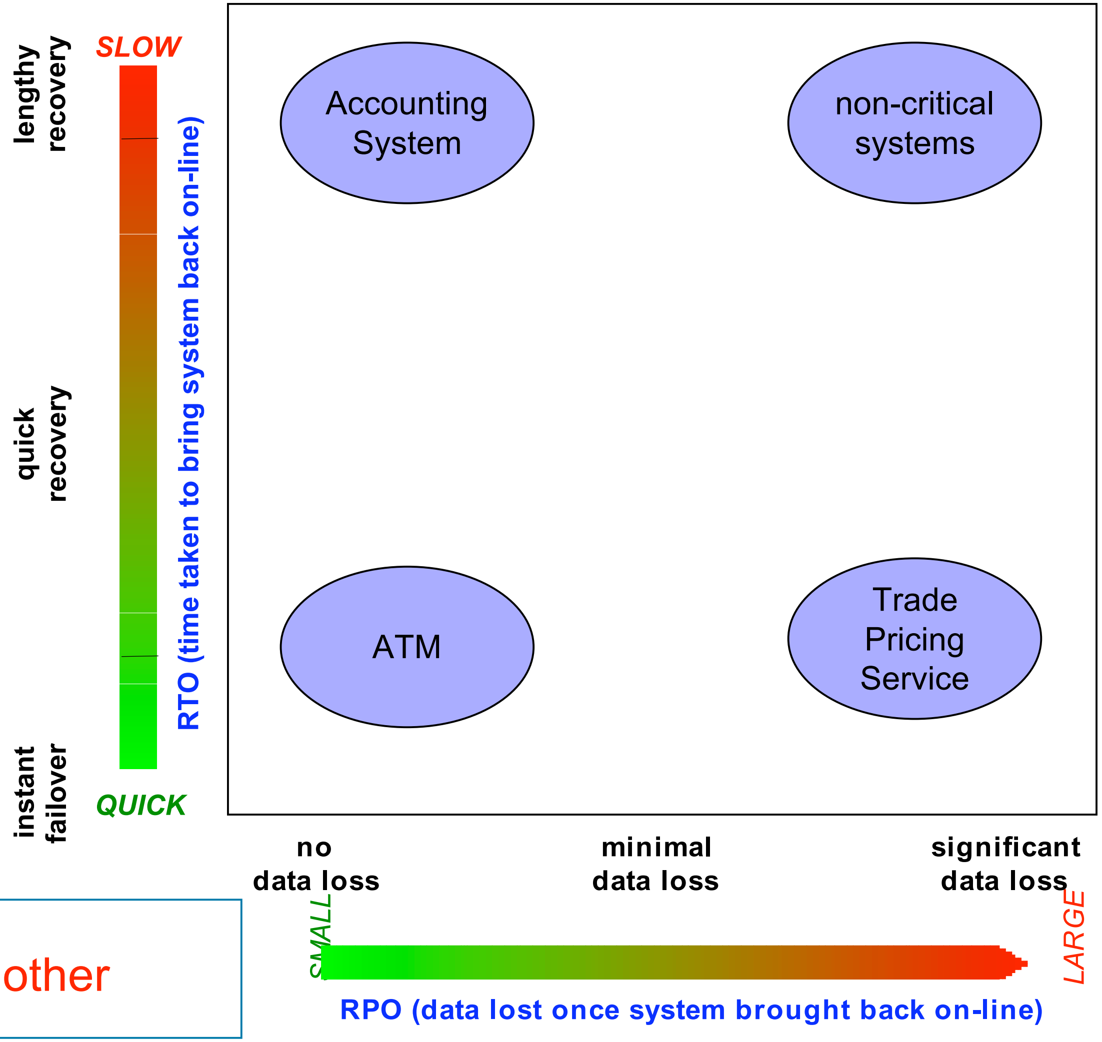
Standard Availability Metrics

- Mean Time Between Failure (MTBF)
  - aka Mean Time Before Fault
    - MTBF = elapsed time / number of failures
- Mean Time To Repair (MTTR)
  - average time taken to repair a fault once notified
- Availability
  - usually expressed as a percentage (“five nines” or 99.999% represents one hour downtime a year)
    - availability = time the system is available / elapsed time = MTBF / (MTBF + MTTR)

- these metrics can be applied at different levels of the stack
  - often linked in to SLAs (Service Level Agreements)
  - MTBF specifications are often available from hardware manufacturers (especially disk and mainframe manufacturers)
  - it is much harder to get objective metrics for the software stack (operating system or web server availability) or at the “service” level (eg availability of order processing service)

Standard Recovery Targets

- Recovery Time Objective (RTO)
  - the maximum amount of time it can take for a component or service to become fully operational following a failure
- Recovery Point Objective (RPO)
  - the maximum amount of data loss which is acceptable following recovery of a component



these two targets are largely independent of one another

Calculation of Risk

risk of failure = likelihood x duration x impact x cost of downtime

- likelihood of failure
  - how often the failure is expected to occur over the remaining life of the component / service / site
- duration of failure
  - the length of time that the failed component / service / site will be unavailable in the event of failure
- impact of failure
  - the percentage of users who are affected by the failure of the component
- cost of downtime
  - how much it costs per unit of time if the component / service / site is unavailable

- in this model risk is expressed as a monetary value

Blueprints for High Availability, Marcus and Stern (Wiley 2003)

Exercise 1 – Threats and Risks

- review our outline descriptions of large information systems (or use your own)
- identify the potential availability threats to the system and the risks to the systems that these threats imply
- identify some appropriate targets for availability metrics
- categorise the threats in order to allow broad themes to be identified

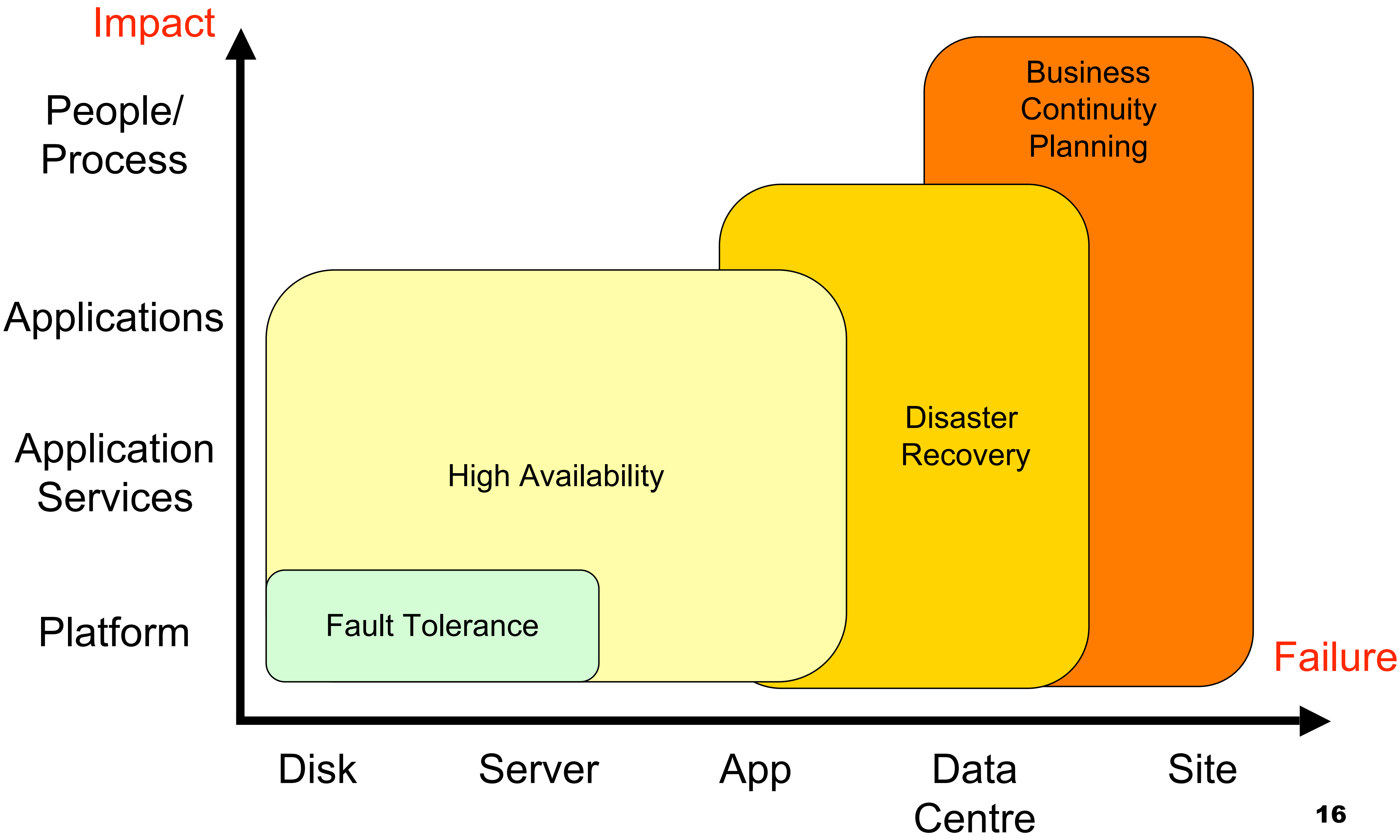
- overview: 5 minutes
- exercise: 20 minutes
- break: 10 minutes

Presentation 2 – Achieving Availability

Types of Solution

- fault tolerance
  - masking the failure of a single sub-component (disk, network card, …)
  - automatic recovery from the failure (in milliseconds to a few seconds)
  - no impact on end-users
- high availability
  - coping with the failure of a major component or entire application
  - (largely) automatic recovery from the failure (in seconds to a few minutes)
  - low impact on end-users (in-flight transactions may be lost)
- disaster recovery
  - coping with the failure of part of the IT environment (e.g. data centre)
  - manual recovery processes required (taking minutes to hours)
  - medium impact on end-users (last few transactions lost)
- business continuity planning
  - coping with loss of part of the organisation (e.g. head office)
  - manual recovery processes across the organisation (taking hours to days)
  - large impact on end-users (services unavailable, emergency processes)

Applicability of Solutions



General Problem: Single Points of Failure

- eliminate as far as possible all “single points of failure”
  - any individual component or instance which, if it fails, makes the service unavailable
- defence is usually some form of component replication
  - see next slide
- service availability must be decoupled from component availability
  - clients have to rely on the *service* not the components that provide it
  - applies to both hardware (e.g. disk storage) and software (e.g. web server)
  - the availability of a service is only as good as its weakest (least-available) component

the trade off is increased complexity - itself a threat to availability!

General Solution Principles

- replicate system components
  - the fundamental approach for all continuity solutions
  - if something breaks, have a spare one and switch to using it (“fail over”)
  - this approach is used from micro-component to environment level
  - the degree of automation of failover varies dramatically
- decouple users of a service from its implementation
  - logical naming services (DNS, service registry)
  - rely on interfaces not implementations (…)
- provide a mechanism for moving to replica components
  - manual process and restart
  - automated process with automated retry
  - mask failure entirely (“hot swap”)

also need a way to fail back once failed component is repaired!

Approaches to Replication

- component replication
  - use idle “standby” components
    - retain full capacity if failure occurs
    - wasteful and expensive if no failures occur
    - failover may be more complex (or at least less well tested)
  - routine workload distribution across all available hardware
    - uses load balancing techniques
    - more complex and expensive in normal operation
    - usually seamless in the event of failure
    - reduced capacity if failure occurs

- data replication
  - synchronous or asynchronous
    - performance vs. possible data loss trade off
  - persistent data (e.g. db tables)
  - transient data (e.g. messages in queues)
    - replicated themselves
    - used as a replication mechanism

Environment-Level Solution: Site Replication

- replicate the entire environment
  - replicate at the level of the data centre
  - Means no single point of failure (at least in the data centre!)
- active/passive site replication
  - leave the replicate site unused during normal operation
  - allows you to fail over all or part of the service if there is a disaster at the primary data centre
  - the simpler (and more flexible?) model
  - expensive and wasteful of capacity
- active/active site replication
  - a more cost-effective model is to run some workload at the secondary
  - could host development or test environments, or even production workload
  - more complex (and less flexible?) model
  - requires sophisticated (and expensive) load-balancing and data sharing / replication technologies

Component-Level Solutions

- standby systems and components
  - **hot standby**: minimal interruption of service or loss of data in the event of failure
  - **warm standby**: brief interruption of service (and possible data loss) in the event of failure
  - **cold standby**: extended interruption of service (and probable data loss) in the event of failure
- workload distribution
  - use all of the hardware for production operation
  - accept degraded service in the case of failure
  - symmetric distribution (load balancing of entire workload)
  - asymmetric distribution (e.g. MIS reporting uses the DR database)
- clusters
  - HA clusters
  - scalable clusters
- virtualisation
  - large emerging area - see next slide

Component Virtualisation

- server virtualisation
  - mainframe / Unix logical partitioning (e.g. z/OS, pSeries LPARs)
  - software machine virtualisation (e.g. VMWare, XenSource, Virtual Server)
- network virtualisation
  - VLANS etc
  - virtual network addressing (e.g. for HA clusters)
- network-based storage virtualisation
  - storage area network hides the physical characteristics of the underlying disk
  - takes the NAS / SAN model to the next level of abstraction
  - emerging technology
- abstraction techniques in software design
  - standard application design patterns (eg logical names for servers)
  - needs to be rigorously enforced

virtualisation techniques provide mobility and flexibility which are necessary (but not sufficient) for high availability and disaster recovery

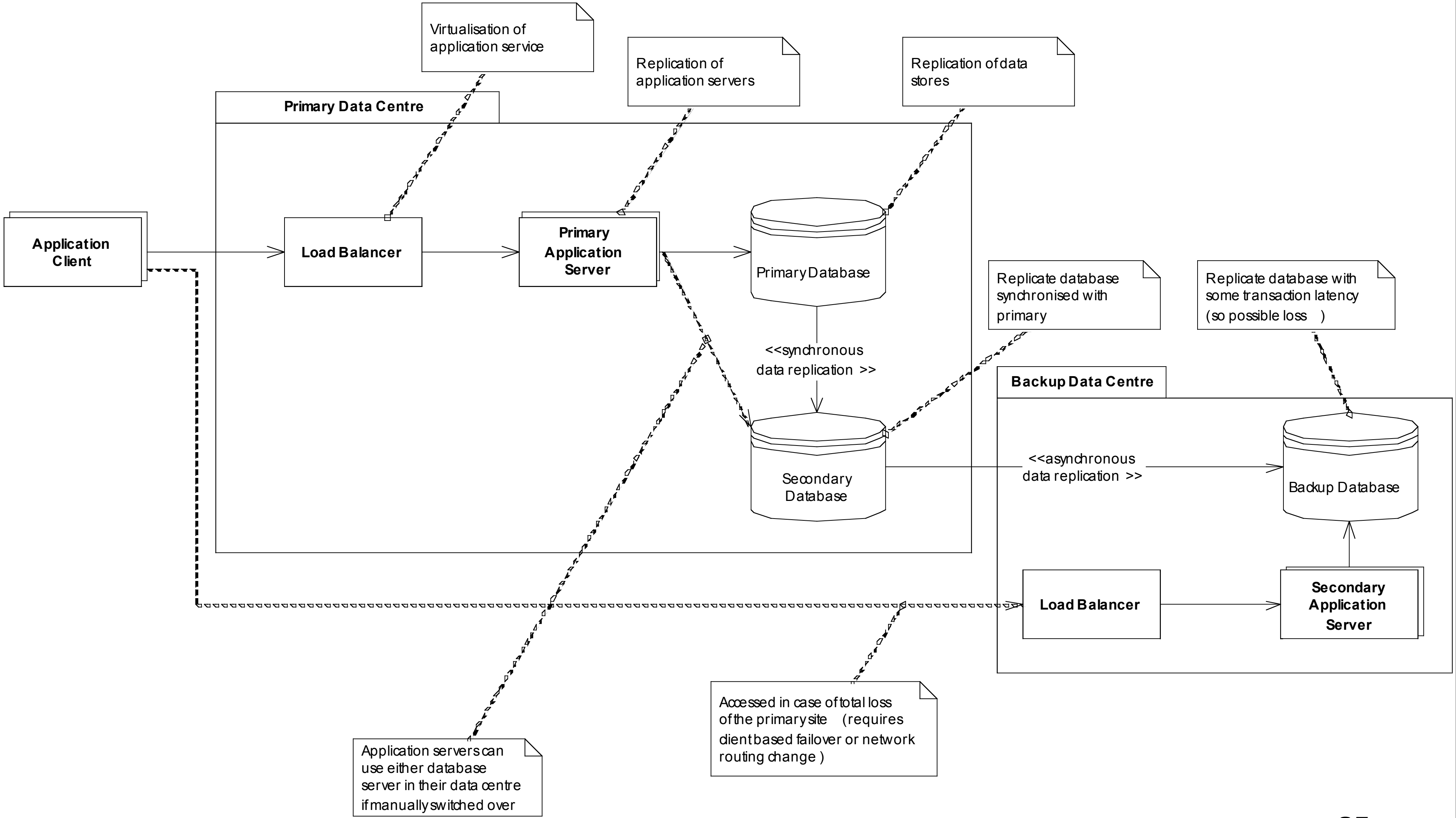
Data Solutions

- fault tolerant storage
  - RAID, SAN, NAS
  - data centre wide transparent protection
- synchronous replication
  - LAN to metro-area replication distance
  - slows down data storage performance as distance increases
  - application transparency and transactional integrity
  - no unexpected loss on failure
- asynchronous replication
  - geographically distant replication possible (e.g. London → Zurich)
  - minimal impact on performance
  - application transparency but may or may not have transactional integrity especially across distributed data stores
  - unpredictable data loss on failure (depending on replication latency)
- application resilience to data loss
  - restore from backups and logs / Replay messages / Re-key transactions
  - additional complexity to application and recovery processes

Environmental Factors

- standardisation and simplicity
  - less chance of confusion, more flexibility in recovery strategy
- remote administration
  - allow recovery and operation even when sites are unavailable
- automated monitoring
  - ensure that failures are not overlooked
  - but need manual confirmation and override for critical operations
- automation of routine operations
  - ensure that environment is as expected when failures occur
- testing of replicates and procedures
  - routinely failover and run the business from the replicates
- accessible documentation
  - make sure information for recovery is online, accurate and minimal
- learning from incidents
  - log incidents, recovery performed and success (or otherwise) of procedure

Example of Solutions in Action



Financial Benefits of Continuity Solutions

- recap: risk
  - risk = likelihood x duration x impact x cost of downtime
- compare risks before and after solution implementation
  - savings = risk before implementation – risk after implementation
- use this to estimate return on investment (ROI)
  - ROI = savings / cost of implementation
- warning 1: there are many subjective judgements in these calculations!
- warning 2: there are other costs of downtime than financial ones!
  - reputational loss or brand damage
  - share price impact
  - loss of customers / future business
  - legal liability / regulatory penalty

formulae from *Blueprints for High Availability*

Exercise 2 – Design Solutions

- review our typical failure or disaster scenarios your systems could face (or use your own)
- identify combinations of solutions needed to survive these situations
- assess their ability to meet their HA/DR targets
- identify any potential problems in deploying these solutions or invoking them in the face of a real disaster

- overview 5 minutes
- exercise 20 minutes
- break 10 minutes (w/presentation preparation)
- presentations 15 minutes

Summary and Conclusions

- levels of continuity
  - system – service – business
- assets that need to be protected
  - people and processes
  - applications
  - infra-structure
  - environment
- failures
  - by scope (component, service, site)
  - by severity (soft vs. hard)
- recovery metrics
  - RTO (outage time)
  - RPO (extent of data loss)
- risk calculation
  - likelihood x duration x impact x cost of downtime

Summary and Conclusions (ii)

- levels of solution
  - fault tolerance vs. HA vs. DR vs. business continuity
- general solution approaches
  - replication of system elements
  - decouple service users from service implementation
  - provide failover and failback mechanisms
- specific technology solutions
  - environment level replication (active/active vs. active/passive)
  - component solutions (standby, workload distribution, clusters, virtualisation)
  - data solutions (fault tolerance, sync/async replication, resilience to loss)
  - virtualisation (server, network, storage)
- environmental factors
  - standardisation, processes, testing, automation, …

Artechra

Comments and Questions?

Eoin Woods
eoin@eoinwoods.info

Nick Rozanski
nick@rozanski.com