IT Architecture Module

Qualities & Constraints in IT Architecture

Part II –

- Security
- Usability & Accessibility
- Maintainability & Flexibility

Agenda (Part I - previously)

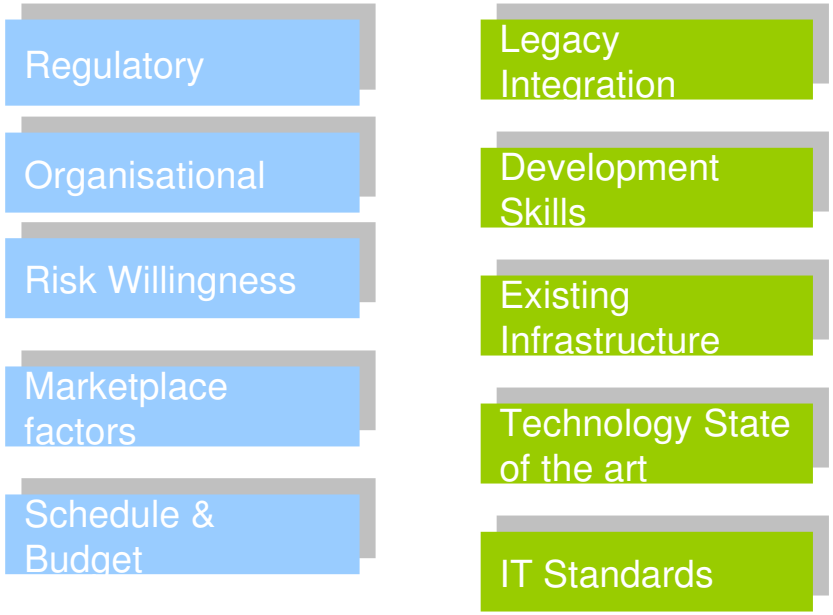
- *Qualities & Constraints in IT Architecture – overview*
  - What are “qualities and constraints” in IT Architecture?
  - Non-Functional Requirements and their quality
- Focus on *Availability*
  - Availability modelling
  - Availability design techniques
- Focus on *Performance*
  - The Performance Engineering Lifecycle
  - Volumetrics
  - Estimation and Modelling
  - Optional exercise

Agenda (Part II – this lecture)

- Focus on *Security*
- Focus on *Usability & Accessibility*
- Focus on *Maintainability & Flexibility*

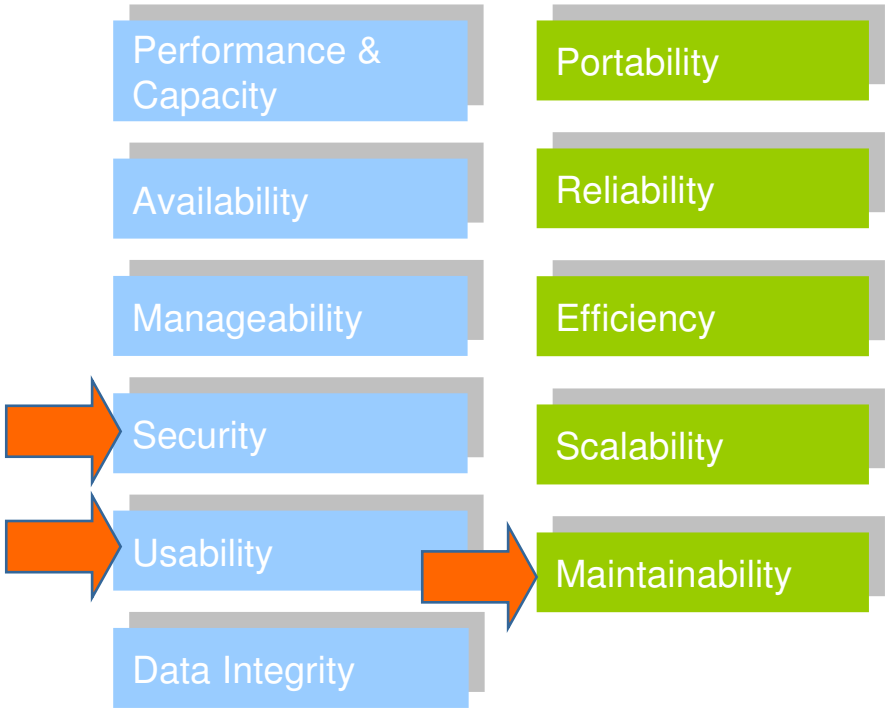
(Reprise) Constraints

- **The business aspects of the project, customer's business environment or IT organization that influence the architecture**
- **The technical environment and prevailing standards that the system, and the project, need to operate within**



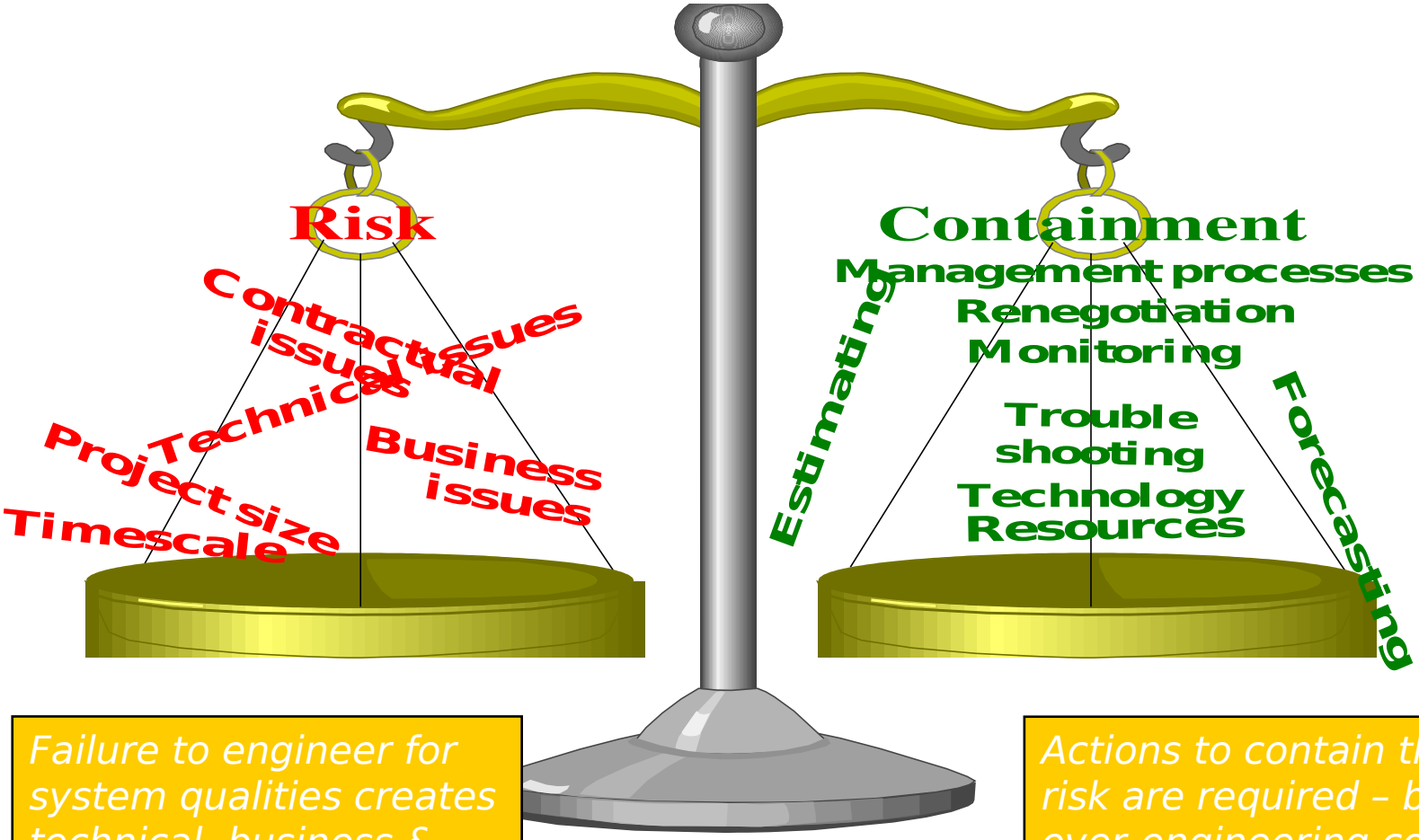
(Reprise) Qualities

- **Runtime qualities are 'measurable' properties, often expressed as "Service Level Requirements".**
- **Qualities might also be related to the development, maintenance, or operational concerns that are not expressed at runtime.**



focus of this session

(Reprise) Beware: a BALANCE must be maintained between *risk* and *cost*



Failure to engineer for system qualities creates technical, business & commercial risks

Actions to contain the risk are required – but over-engineering could be unnecessarily costly

Security in IT Architecture

Defining Security

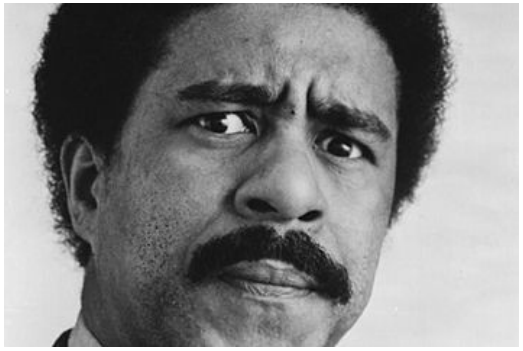
- Security is a wide and fascinating topic encompassing a vast range of issues, arenas and disciplines
  - from deep mathematics to international espionage
- In IT systems, “security” can be associated with the following qualities:
  - Not open to intentional misuse
  - Not open to accidental misuse
  - Protects the truth – maintains integrity
  - Protects service in the face of attack (overlap with Availability)
- Secure means SAFE:
  - Your data, your assets, your reputation

(Amusing?) Examples of insecure systems

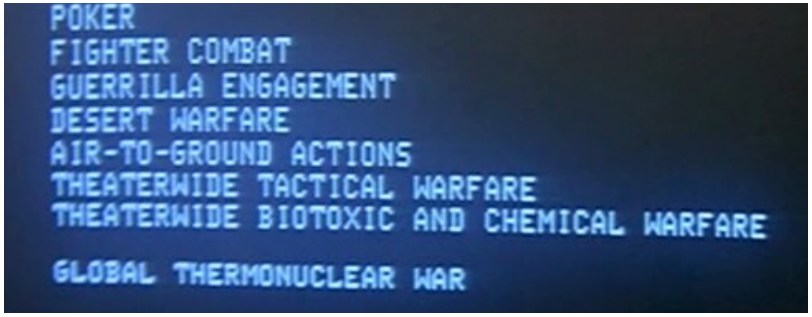
- Superman III – Richard Pryor's character bypasses access controls by typing:

```
> override all security
```

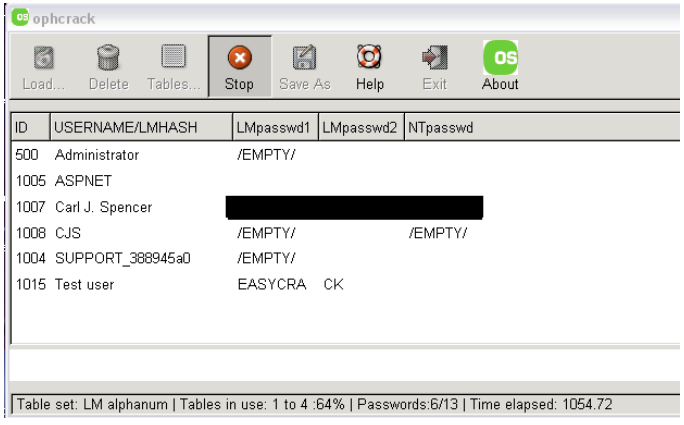
.. into the console



- In the film "War Games", Matthew Broderick gains access to the WOPR computer using a password "backdoor"



- Tools freely available to "hack" your Windows passwords (e.g. OPHCRACK)



Security is a critical concern in IT Architecture

- Wherever systems are responsible for important data and processing, there is a risk that misuse of the system leads to a negative outcome for those associated in any way with that system
  - Typically in a commercial setting, IT Architects need to think about protecting our customers (e.g. a bank)
  - … and *their* customers (e.g. an account holder)
  - (… and both our reputations!)
- The scale of the risk depends on the nature of the organisation(s) and the nature of the purpose of the system …

Scale of Security Risk – from war to web browsing

Arena	Sample applications	Example risks
Military systems	Identify Friend or Foe (e.g. aircraft) Nuclear command and control	Prevent identification, present false identity (lose battle => lose war) Unauthorised use of nuclear weapon (e.g. in unstable state)
High value financial systems	Payment instruction exchange (e.g. SWIFT), foreign exchange, stock trading	Money siphoning; value alteration Lax controls (e.g. Barings back – Nick Leeson)
Retail banking	ATMs, Online banking	Expose private data Fraud – e.g. false transactions initiated (loss of money)
Home computing	Email, word processing, web browsing, picture management	Virus attack – data corruption, loss of data, … Privacy invaded (files accessed)

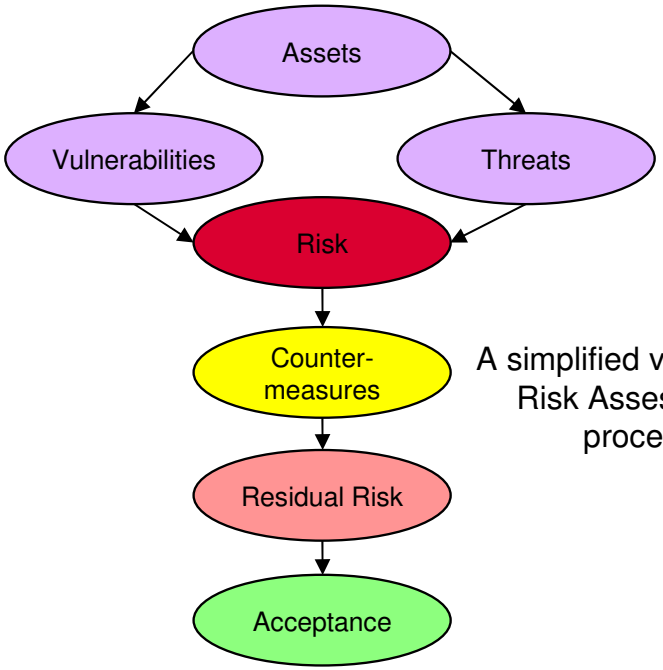
Impact to businesses

- Fraud and theft of data and other assets
  - Bottom line losses, e.g. 2006 CSI/FBI Computer Crime and Security Survey
    - Survey of 313 businesses of various sizes in the US
    - Average loss per respondent: $167,713
- Loss of Reputation and trust
  - Will customers trust companies that can't look after their data?
- Disruption to operations
  - This is not about creating new value
- Cost of enforcing security – ref. balancing scales
  - From the same survey: combined average annual security expenditure per employee: $1,349 for businesses with revenues < $10m

A good general approach to tackling IT security is to take a 'threat-based' approach

- **Document assets**
  - Identify and decide what you need to protect. This could be data, intellectual capital, processes, physical resources, or any other thing of value in the organisation
- **Understand threats**
  - Know your enemy. Determine from whom or what are you protecting your system and/or network
- **Define policy**
  - Create a comprehensive security policy and implementation plan which is appropriate to the level of threat
- **Implement policies**
  - Apply the security policies to your organisation and systems
  - Update or include security elements and configurations in IT solutions
- **Monitor policy**
  - Continually monitor to detect any deviation from your policies and take actions if needed

Threat assessment needs to be combined with assessment of vulnerabilities to determine risk



A simplified view of the Risk Assessment process

- Information security **risk** can be viewed as the **cost** to an organisation of **compromise or damage** to an information asset
- There are many ways to assess risk, some formal and **quantitative**, some informal and **qualitative**.
- In all cases, the purpose is to identify **significant threats** and address them through appropriate **countermeasures**

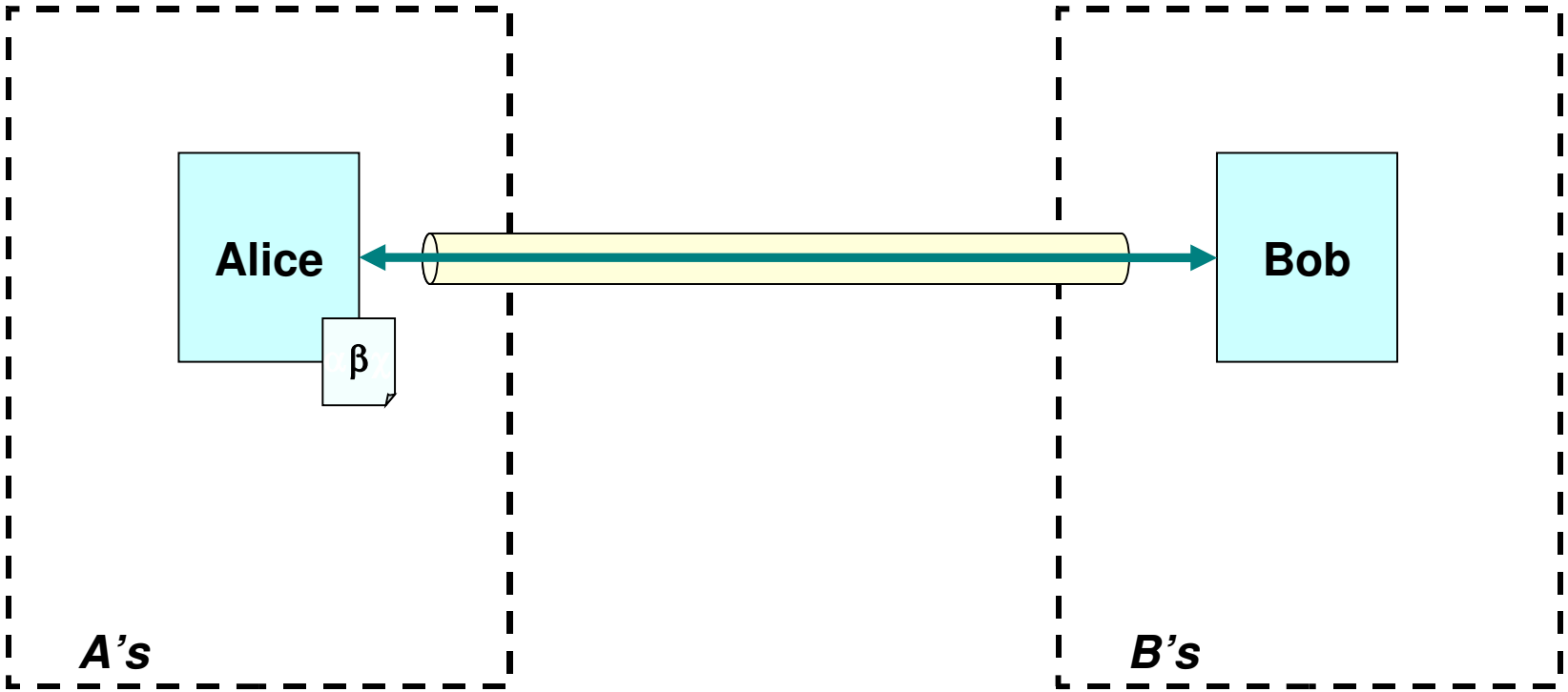
- In general, to assess risk it is necessary to know:
  - **Threats** – the bad things that might happen to an information asset
  - **Vulnerabilities** – the ways those bad things might come to pass
  - **Likelihood** – the probability of a vulnerability being exploited to make a bad thing happen
  - The "**value**" or "**sensitivity**" of the asset – the impact on the organisation if a bad thing happened

A few examples of sensitive assets

- Data
  - Customer accounts
  - Financial information or other critical MI
  - Intellectual Capital
- Processes
  - Financial processes – e.g. ones with purchasing power
  - Command and control processes
  - Other privileged processes
- Physical / infrastructure
  - Equipment
  - Hardcopy data
  - Bandwidth
- Intangible
  - Reputation

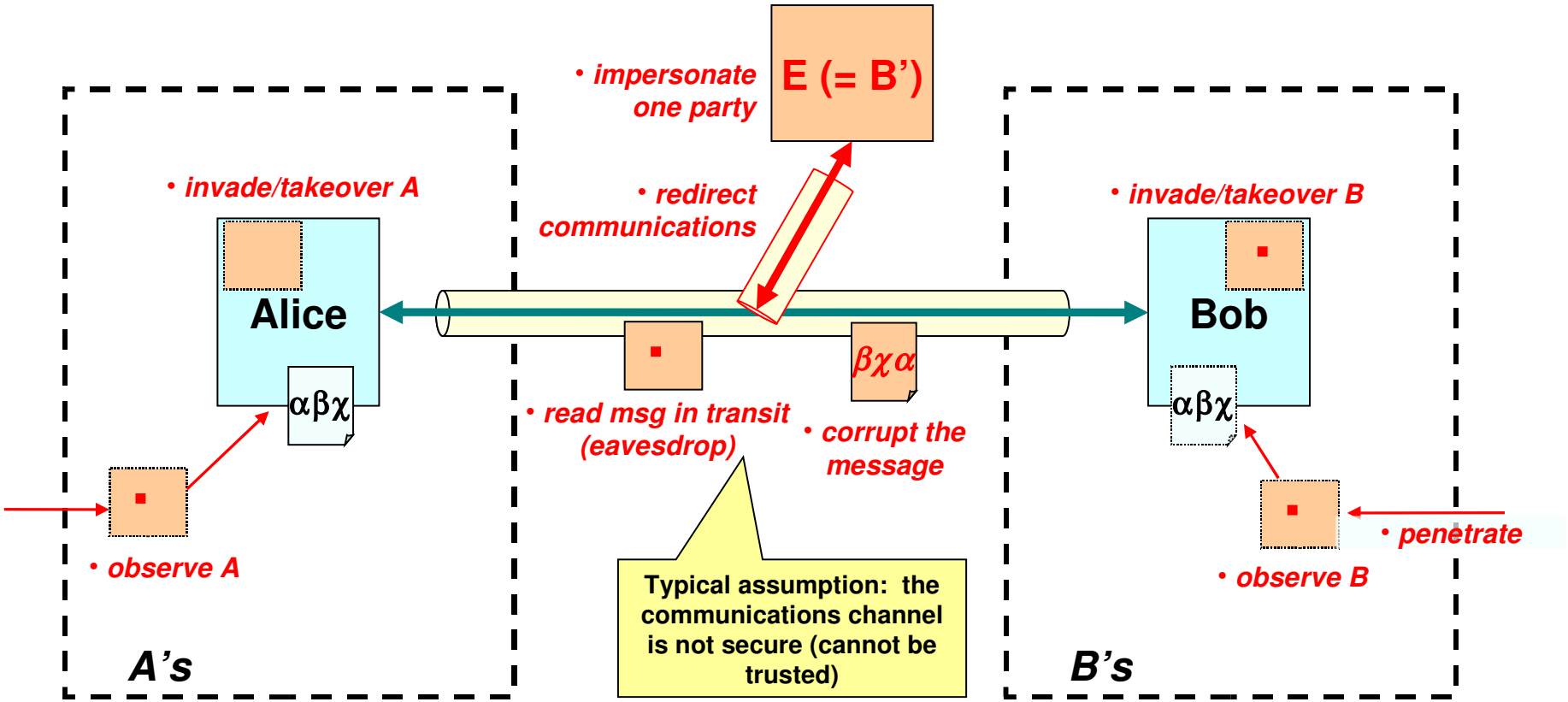
Security : Fundamental Concepts

Consider: Alice wants to send a message to Bob (securely)



Exercise: In what ways can we "attack" the communications between A and B?

Consider: Alice wants to send a message to Bob (securely)



=> Threats arise at both ends and everywhere in between

Threats - Where do threats arise from in IT System? And what can they do to us?

- Malicious
  - third party motivated to make money or other gain
  - competitor or parties acting on behalf of a competitor
  - hacker seeking “kudos”
  - employee seeking personal gain or to inflict damage on the corporation
- Unwitting
  - damage to assets through accidental action (insufficient safeguards)
  - accidental sharing of confidential information
  - program / system errors causing corruption or violating rules
- Combinations

- What can they do to us?
  - Observe, capture and forward confidential data
  - Alter data (to alter outcomes)
    - includes reputation damage, e.g. web site defacement
  - Delete data
  - Initiate unauthorised processing
  - Prevent (or disrupt) authorised processing
  - Deny access / service
  - Reduce system security
    - to ease other attacks
  - Steal assets (physical or otherwise)
  - …

Other attack types and terms

- DoS (Denial of service)
  - An attack on a computer system or network that causes a loss of service to users, typically the loss of network connectivity and services by consuming the bandwidth of the victim network or overloading the computational resources of the victim system
- ‘Malware’
  - A generic term given to malicious code. Can include spyware, adware, viruses, worms and other scams
  - Made particularly common by the Internet and the widespread use of the Windows operating system

Beginning the fight back: IT security relies ultimately on the products of cryptography (the science of designing ciphers)

- In order to protect the communications between A and B, we can encrypt the content of messages in transit
- A system of establishing and sharing keys (which are combined with the source message at time of sending) is required
- $\{Plaintext\}_{Key}$ => Ciphertext
- There are many different forms of encryption with varying properties and levels of protection
- The most commonly used algorithms in commercial systems are “Block ciphers”, which come in two flavours:
  - Symmetric key – same key for encryption and decryption
    - e.g. the Data Encryption Standard (DES)
  - Asymmetric (“public”) key – different keys for encryption and decryption
    - e.g. RSA, used in Secure Sockets Layer (SSL) on the web
- Key management itself is obviously critical and a significant challenge
- Cryptographic principles are used to build protocols which allow us to achieve objectives such as authentication

Key objectives of Security Engineering (1/2)

- **Authentication** – knowing who
  - The process of determining who users (human or otherwise) are and that they are who they claim to be. The most common technique for authenticating is by user ID and password. Others include certificate-based methods or biometrics
- **Authorisation** – knowing what can they do
  - The process of establishing the 'rights' that a user has to access and to perform actions on resources. (Simple example – the permissions to read and/or write a file)
- **Confidentiality** – protecting confidential data
  - Ensuring that data classed as confidential is only seen by appropriately authorised parties
  - Often achieved through cryptography – i.e. encrypting data

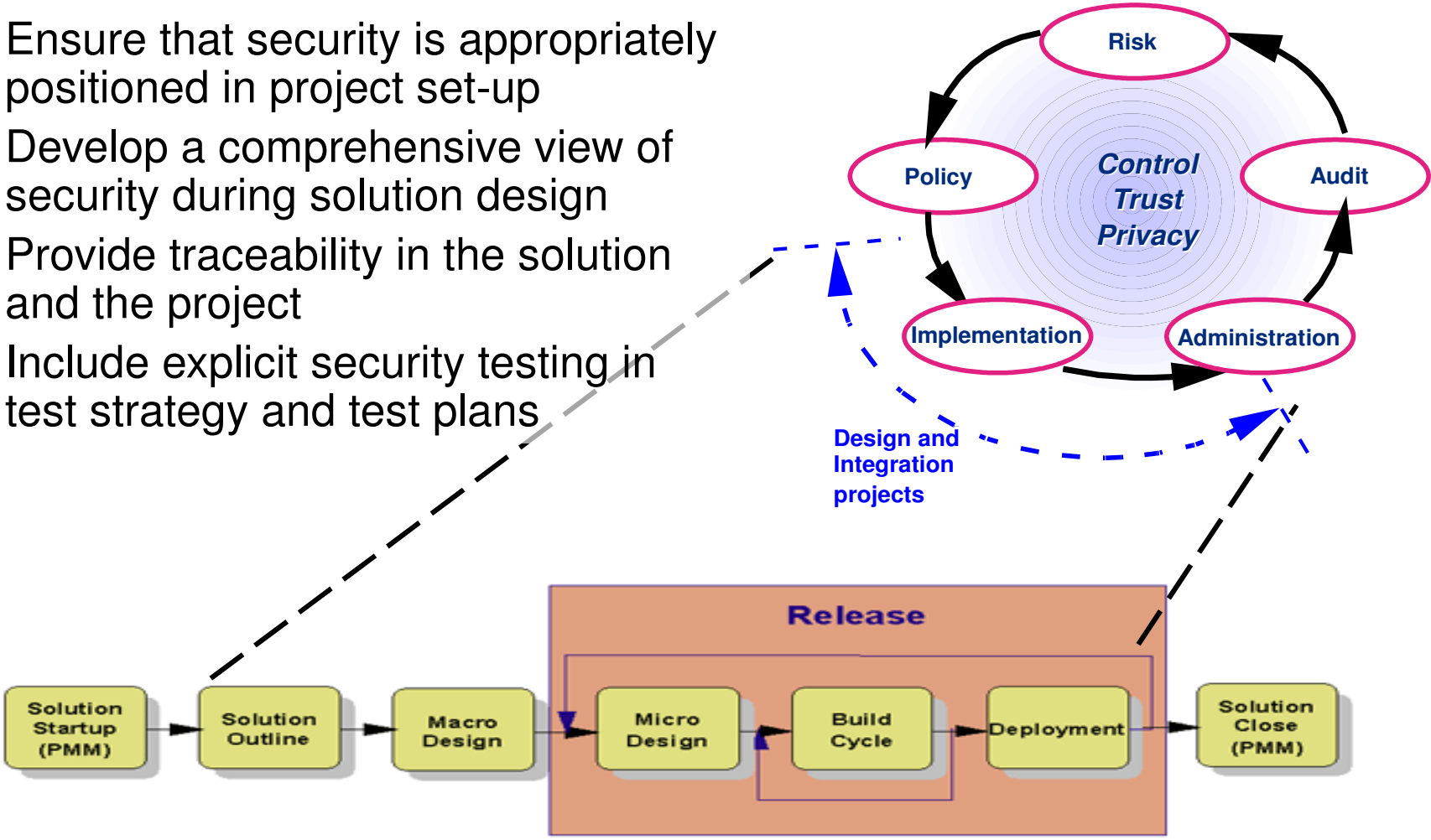
Key objectives of Security Engineering (2/2)

- **Integrity** – protecting the "truth"
  - The quality of a system whereby data and processing *always* conforms to the specified rules and constraints within the system
- **Auditable** – what did they do?
  - The trail of evidence proving the activities that have been performed on an internal asset – and attributing this to a known identity. This must be stored in a non-repudiable (tamper proof) format.
- **Non-Repudiation** – proving what happened happened
  - The ability to prove without contradiction that a transaction or event which is recorded as having taking place did take place
  - May need to be able to prove events in a court of law

Security :
Method and the Security Architect Role

The system design method should contain a risk-related approach to security

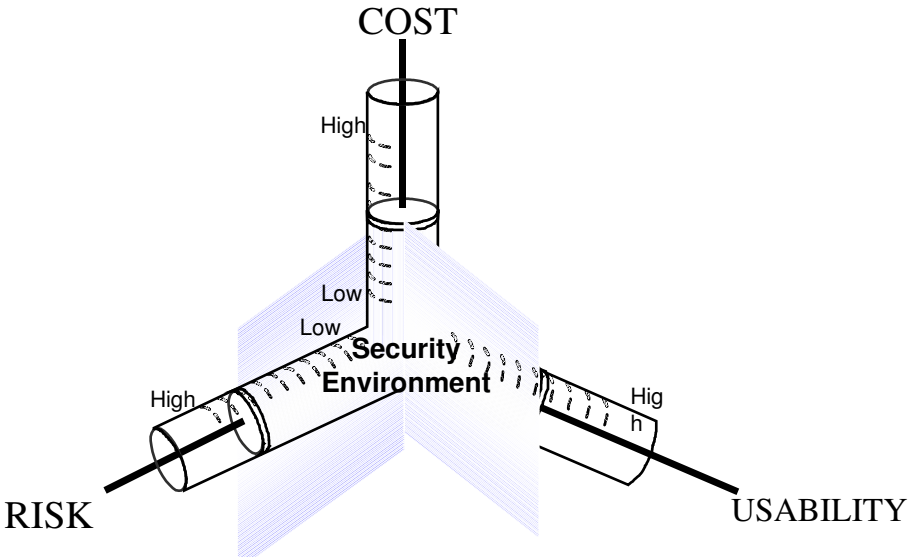
- Ensure that security is appropriately positioned in project set-up
- Develop a comprehensive view of security during solution design
- Provide traceability in the solution and the project
- Include explicit security testing in test strategy and test plans



At the solution outline phase, security architecture is about answering the question "how much security is enough (but not too much) security"

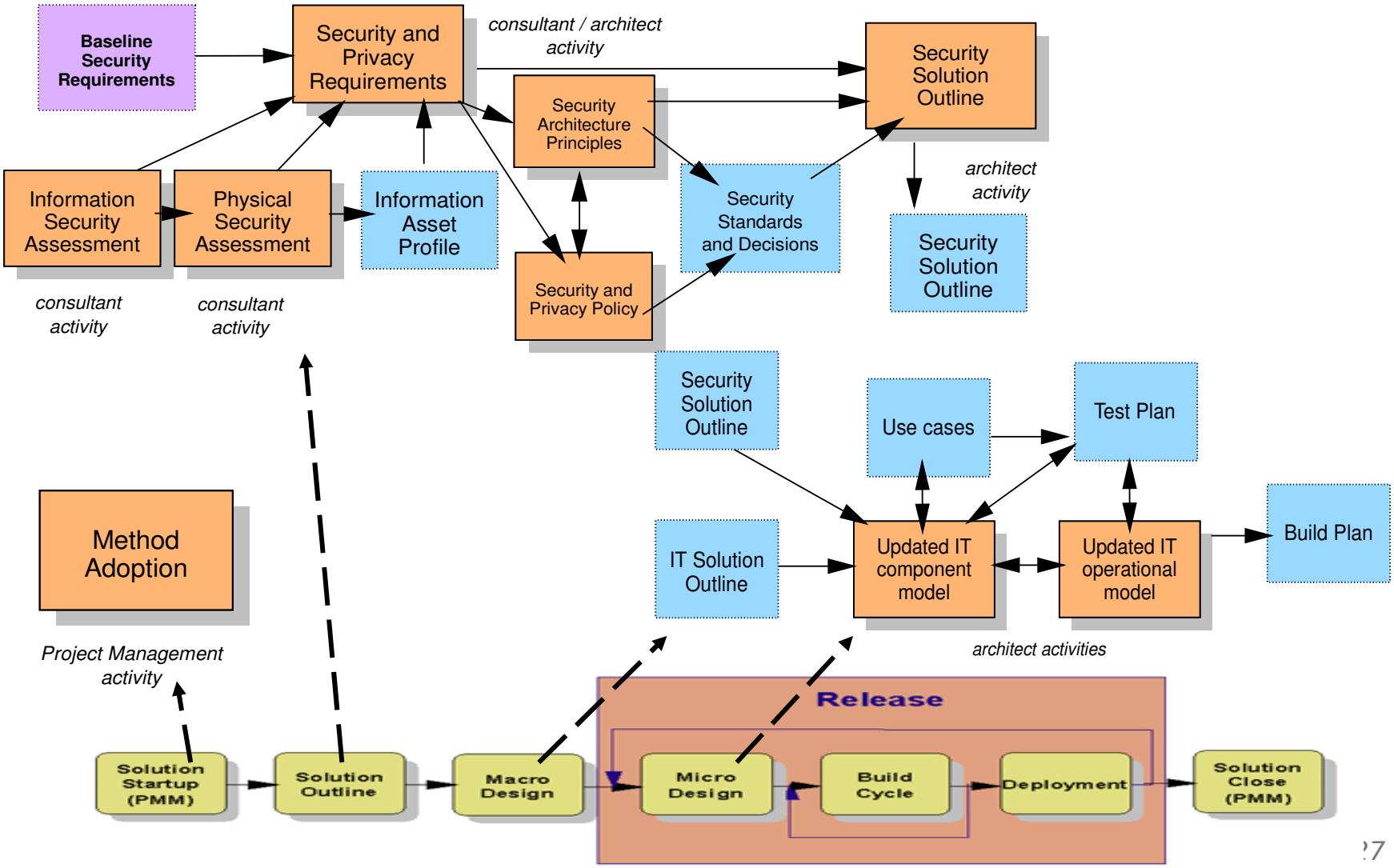
From a security perspective, all IT solutions must balance three conflicting factors:

- **The risk** – to the organisation
  - of operating the IT solution
- **The cost** – of implementing *and operating* the security controls



- The resulting set of controls must be, as far as possible "**necessary** and **sufficient**".

Early efforts focus on the security requirements and relationship to business processes



The "soup to nuts" view of a proactive security architect's role: addresses security issues at all phases in the lifecycle, across all the domains of the solution

Domain	Phase: Solution	Phase: Macro Des	Phase: Micro Des.	Phase: Build	Phase: Deploy
Bus	Bus Env Asset Profile Risk Assess	Authorisation & Access Control Security Bus rules			
Arch	Client IT Env Threat Analysis Security NFRs	Comp/Op Arch Security Test Strat Workstream Security	Authorised Dataflows		
App	Security Use Case Model	Security Use Cases	Security Dev Standards	Security Testing Application Ethical Hacking	
Ops		Security Process & Delivery Orgs	Dev/Test Security Define Security Baselines	Security Procedure development & implementation Implement Security Baselines	Infrastructure Ethical Hacking

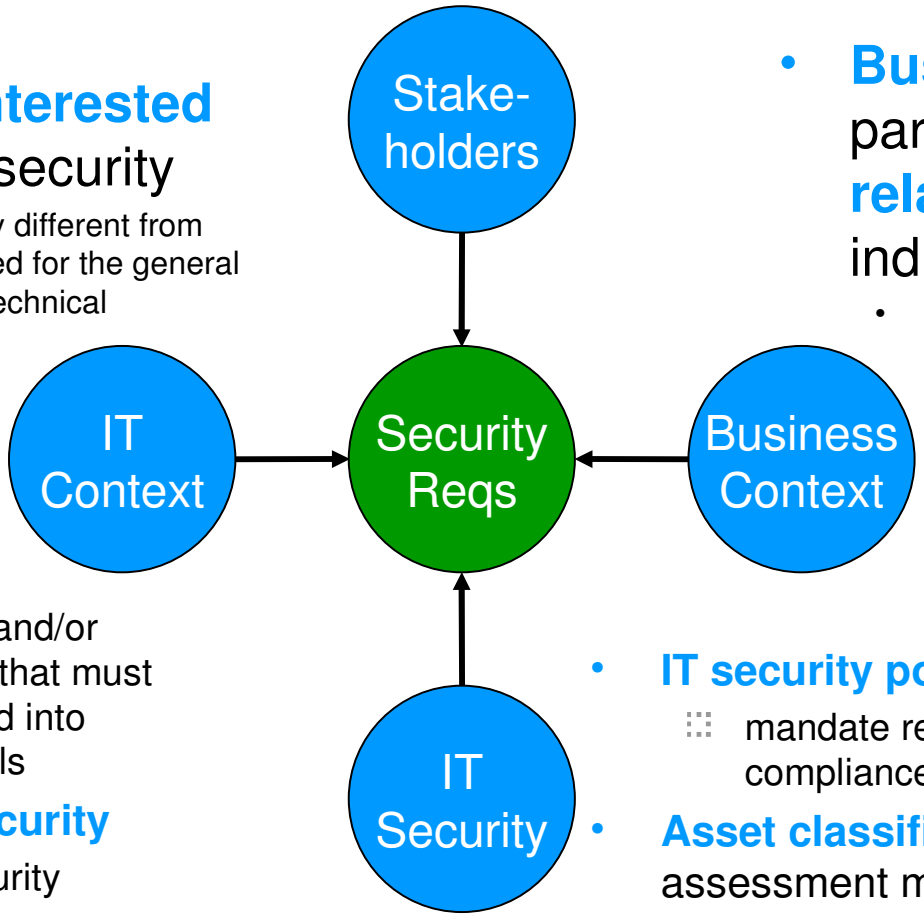
Like other branches of the IT architecture process, Security Architects rely upon patterns for the basic structure of a solution

- **Reference architectures**
  - Provide patterns for a particular class of IT solutions – IBM maintains internal reference architectures for use by its architect community
  - Reference architectures should include patterns for addressing security within an instantiation
- **Product/Supplier-specific patterns**
  - Security component suppliers often provide patterns that show how their products can be deployed as part of a business system
  - IBM's *Patterns for eBusiness* has several patterns that show how an ebusiness solution can address security requirements - http://www.ibm.com/developerworks/patterns
- **Business solution level patterns**
  - For example the SAP security concept shows how the various package security controls are used, and identifies what controls the infrastructure must provide for secure operation
- **Function group patterns**
  - It is often useful to have conceptual model for a particular grouping of security function
    - IBM's security architecture methodology includes models showing the basic components that make up a particular service, an audit service for example
  - Provides a model for analysing how the function is addressed within an architecture

Security :
Requirements & Functional Architecture

External to the project, security requirements come from understanding the business and technical context in which an application or service exists

- the set of **interested parties** for security
  - may look very different from those identified for the general business or technical viewpoints



- **Business drivers**, partner **relationships**, industry **portals**, etc
  - influence the types of **trust relationships** and **access paths** that must be supported, and therefore the security controls required

- **Corporate IT architectures**
  - pre-requisites and/or dependencies that must be incorporated into security controls
- **Enterprise IT Security**
  - mandated security standards, technologies, and services

- **IT security policies** and standards
  - mandate requirements – requiring compliance or exception
- **Asset classification** and risk assessment methods

Common influences in IT Security

- Conform to Corporate Security policies & standards
  - May include external and industry standards
  - Internally defined policies and procedures
  - Enforced usage of already selected technologies
- Minimising impact to users, e.g.
  - Single Sign On – the ability for a user to logon just once in order to be granted access to multiple systems
- Resilience – Maintain operations in the face of attack

Models for Security are commonly derived from recognised Standards in the field of Information Technology Security.

Security related Standards	General Description
National Government Standards US TCSEC (orange book), FIPS UK ITSEC CA CTCPEC	Sets of specifications and evaluation criteria for Trusted Computing products. *In most cases, these have been superseded by IS 15408,* ***Common Criteria.***
International Standard 7498-2 ISO/IEC 7498-2 (also ITU X.800)	System level security, to include: security services, mechanisms, management
International Standard 17799 ISO/IEC 17799 (also BS 7799)	Code of Practice for Information Security Management, including design and deployment of security processes, technology focus areas as well as compliance reviews`
International Standard 15408 • ISO/IEC 15408 (also Common Criteria)	Combined and updated evaluation criteria from national security standards plus a product evaluation and certification method
Internet Reference Documents RFC 2196 Site Security Handbook RFC 2504 User Security Handbook RFC 2828 Internet Security Glossary	General guidance for site security and user security and security terminology for the Internet environment
Industry Group Standards J2EE Security (from Sun) PKIX (from Internet Mail Consortium) WS-Security	J2EE – Java PKIX – Public Key Infrastructure (digital certificates) WS-Security – family of standards specifying security services to support Web Services applications

From a security viewpoint, a solution has two aspects which must work together to deliver end-to-end security for a business system

Application (functional) security aspect

- The application runs "within" a secure infrastructure
- Authentication of users and authorisation of their actions
- Control of access to information, including data privacy
- Protection from unauthorised disclosure or modification of information, in transit and in storage (including backups) – including data protection
- Capture, storage, protection, and management of transaction-level audit trails

Infrastructure security aspect

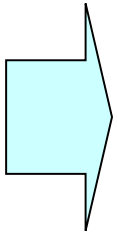
- The infrastructure supports one or more business applications
- Secure server and middleware environment
- Network-level access controls
- Identity and Access Management infrastructure
- Desktop security environment
- Wide Area Network environment

- These aspects are often built and maintained separately
  - For example an application hosting centre
- When a project encompasses both aspects it may be helpful to view them as separate mini-projects to maintain the clear distinction between application and infrastructure security controls

A conceptual model for security functions from a common set of security-related requirements

Identified security "Common Criteria" functional requirements classes:

- Security Audit (FAU)
- Communication (FCO)
- Cryptographic support (FCS)
- User data protection (FDP)
- Identification and authentication (FIA)
- Security management (FMT)
- Privacy (FPR)
- Protection of functions (FPT)
- Resource utilization (FRU)
- TOE access (FTA)
- Trusted path/channels (FTP)

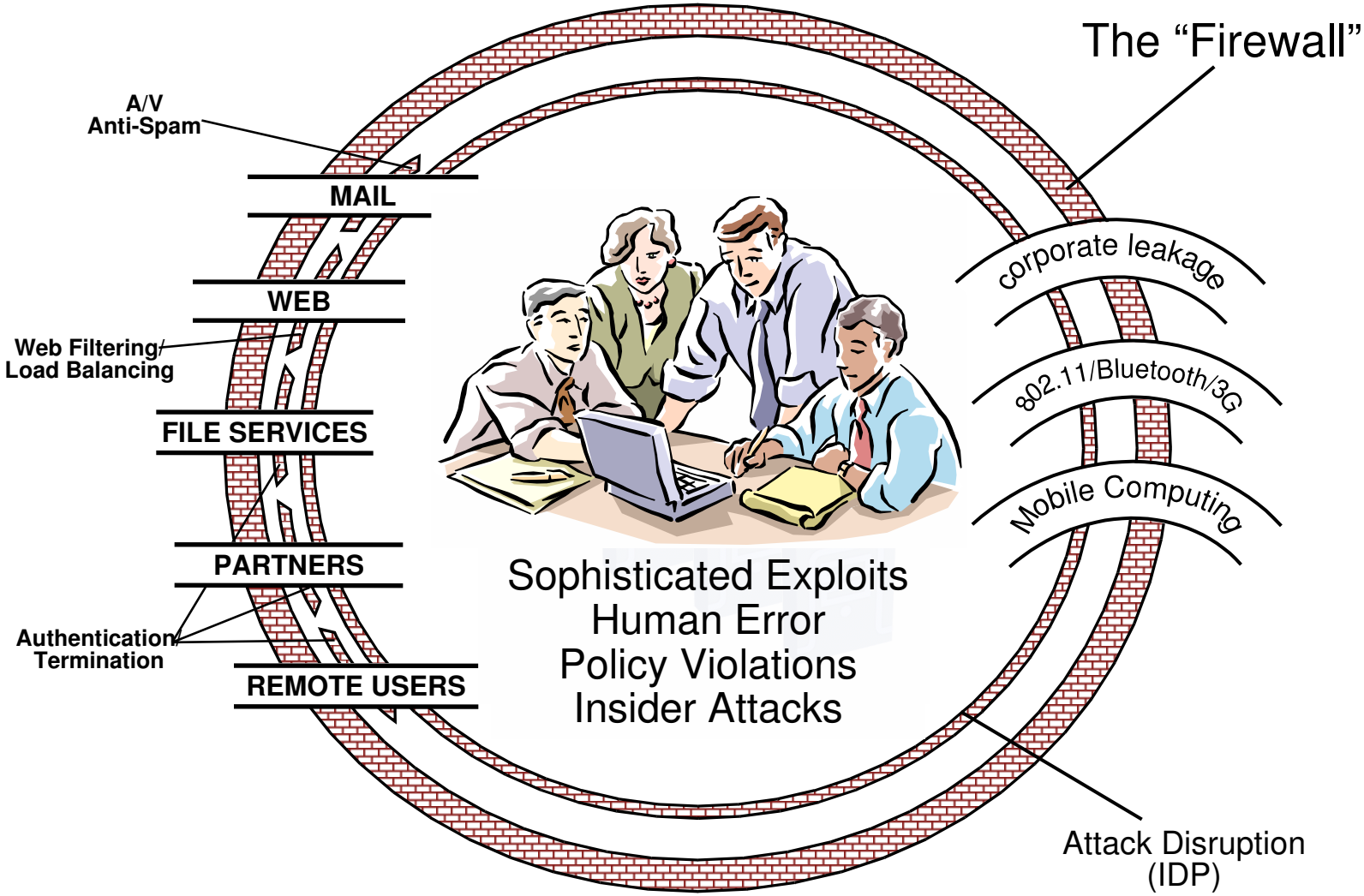


Security Subsystems

- Credential Subsystem
- Access Control Subsystem
- Information Flow Control Subsystem
- Security Audit Subsystem
- Solution Integrity Subsystem

Security : Technology and Operational Architecture

Increasing expectations, range of channels and IT complexity has increased the Security challenge



In order to help us structure the infrastructure necessary to protect the enterprise, we employ the concept of Zones

Security Zones might be classified (and colour-coded) as follows:

- **Uncontrolled** – anything outside of the organisation,
  - including, but not limited to the home, street etc.
  - via a wide number of channels including, but

- **Restricted –** where access is restricted to users or systems that are trusted to some degree
  - For example, a user or system in a controlled zone

- **Controlled** – where access is limited, but users are allowed access on a controlled basis.
  - Public access to a DMZ.
  - Employee access to a corporate LAN

- **Secured –** where access is available to only a small group of highly trusted users or systems.
  - access to one secure area does not necessarily give you access to another secure area.

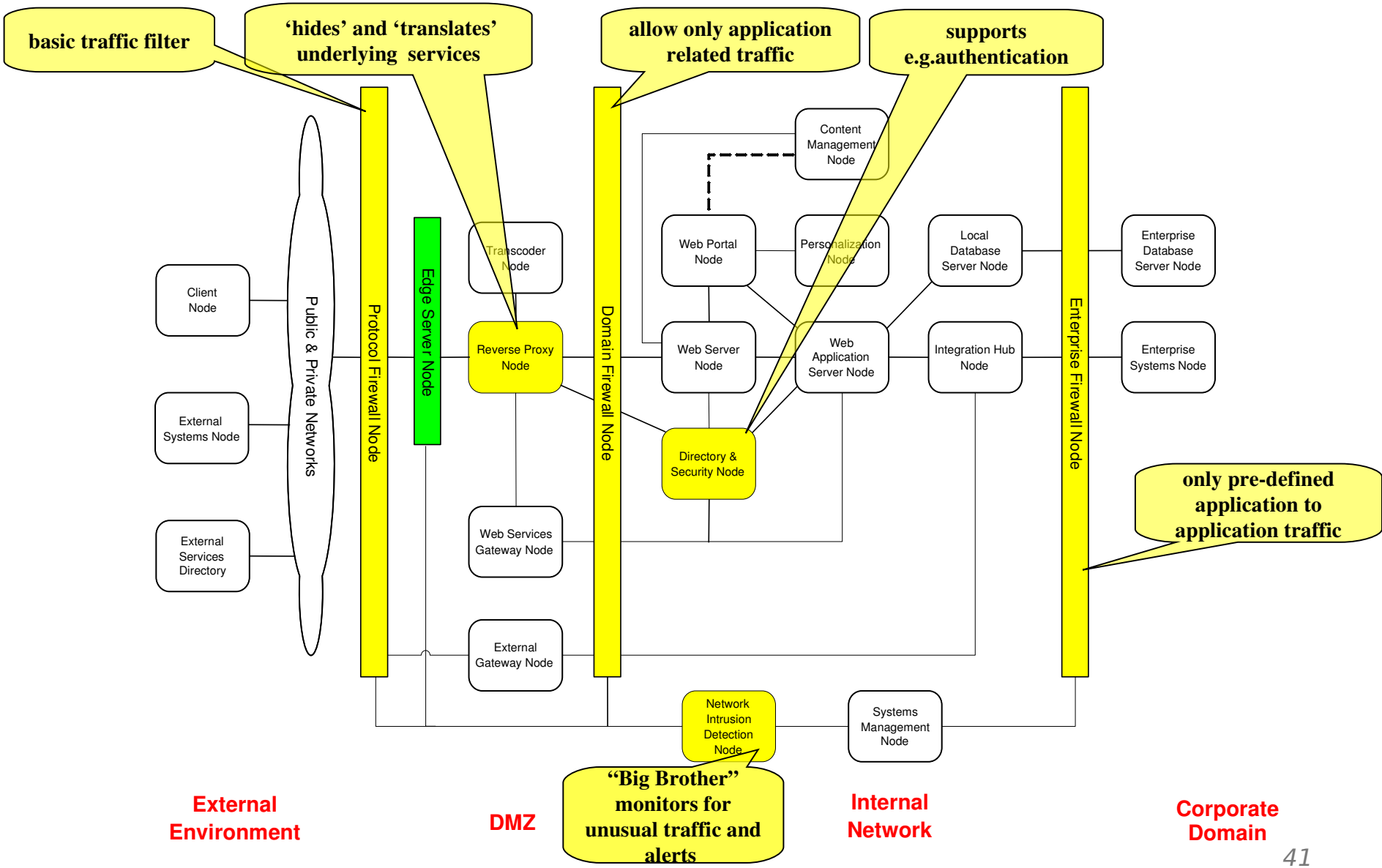
We need to elaborate the zone classification to reflect who has management control of a zone…

- Descriptors may be added to a zone classification – for example:
  - **External –** An external zone has the same characteristics as defined above,
    - control is in the hands of an external organisation *with which this organisation has a contractual relationship,*
    - The external organisation has a responsibility to operate the zone according to their own security policies.
  - This is distinct from an outsourced service provider relationship, where the security controls are operated as part of a service being provided on behalf of the Council and are consequently considered to be part of the Council's infrastructure.

Common Security related infrastructure components

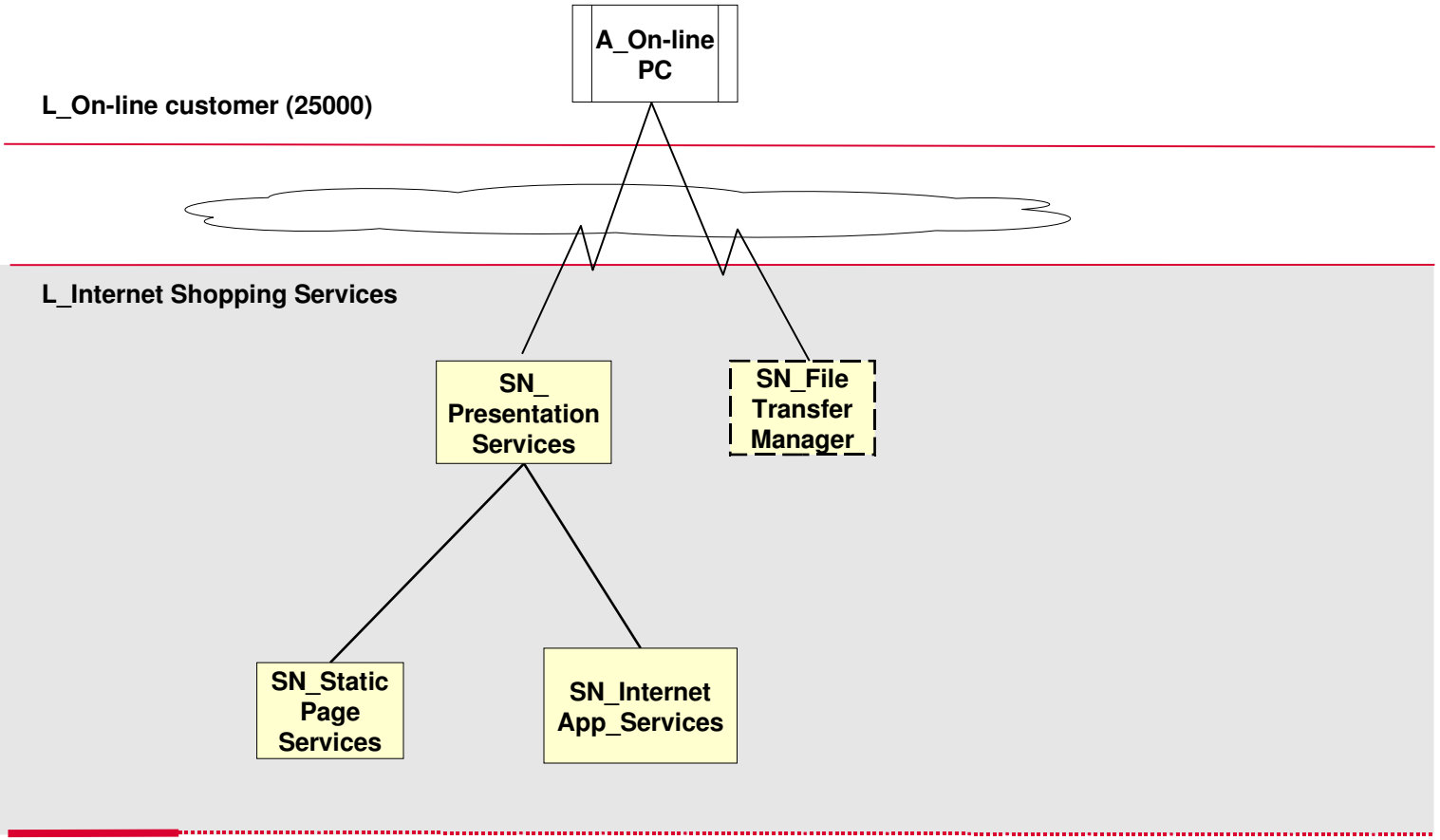
- **Firewall**
  - A hardware or software component which protects against unauthorised network access into or out of a particular zone
  - Firewalls aim to filter unwanted traffic out by observing packet contents and applying rules
- **Security & directory servers**
  - Dedicated servers hosting components managing user databases including user credential and profile data
- **Intrusion detection systems**
  - Components placed within the architecture with the explicit role of detecting intrusions
- **Cryptographic hardware components**
  - Cryptographic operations in software can be very time consuming
  - For secure systems, it is common to implement specialised hardware to perform necessary cryptographic functions quickly

Security and access related Nodes in the IBM e-Business Reference Architecture Logical Operational Model (v2.3)

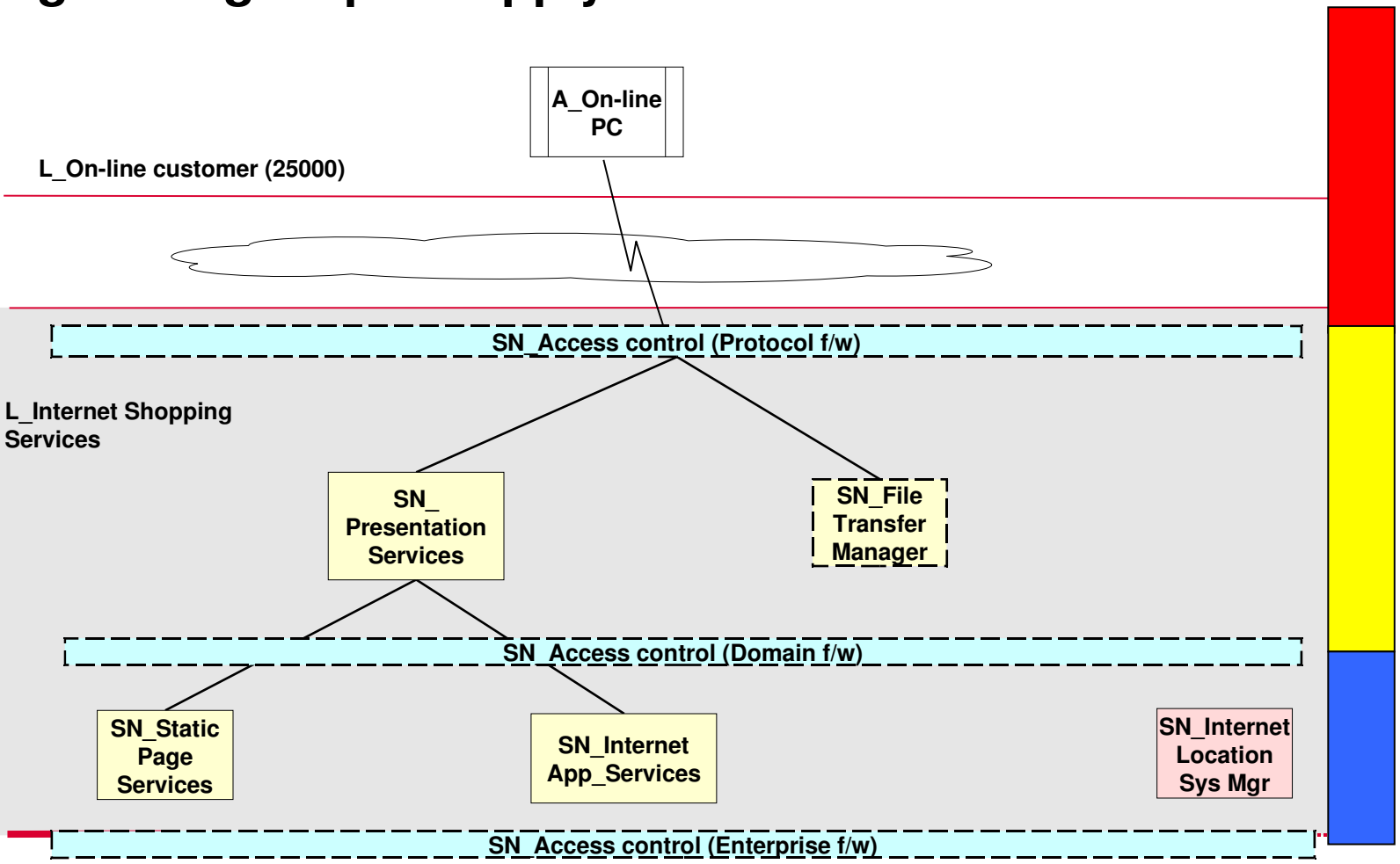


We can use the concepts of Zones and the Reference Architecture to strengthen an Operational Model

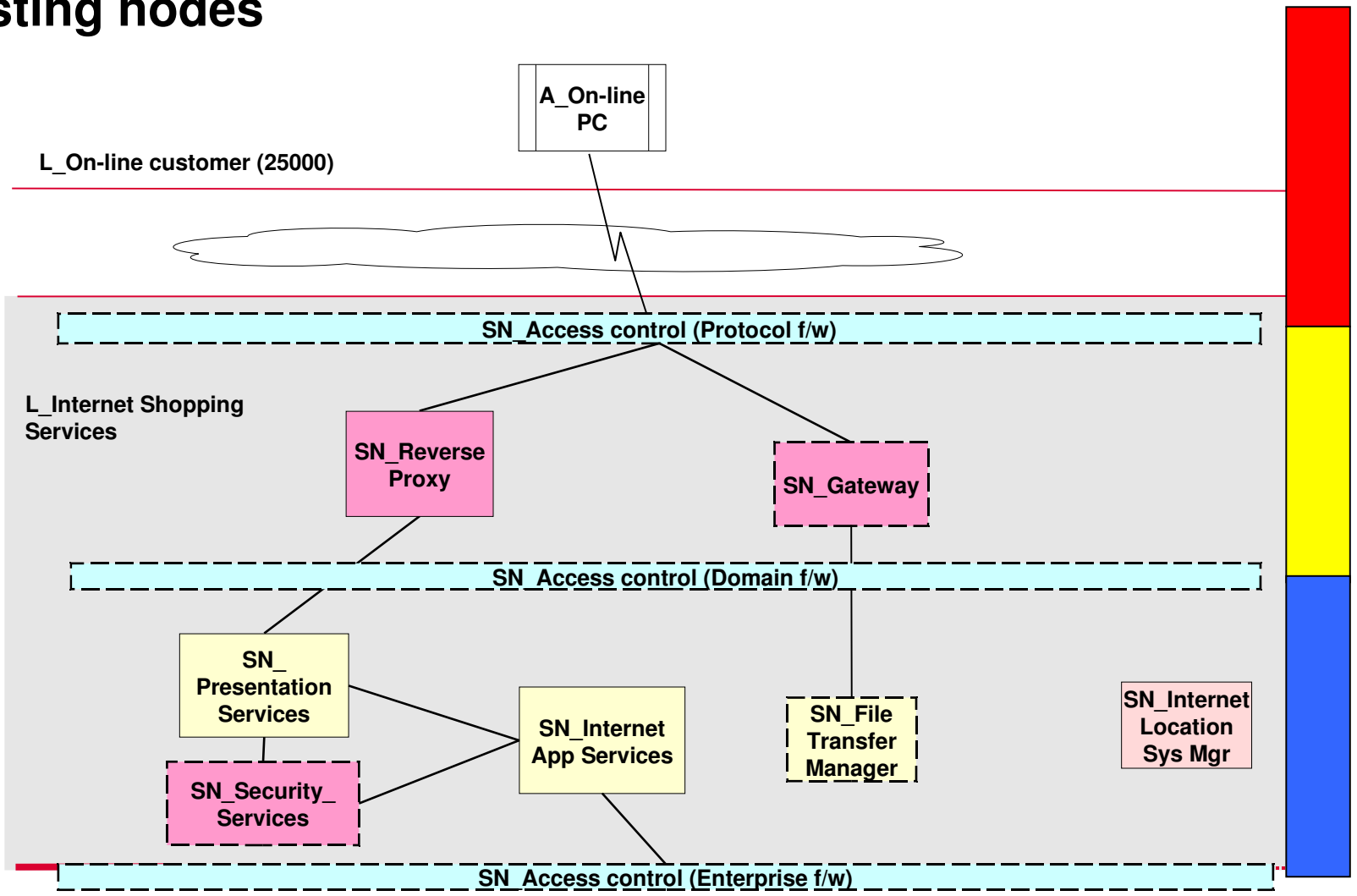
Starting point – simple (and insecure!) architecture



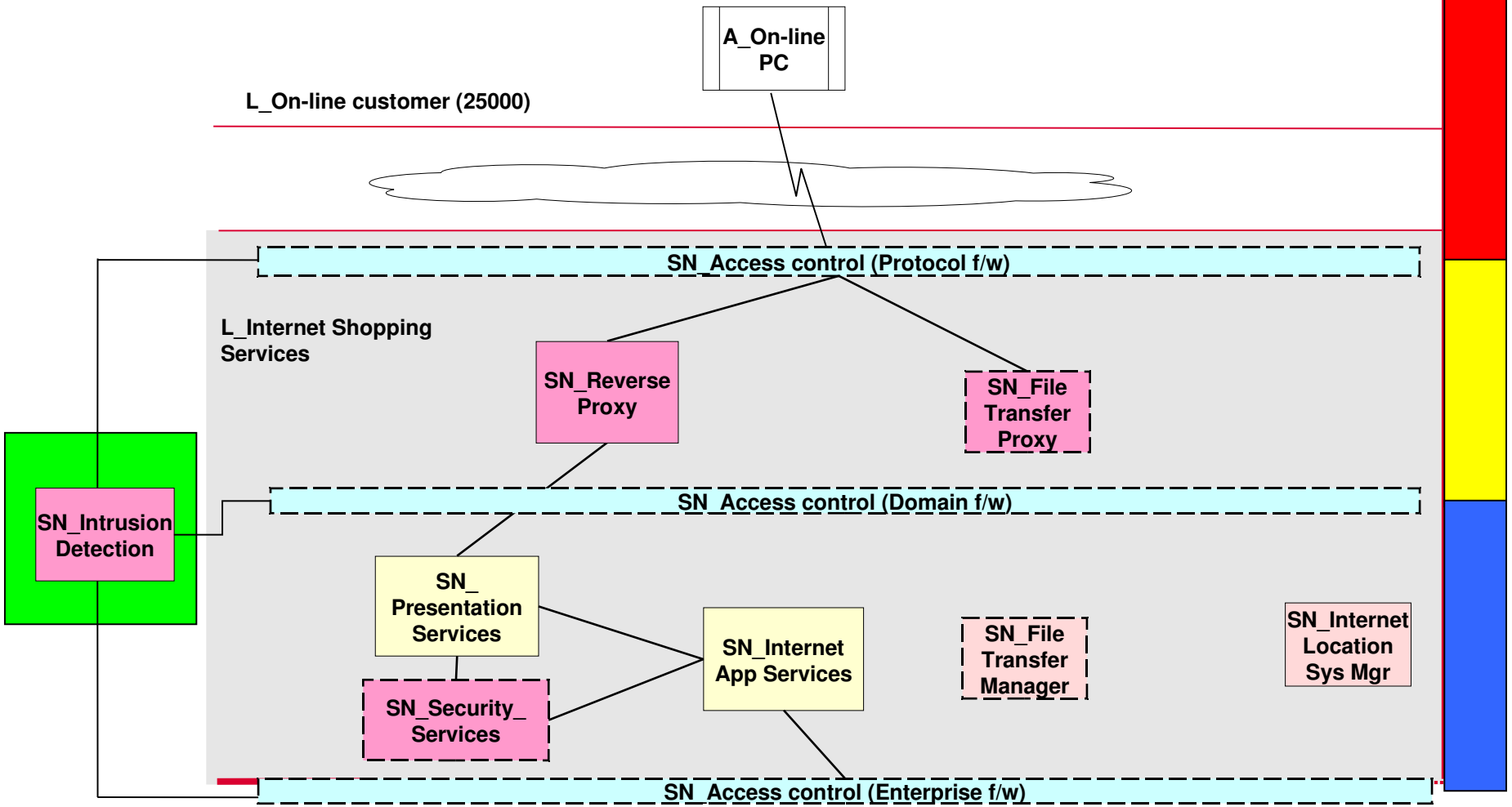
Strengthening step 1: Apply firewall and zone model



Strengthening step 2: Add security nodes and replace existing nodes

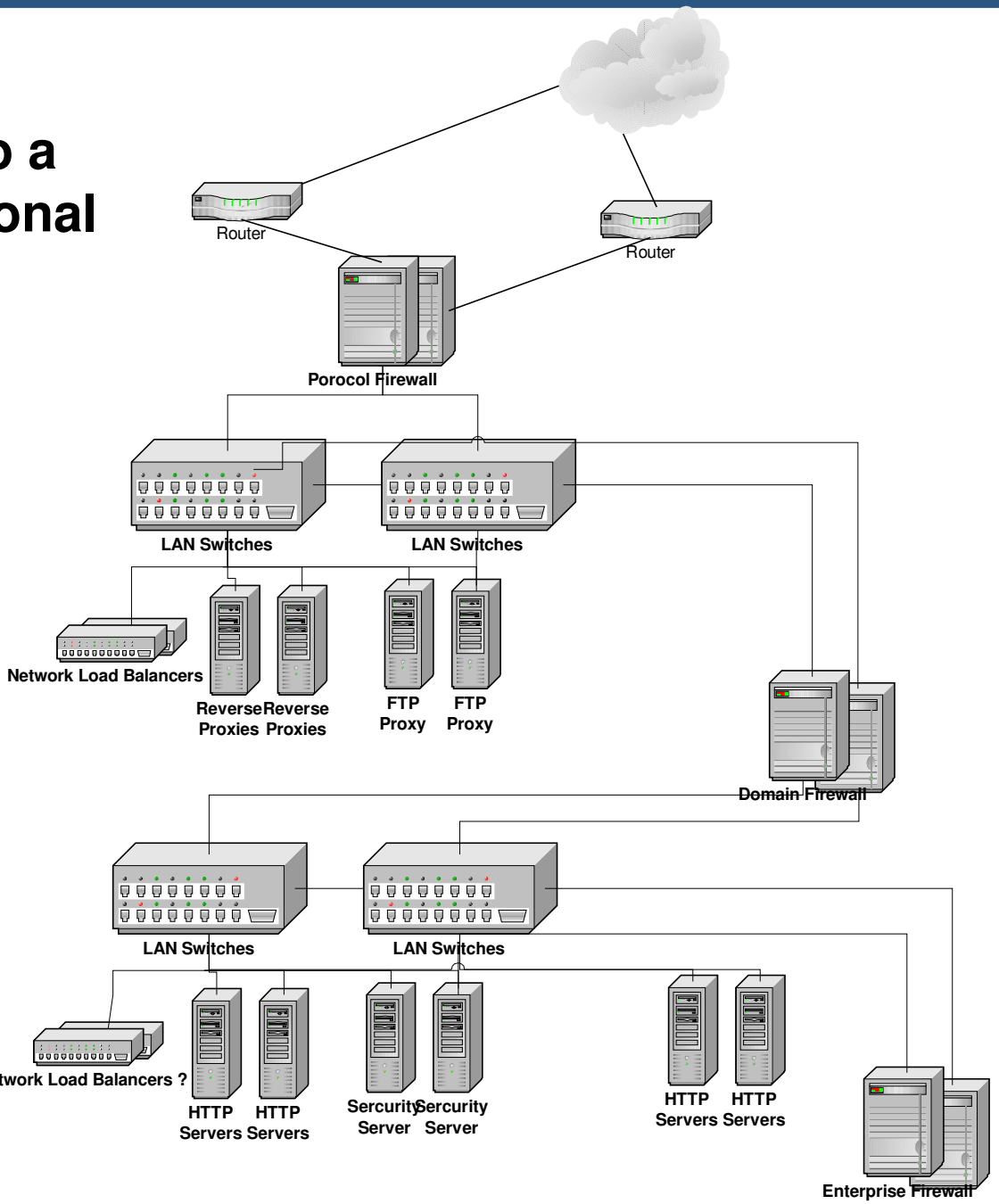


Strengthening step 3: Add intrusion detection



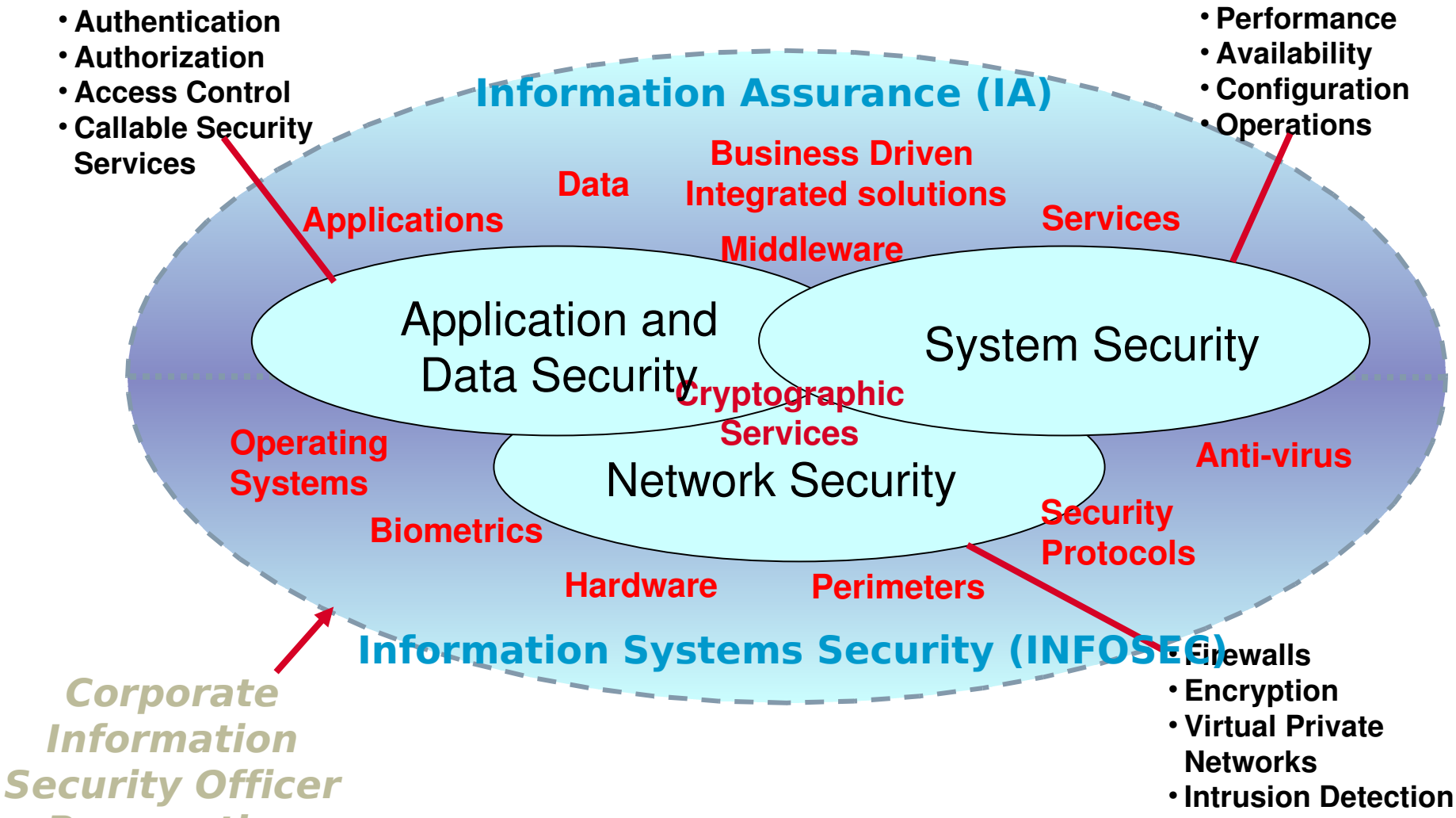
Example transformation to a Physical Operational Model

- 100Mbps LAN chosen for cost effectiveness
- Cisco 3550 Switches used for LAN infrastructure
- Cisco PIX Firewalls

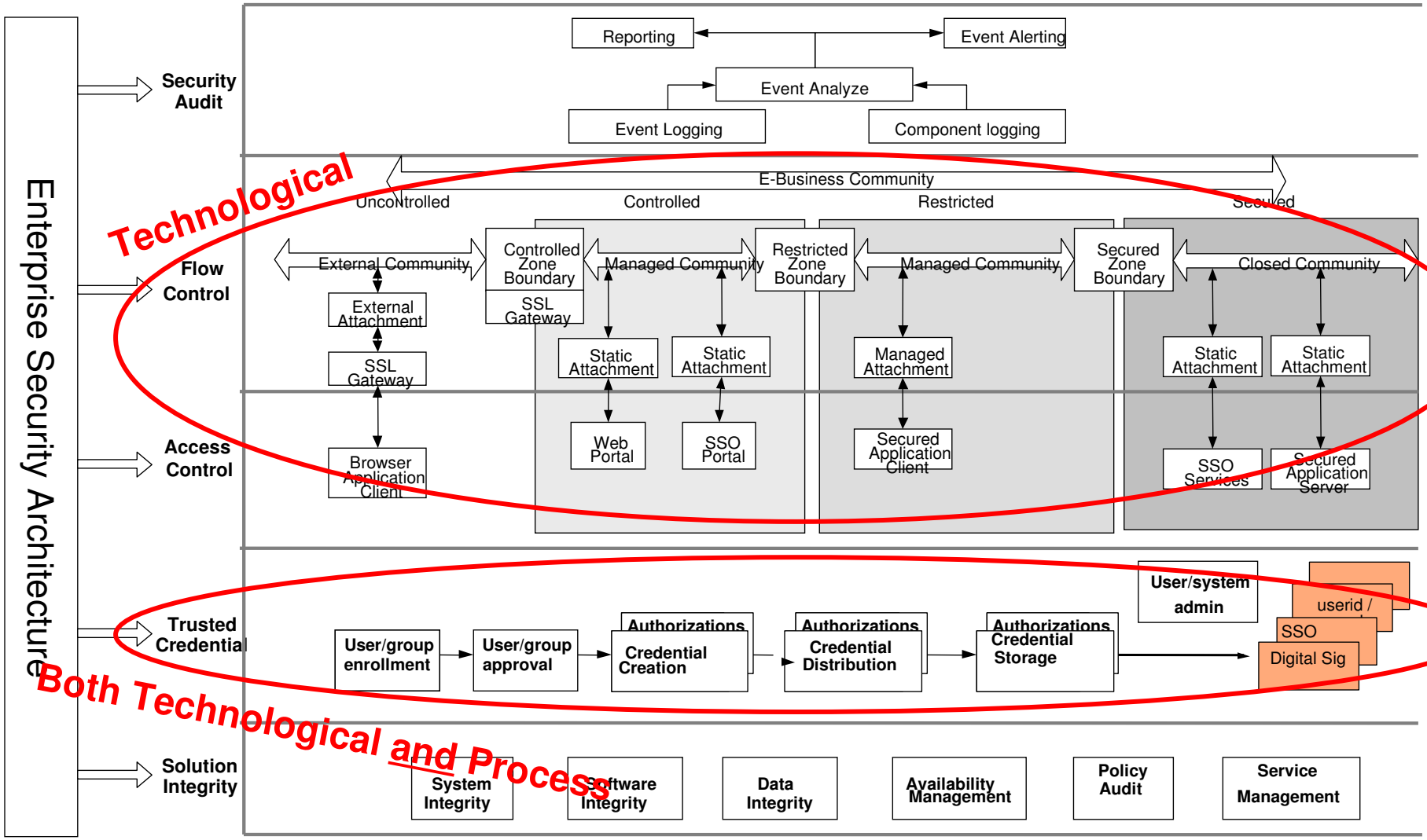


Security : Summary

The 'big picture' – security policy and architecture must include logical and physical protection to counteract the threats



In implementing the defined policy in an IT Architecture, both process and technological elements must be considered



Accessibility, Usability & People Centred Design

Accessibility, Usability and People Centred Design

- Consider:
  - Accessibility – making systems available to as wide a range of people as possible
  - Usability – making systems easy to use
- Both of these elements are complex topics in their own right, and though they have some similarities, they have a different focus
- The slides give an overview of a process that can be used – the work is specialised, but it is useful for the IT Architect to have some understanding of the challenge

Accessibility & Usability: Background and Drivers

Why bother with making technology accessible?

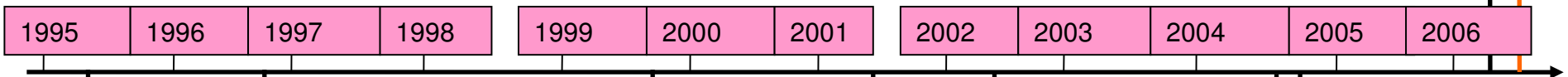
The key drivers:

- **Inclusion … in the UK:**
  - Over ***10 million people*** are registered with a disability
  - Over 2 million people are blind or partially sighted
  - Over 9 million people are affected by deafness and hearing loss
  - Over 7 million people have literacy problems
  - Over 1 million have learning difficulties
- **Legislation**
  - UK The Disability Discrimination Act 1995, Part II Employment 1996, Part III Goods & services (1999), DRC Code of practice (2002), Disability Equality duty (2006)
  - The Employment Equality (Age) Regulations 2006
- **Employment**
  - Ageing workforce: *Adapting to the physiological and cognitive needs of an older workforce*
  - Labour engagement: *Lowering the skill required to use technology in the workplace*

Legislation DDA

Disability Equality duty (December 2006) Covers the duty of care of public sector organisations to include equality for disabled people in the culture of the organisation

DRC funds a BSI PAS78 to provide best practice guidance on commissioning accessible websites



The Disability Discrimination Act 1995 introduces new laws giving disabled people new rights in the areas of employment, access to goods, facilities and services and buying or renting land or property

DDA Part III **Access to Goods and Services** came into force on 1 October 1999, covering the need for service providers to make **reasonable adjustments** to the way they deliver their services so that disabled people can use them.

March 2002, DRC Code of practice clarifies that services provided through **websites** that are covered by DDA Part III are subject to the Act

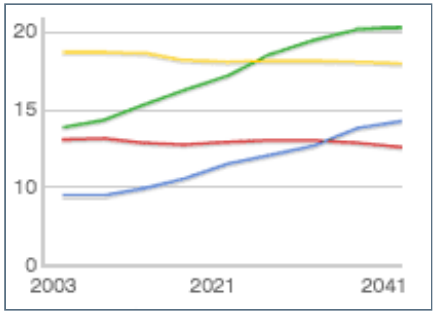
DRC study finds that >80% of public websites fail to meet minimum accessibility standards

DDA Part II came into force on December 1996 aimed at protecting disabled people from discrimination in the field of **employment**. The code of practice covers companies making "reasonable adjustments" to computers systems to allow access.

The Special Educational Needs (SEN) and Disability Bill came into force May 2001 making it unlawful for **education providers** to discriminate against disabled pupils, students and adult learners.

DDA Part III came into force, 1 October 2004, applying to service providers where **physical features** make access to their services impossible or unreasonably difficult for disabled people.

Ageing workforce



"the reality is that, as older people become an ever more significant proportion of the population, society will increasingly depend upon the contribution they can make."

Tony Blair

- By 2025, more than a third of the UK's population will be over 55.
- There is a trend of extended working life. The long term aspiration is:
  - To achieve an employment rate equivalent to 80% of the adult population, including:
    - One million **older workers** into employment
    - One million **people moving from Incapacity Benefits** into employment
- An ageing population will require accessible technologies:
  - With age, people develop new physiological and cognitive impairments.
  - With age, mild difficulties and impairments become more severe.
  - In our society, the total number of people with difficulties and impairments will increase.

Why bother with making technology usable?

The key drivers:

Increase sales

- For each $1 spent on improving the visual design or style of your site, there will be virtually no improvement in sales.
- The same $1 spent on improving core behavioral interactions with a site's critical way-finding and form-filling functions, will however, return $50-$100 if done professionally and rigorously.
- For each $1 spent acquiring a customer, it will cost $100 to re-acquiring them after they leave because of poor usability or bad customer service.

Reduce costs

- The single largest predictor of call center volume is web site's usability. Calls average $22-$30 per call.
- For every $10 spent defining and solving critical usability problems early in development using professional usability disciplines, saves about $100 in development costs.

Other business drivers

- Safety, efficiency, adoption, satisfaction, effectiveness, flexibility, inclusion

Usability is an example of a run-time quality

- Usability is defined as “the design of interactive systems used by people to satisfy personal and organisational goals.”
- Interactive systems
  - Any technology, any platform
  - Desktop, thin-client, intranet or Internet, mobile, and so on
- People
  - Any direct or indirect user of a system
  - Staff, managers, customers, citizens, learners, and so on
- Goals
  - Make money, save money, time, and lives and so on
  - Communicate, engage, persuade, retain, and so on
  - Find, buy, learn, grow, progress, and so on

Today's picture: the majority of technology is not even technically accessible

- Only 3% of the 436 online Public Service websites in EU were considered to meet minimum accessibility standards
  Source: Cabinet Office report November 2005
- 81% of UK websites failed to satisfy basic accessibility criteria
  Source: Disability Rights Commission Study 2004

Last Updated: Wednesday, 14 April, 2004, 08:30 GMT 09:30 UK

E-mail this to a friend Printable version

Websites 'failing' disabled users

Geoff Adams-Spink
BBC News Online disability affairs reporter

An investigation by the Disability Rights Commission shows that most websites are unusable by disabled people.

Stuck on the hard shoulder of the information superhighway

This means that many everyday activities carried out on the internet - booking a holiday, managing a bank account, buying theatre tickets or finding a cheaper credit card - are difficult or impossible for many disabled people.

> "Few designers seem to care that they are excluding millions of people from seeing or using the sites they are building"

And many interfaces have usability problems

A study from Zona Research found that:

- 62% of online shoppers gave up at least once while looking for the item they wanted
- 20% of online shoppers gave up more than three times during a two-month period
- 42% turned to traditional channels to make their purchase

A study by research group Creative Good found that:

- 39% of the customers who tested the sites for the study could not figure out how to buy
- More than 50% of search attempts failed to find something relevant.

A study cited in “Build a Site, Not A Labyrinth” (Jefferey, G.) stated that:

- 33% of online banking customers closed their accounts within a year. 50% said it was because the site was too difficult to navigate

A study by Jared Spool's found that:

- Users could only find information 42% of the time even though they were taken to the correct home page before they were given the test tasks

And some real examples of usability failures

- London Ambulance service implemented a new dispatching system. **Severe delays in ambulance arrivals** caused by technology and user interface design errors.

• "A financial services company had to scrap an application it had developed, when, shortly before implementation, developers doing a User Acceptance test **found a fatal flaw in their assumptions about how data would be entered**. By this time, it was too late to change the underlying structure, and **the application was never implemented."**

Some definitions…

Technical accessibility:

- Ensures software **works** with assistive technologies such as screen readers and with browser settings. Common problems include lack of keyboard support and fixed text sizes.
- Ensured software is designed so that **alternative adaptations** can be created e.g. for different user types, environments and devices

Usable accessibility:

- Ensures disabled users can **use** software using their assistive technologies or through personal settings. Common problems include lack of support for users tasks, poor navigation and inconsistencies in terminology.
- **Interfaces are optimised** for different user types, environments and devices

(PAS) 78

Universal usability:

- Aims to ensure that large **percentages of a population can use** and are **motivated to use** digital services irrespective of wealth, language, technology experience and social background.
- Services are device independent , offered through a range of **different access mechanisms**, are user customisable and can be configured for different user profiles. Variations of user control are offered.

Research, pilots, more investment needed

Some more definitions…

The extent to which a product can be used:

- **By specified users**
- **To achieve specified goals**
- **with effectiveness, efficiency and satisfaction**
- **In a specified context of use**

[ISO 9241-11]

is about designing processes and interfaces that:

- **Improve the way customers interact with a company**
- **Improve the way employees do their job**
- **It covers products and services, environment, system policies and human interactions**

Accessibility & Usability: Method and Approach

Inclusive design relies on a rigorous process

1. Business opportunity

Defining business goals, stakeholder value, **target audiences**, opportunities, risks, segmentation.

2. Understanding users

Researching goals, **values**, **tasks**, context of use, **barriers to use**, environment, **access mechanisms**.

3. Conceptual design

Creating consistent concepts and behaviours matching **user's cognitive constraints**

4. Physical design

Applying crafted and flexible representations to increase **access**, credibility and appeal.

5. Development & Training

Coding validated concepts and designs following **defined accessibility standards**

Evaluation is central

Iterative evaluations remove errors, **check access**, reduce risk and **ensure targets are met.** Evaluations can also be used to identify and quantify new opportunities

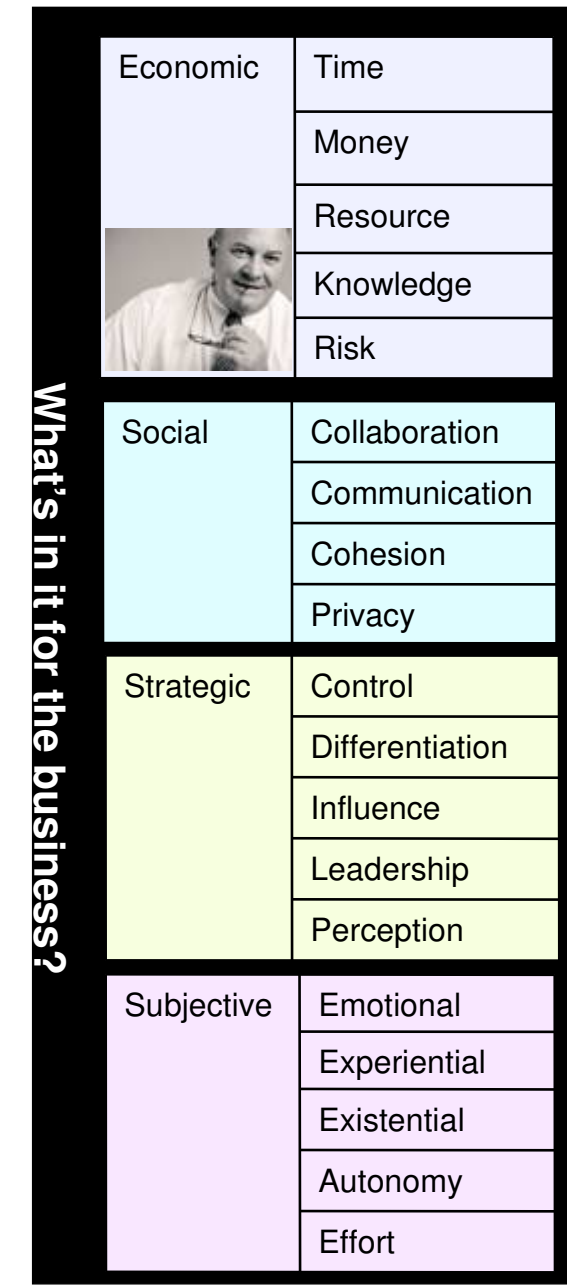
(P.S. Many standard work products exist within the IBM GS Method to help the Usability and Accessibility design processes)

- Usability
  - APP 129 Usability Requirements
  - APP 130 Use Case Model
  - APP 142 Current Solution Evaluation
  - APP 143 Early Usability Evaluation
  - APP 145 Use Case Validation Report
  - APP 146 User Interface Conceptual Model
  - APP 146 User Interface Design Guidelines
  - APP 146 User Interface Design Specifications
  - APP 146 User Interface Prototype
  - APP 146 User Profiles
- Business
  - BUS 320 Customer Needs and Wants
  - BUS 411 Business Direction
- Organization
  - ORG 017 User Support Specifications
  - ORG 153 User Support Materials
  - ORG 307 Current Organization Assessment
  - ORG 308 Human Capability Assessment

Understand the business opportunity

Understanding the business context, goals and vision for the project, such that the User Experience Design team are properly focused.

- This will include defining and prioritising:
  - Business goals:
    - E.g. Make money, save money, communicate, engage, persuade, retain, find, buy, progress…
  - Target audience:
    - E.g. Claims handlers, Supervisors
  - Measures:
    - E.g. % task success through claims process, Reduction of call centre queries about a claim
  - User experience goals
    - E.g. Efficiency, effectiveness, satisfaction. Ease of Learning, credibility, compliance ….
- And understanding
  - Current application/process/website:
    - E.g. current task support, design innovations, usability barriers
  - Current customer/employee data:
    - E.g. Customer or employee feedback, survey results, queries

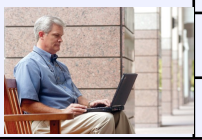


Understanding users

Gathering data about the target audience is critical to success

- **Who** and **how many** need to be included in the study:
  - User profiles are created to capture
    - Target user characteristics (Age, gender, experience),
    - Social and Environmental context of use,
    - Language,
    - Usability factors (that drive the design).
    - Representative users are then invited to participate in user research studies.
- What **data** needs to be collected:
  - User researchers design the study to collect necessary data such as
    - user goals, tasks, barriers to use, terminology, classification, mental models.
- How the data will be **gathered**:
  - Study methods are selected such as
    - Field studies (ethnographic studies; contextual enquiry),
    - Workshops (short on time),
    - Focus groups (well defined audiences; easy to get),
    - Interviews (often used in combinations with another method),
    - Surveys (large statistical sample; difficult to get to see the users)

What's in it for the users?

Economic	Time
	Money
	Resource
	Knowledge
	Risk
Social	Collaboration
	Communication
	Cohesion
	Privacy
Strategic	Control
	Differentiation
	Influence
	Leadership
	Perception
Subjective	Emotional
	Experiential
	Existential
	Autonomy
	Effort

Define and agree critical requirements

- Provides an opportunity for the User experience design team to **feedback** to the business and the technical implementation team about the the **key findings** from the stakeholder and user research studies.

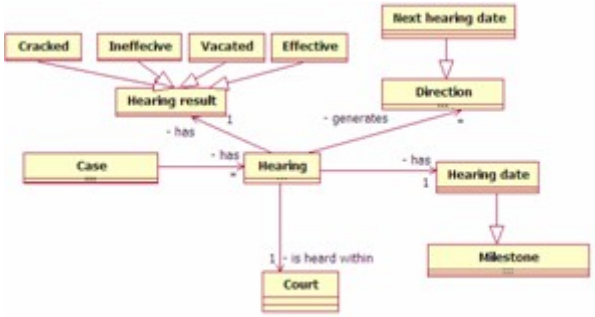
- Enables the group to collectively identify any **business or technical constraints** that could impact the design direction.

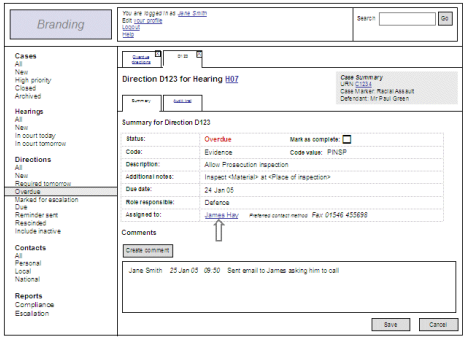
- Provides a forum to **reassess** business, design and development **priorities** as a result of the user research findings.

Conceptual design

In general, 70% of usability problems are as a results of errors within the conceptual model

- Many problems relate to a poor information architecture
  - *It is not clear to users where the information is*
  - *Users are unsure of specialist terminology*
- Conceptual design involves:
  - Modelling human activity using task models
  - Modelling objects, labels and relationships using information modelling
  - State modelling is also used to capture the lifecycle of complex objects
  - Creating a wire frame to test with users
  - Reworking the design to remove usability errors

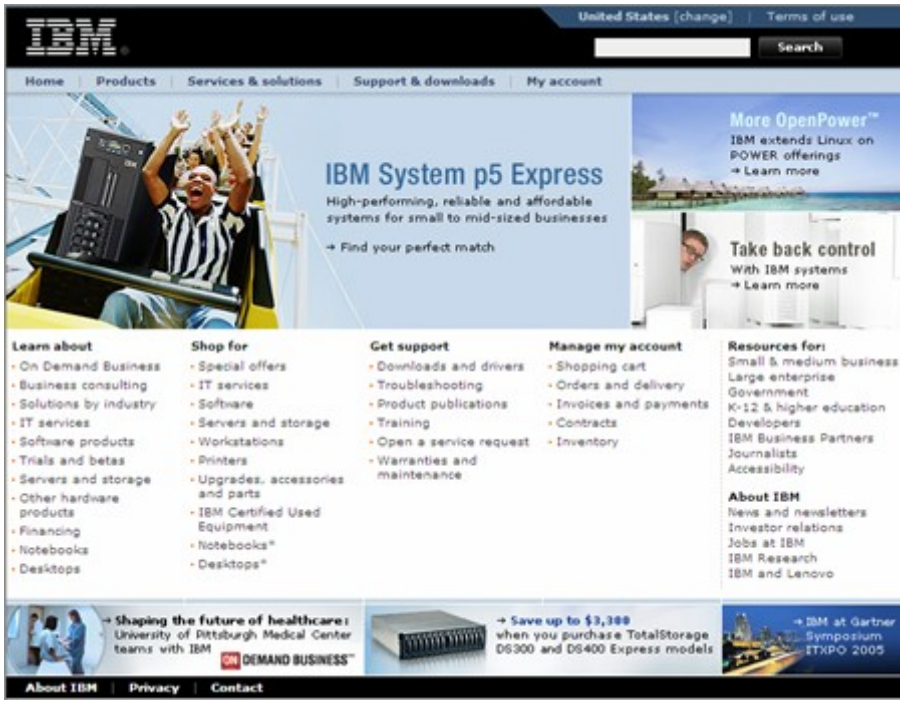




Physical design

Applies branded look and feel

- Finishes the design by defining and applying system ‘look and feel’
- Produces a user interface specification derived from the style guide
- Generates high-fidelity graphical and sometimes interactive prototypes

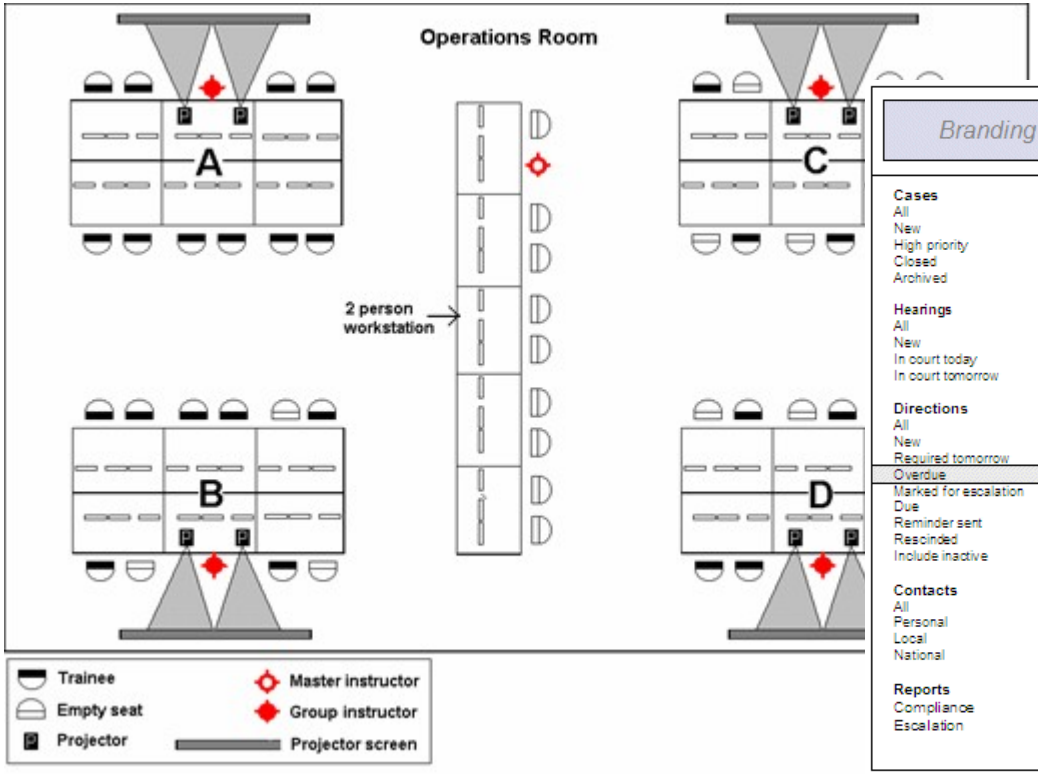


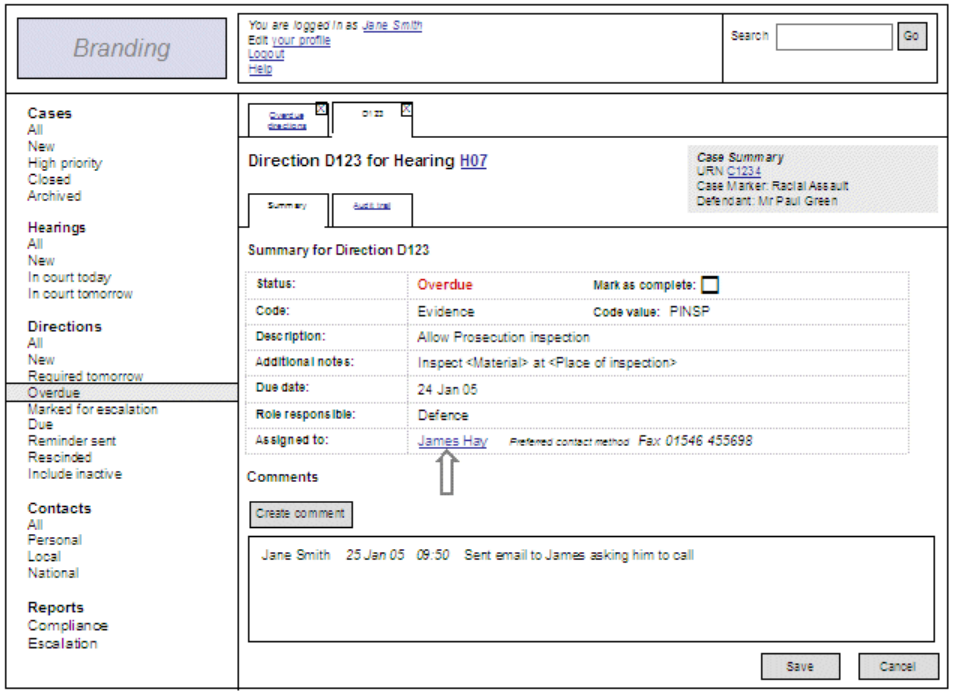
Documents agreed UI elements

- Ensures key elements are identified and documented as part of a style guide to
  - Protect critical assets
  - Assist future designers/developers to apply the correct design

Evaluation

- Evaluation tests designs in context:
  - By observing representative users attempting typical tasks
  - By eliciting users' opinions
  - Through structured analysis by user interface specialists and ergonomists

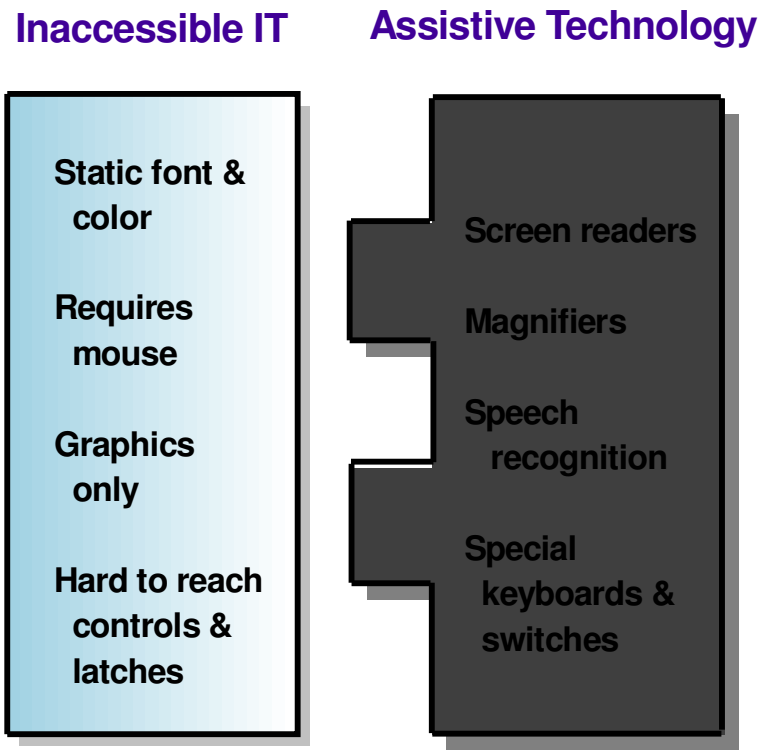


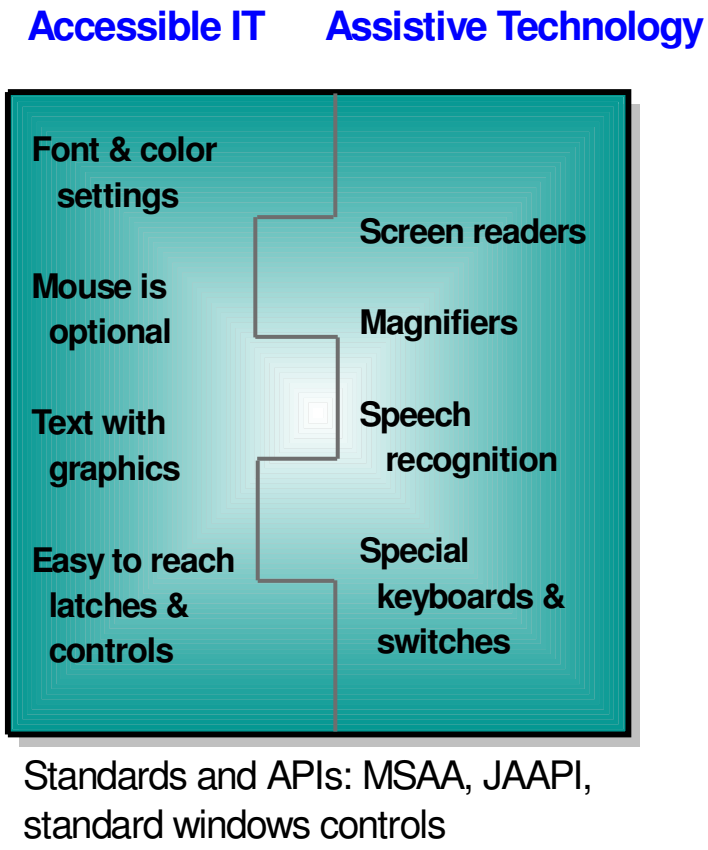


Accessibility & Usability: Solutions

"Accessibility" is both a quality and a constraint, for which however there is technology to assist us

- **Assistive Technology:** Specialised IT that allows a user with a disability to access Information Technology

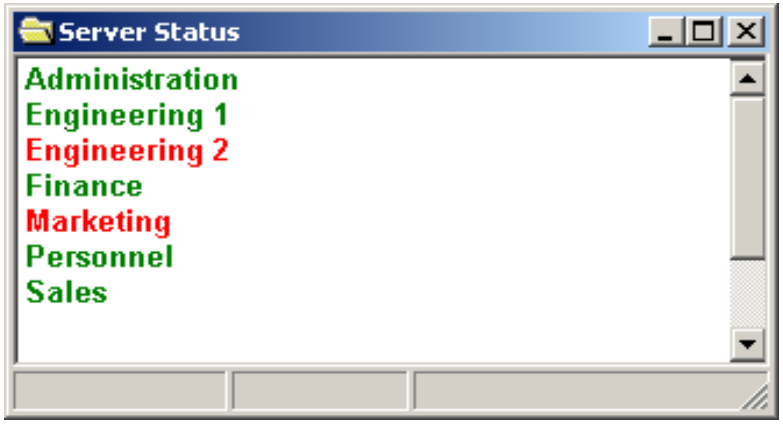


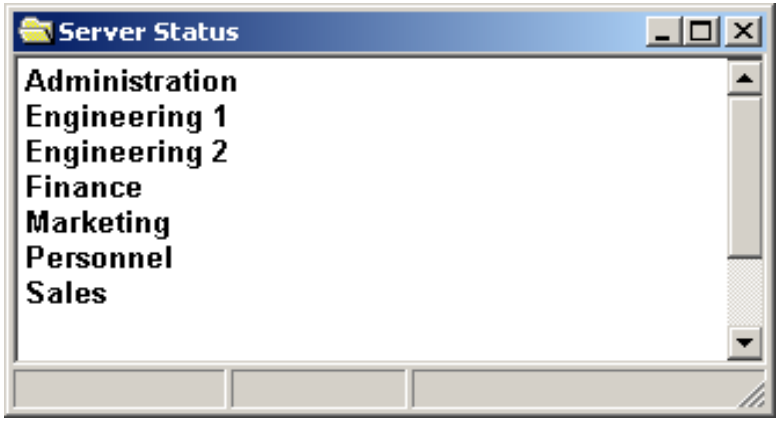


Standards and APIs: MSAA, JAAPI, standard windows controls

What are some examples of systems that comply with IBM and Government accessibility guidelines?

- Users with low vision need enlargeable fonts and high contrast settings.
- Users who are colour blind need more than colour differences to communicate information.
- Users who are blind must use a screen reader and the keyboard.
- Deaf users need captions and visual equivalents for audio alerts
- Hard of hearing users need to increase the volume.
- Users with limited or no use of their hands need keyboard accessibility features and alternative input methods.
- Users with attention or reading disabilities need speech synthesis, speech input, word prediction, highlighting tools, and so on.

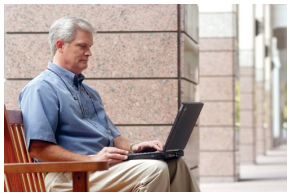




Accessibility tools

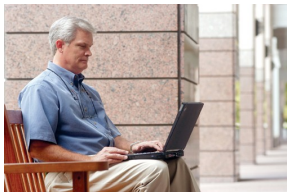
Disability *Assistive technologies can help many people with physiological disabilities*		Example Assistive technologies
Vision	Includes: • people who have a registered disability such as those who are blind, or have limited vision • people who are not registered but still have a visual impairment such as colour blindness	Screen readers Magnification software Braille displays and printers Visual adaptation software (WAT)
Hearing	Includes: • people who have developed audio impairments over time, with some level of hearing loss to those who are now deaf • people who were born deaf and where English is their second language	Captioning software Universal messaging Signing avatars
Dexterity	Includes: • people with a registered disability such as those who have lost limbs, and those with conditions such cerebral palsy and spinal cord injuries • people who may be temporarily disabled, for example people recovering from injuries that affect their ability to use computers	Mouse smoothing software Speech recognition software Eye tracking software Head sticks Sticky keys (OS settings) Alternative mice and keyboards

Inclusive design can help with some cognitive impairments



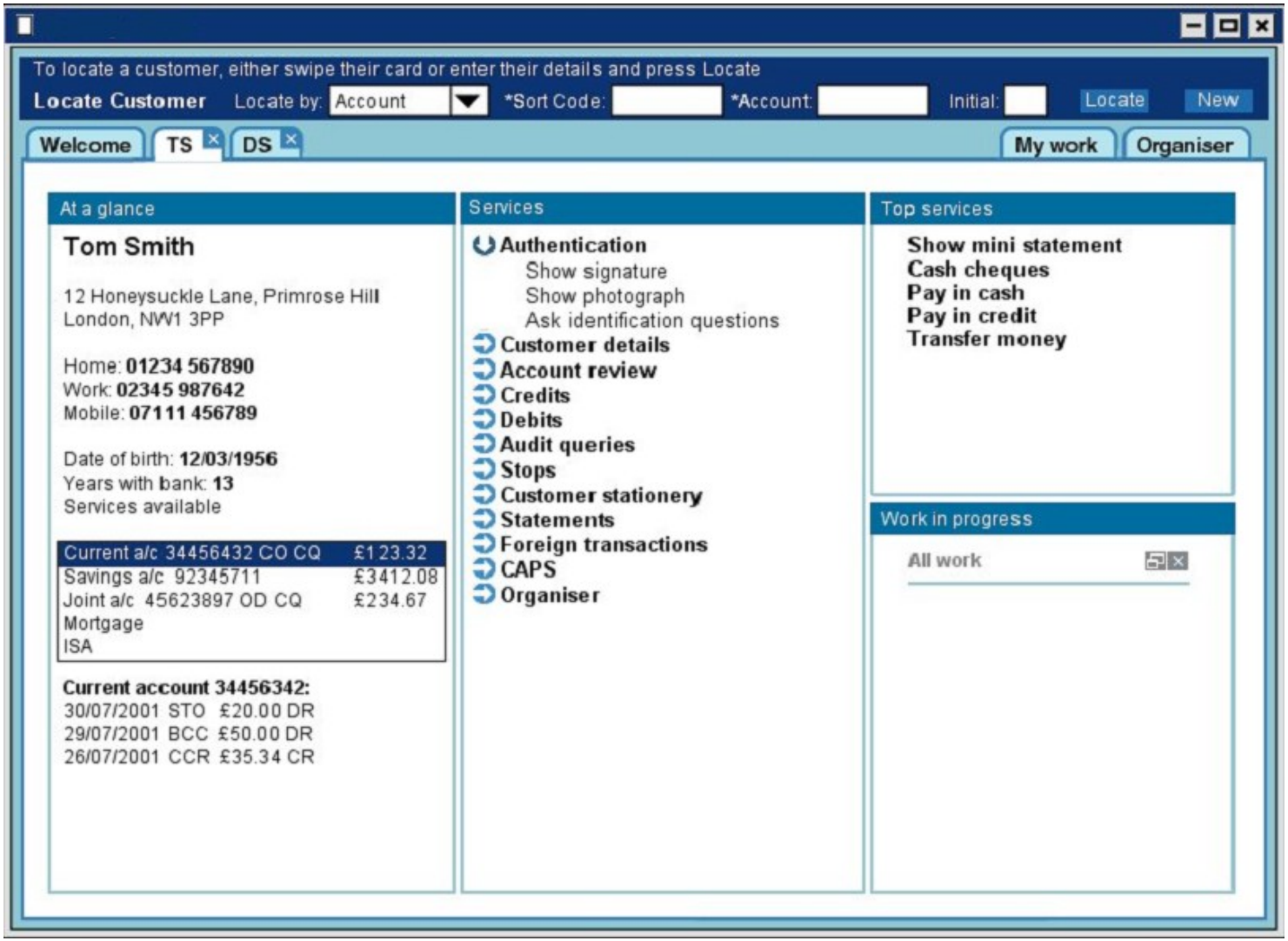
Cognitive impairment		Design approaches
Intelligence *Defined as the ability to solve problems through reasoning and experience*	Includes: • People whose ability to complete tasks is compromised by a lack of understanding and reasoning.	Design for ease of learning, simplified task models, structured and consistent use of concepts and language
Memory *Defined as the ability to encode, store and recall information*	Includes: • People who have difficulty learning new concepts and terminology • People who have difficulty completing tasks that rely on remembering names, objects and processes	Design to reduce memory load, information in context, persistent data, feedback on progress and actions, consistent concepts and language
Attention *Defined as the ability to concentrate on one thing whilst ignoring others*	Includes: • People who have difficulty reading instructions and are distracted when completing tasks resulting in careless mistakes	Design for efficiency and Appeal. Reduce task completion time and increase the use of novel methods to convey familiar concepts. Defensive design.
Perception *Defined as the ability to acquire, interpret, select and organise information*	Includes: • People who have difficulty understanding and interpreting textual, visual or numerical data, for example people with dyslexia and dyscalculia	Designs can be optimized for good information and visual design, symbology and clear writing style (Easy to read)

Inclusive design can help with some adoption issues



Common barriers to technology adoption		Is affected by
Motivation	Where people do not perceive sufficient or indeed any value in the system to invest the effort in learning something new.	**Poor research and communication of user goals and value models**
Confidence	Where people are not confident in their ability to make the right decision or to complete tasks without error. Confidence may be related to a previous bad experience or an inability to accurately remember data required by a system.	Poor information architectures, complex language and task models, **technology mismatch**
Knowledge and learning	Where people do not believe they have sufficient domain or computing experience to use the system effectively. Where people perceive that the system will require an inappropriate amount of time to learn	Unfamiliar concepts, language and metaphors
Trust	Where people may not trust the organization and therefore the services provided by the organization. Issues may include data security, communication ethics, level and quality of service	Poor craftsmanship, communication and writing style
Autonomy	Where people perceive an inappropriate level of control and influence is being exerted by the system	Inflexible interaction styles, mismatch with user's conceptual model
Privacy	Where people perceive an inappropriate intimacy as a result of intrusive questioning or persistent communication.	Conflicting business goals, poor user value communication

An Example Interface from a Large UK Retail Bank



Summary: how do Usability and Accessibility themes impact our requirements, solutions and testing plans?

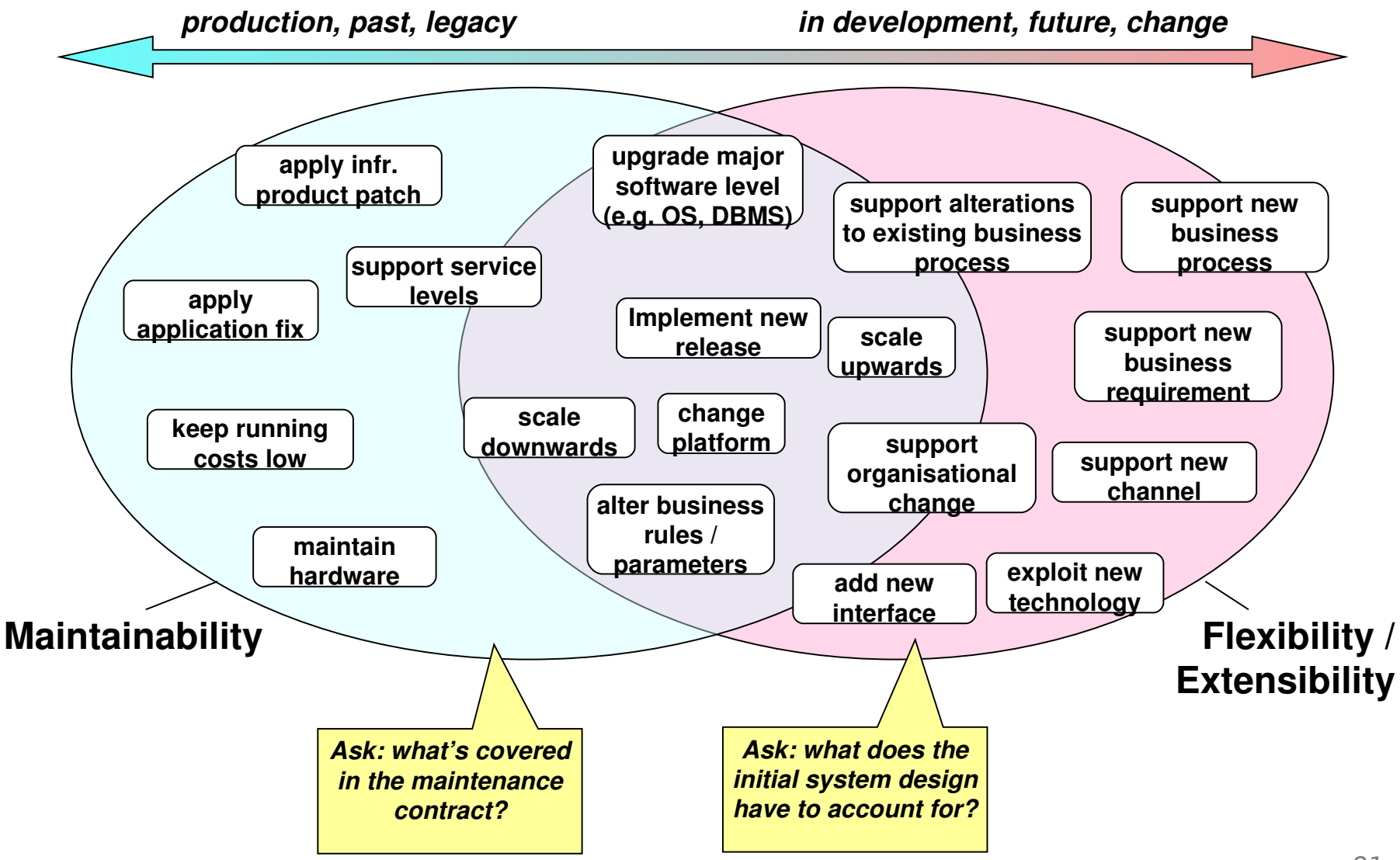
Area	Impact	Examples
Requirements	Include Usability & Accessibility Goals and standards	• "Delivered systems must meet DDA guidelines"
Functional & Content Model	Include components which are required to delivery Usability & Accessibility requirements Design components to meet restrictions implied by requirements	• Transcoding components for different device formats • Limit front end UI to HTML only (no custom applets, etc.)
Operational Model	Infrastructure nodes and deployment design to support accessibility and usability oriented components	• Transcoding node (performance critical) • Client-side deployment of assistive technologies
Implementation & Testing	Ensure additional time is budgeted for to create and test content delivery alternatives Test plans and environment must include appropriate elements	• User acceptance test must include usability & accessibility phase and test cases

Maintainability & Flexibility in IT Systems

Definitions of two related but identifiably different things

- Maintainability:
  - The degree to which a delivered system can be (cost-effectively) maintained in live operations whilst still meeting all business objectives
  - Includes the capacity to apply fixes safely, alter functionality in live, upgrade software, etc.
- Flexibility:
  - The degree to which a system can be changed or extended to meet new or altered business requirements with minimum cost, effort and impact to operations
  - Includes the capacity to change or extend functionality, repurpose for different needs, or scale to different volumes and usage scenarios

Overlap of Maintainability & Flexibility objectives



Method Work Product – the Change Case

IBM Global Services

Change Cases

Work Product Description (WPD)

Unique ID: ARC 118

Version 4.1.1, August 2002

7.Description

The Change Cases work product documents future changes to:

- The system capabilities and properties
- The way the system is used
- The system operating and support environments

Changes are relevant and deserve to be included if they affect the architecture and design now.
This work product clarifies properties of the system described by the phrases, “easy to extend,” “easy to port,” “easy to maintain,” “robust in the face of change,” and “quick to develop.” Change Cases focuses on what is important and likely rather than what is possible. Change Cases try to predict changes, however such predictions may not turn out to be exactly true.

Changes can arise from many sources, for example:

- Business drivers: new and modified business processes and goals
- Technology drivers: adaptation of the system to new platforms, integration with new components
- Changes in the profile of the average user
- Changes in the integration needs with other systems
- Scope changes arising from the migration of functionality from external systems

...
...

Change Case Template

Change Case						
Change Case Name	The future state of affairs or situation that is being considered					
Change Case Subject Area	What area of concern is being addressed for example, platforms, application, users, reuse			**Type of Change**	Modification, Scope Change	
Motivation	Why this is important - what led to its formulation - what goals and expectations are being addressed					
Explanation	A description of the Change Case that expresses the problem situation, the envisioned solution and its effects.					
Probability and Impact Severity						
Time Phase	Which phase: development or after deployment	Probability (high /medium /low or %)	**Impact Severity**	High, Medium, Low	**Provision Date**	When the new capability might be needed
Solution Notes	(Optional) A description of what might be done to respond to the change. This may correspond to what is written in an Architectural Decision.					
Impacted Areas	What the impact of this Change Case will be - what areas of the architecture will be affected?					

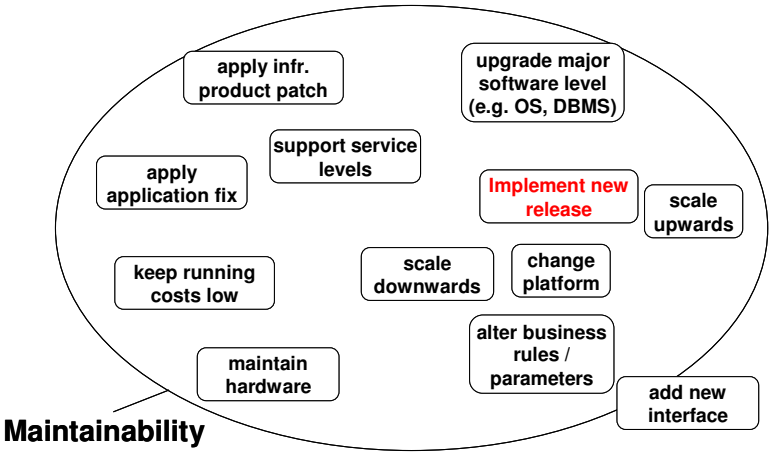
What is the change motivated by?

When is it likely to occur?

What are the implications and impacts of the change?

What will the solution be?

What can the IT Architect do to help those who *maintain* and IT system? Examples ….



- Consider system longevity
  - Select components and technologies with reasonable timespans
  - Avoid highly obscure or specialised components wherever possible
  - Always consider what skills will be required to support the system
- Design for continuous service
  - Minimise events which cause the system to have to be taken down
  - How are application updates to be applied?

- Design in release-to-release update and data update/migration capabilities
- Use performance engineering techniques to design for scalability
- Portability:
  - Select components which are not tied to specific hardware and o/s platforms

- Maximise the types of changes which can be made by users and not by having to recode (e.g. business rules)
  - N.B. Balance this against complexity and other goals, e.g. performance
- Aim for ‘openness’
  - Employ open standards and open technologies to help the system assimilate change in the future

Challenges from the definition of 'Flexibility'

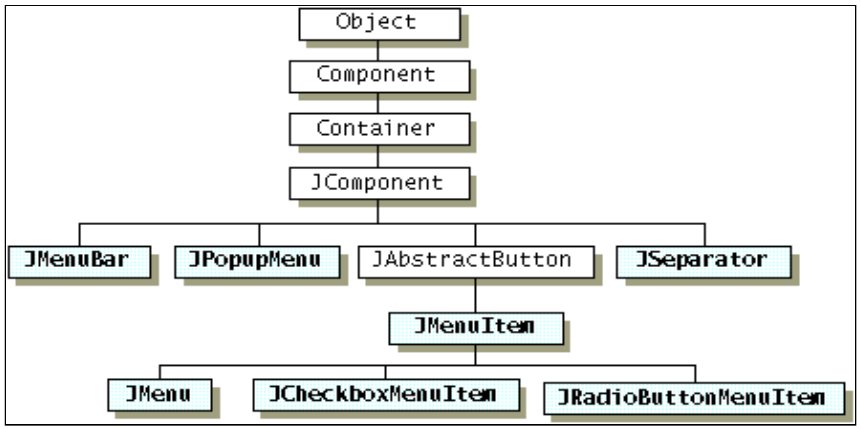
- Flexibility:
  - "The degree to which ..
  - .. a system can be changed or extended ..
  - .. to meet new or altered business requirements ..
  - .. with minimum cost, effort and impact to operations."
- Implications
  - Need to be able to measure flexibility in some way (or at least define "success")
  - Requires change mechanisms, identification of roles, and a extension/reuse framework
  - What is the conceivable scope of changing requirements?
  - Design and infrastructure needs to aim to support change efficiently

Sources of Flexibility & Extensibility constraints

- Architectural & Technical constraints
  - Out of date technology base – cannot be migrated forward
  - Subsystems and components are tightly coupled
    - Can't replace one without replacing the other
  - Functional components not suitable for reuse
    - e.g. wrong level of granularity
  - Business rules hard coded
  - Scalability constraint (e.g. due to logical bottleneck)
  - Skills to modify systems are in low supply
- Constraints not directly caused by system design
  - Business organisation and processes are not flexible
  - No overall Enterprise Architecture or architectural governance
    - replicated functions and data
    - low degree of commonality
  - Client is not prepared to pay for flexibility during solution design and implementation
  - Impossible to see direction of change (! / ?)

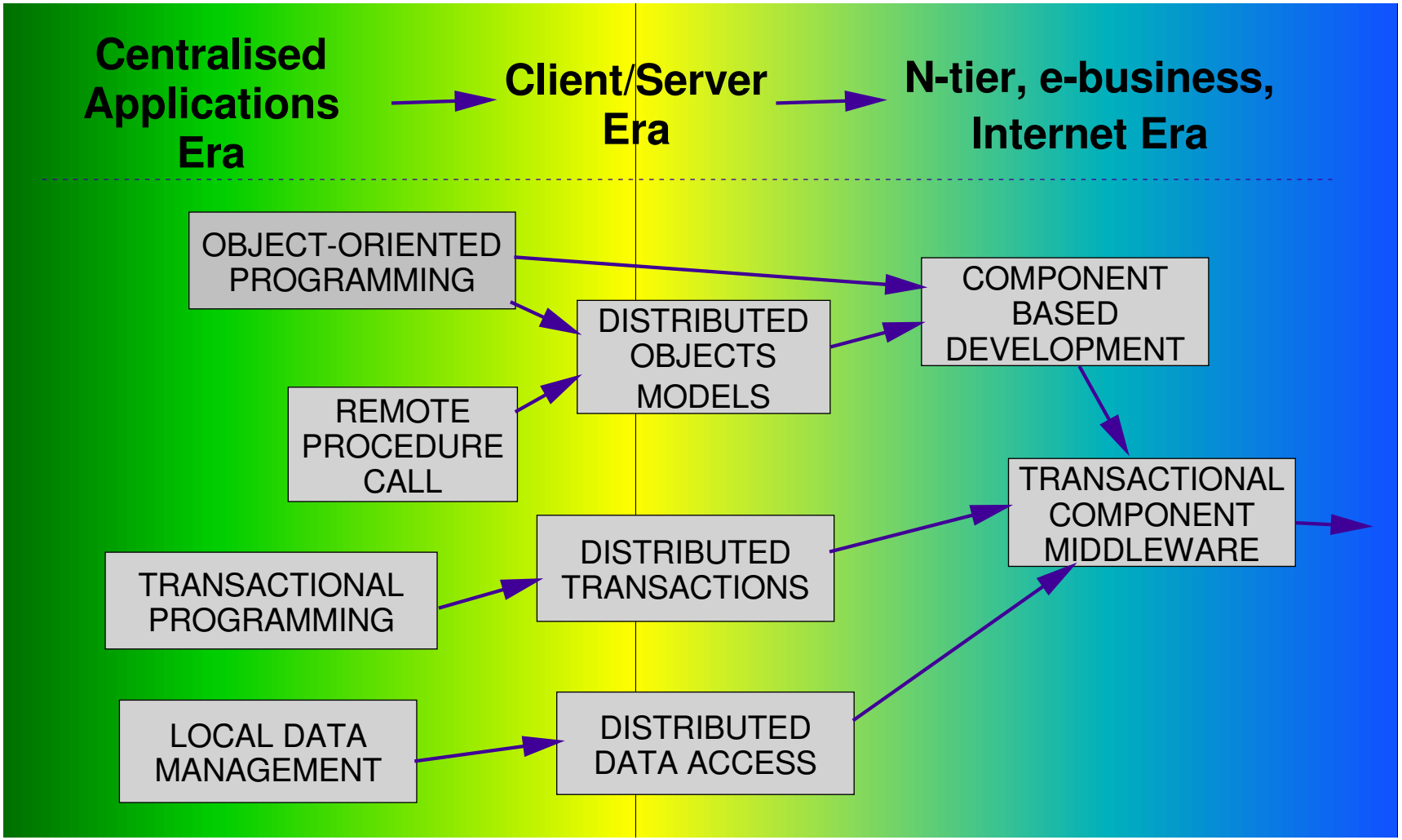
Object Orientated Programming – the original (?) solution to "reuse" and flexibility

- Object Orientation "theory" says:
  - Encapsulation and Inheritance Ö
  - OO speeds development
  - OO permits intuitive modelling of the real world
  - OO aids extension and maintenance of an application
  - OO aids reuse
  - etc. *(yawn)*

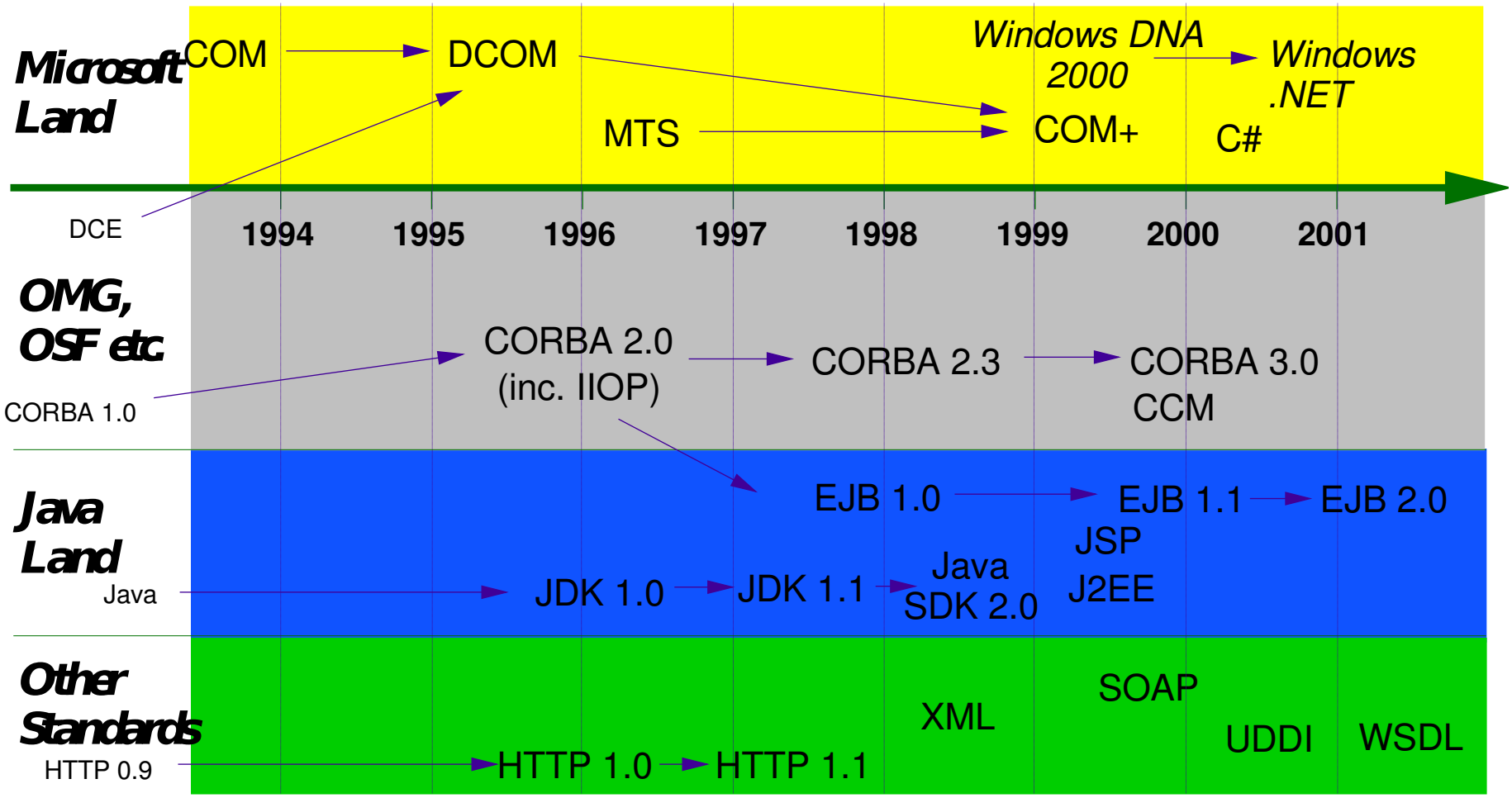


- Why did OO not "solve" the flexibility problem?
  - OO implementations tend to be associated with specific languages and runtimes
    - you are stuck with the one you choose
  - Real world things don't fit into hierarchical taxonomies
  - Fine-grained objects
  - + practical issues:
    - Ability to design & build rapidly can mean s/w engineering disciplines are shortcut
    - Performance issues related to distributed deployment and data access

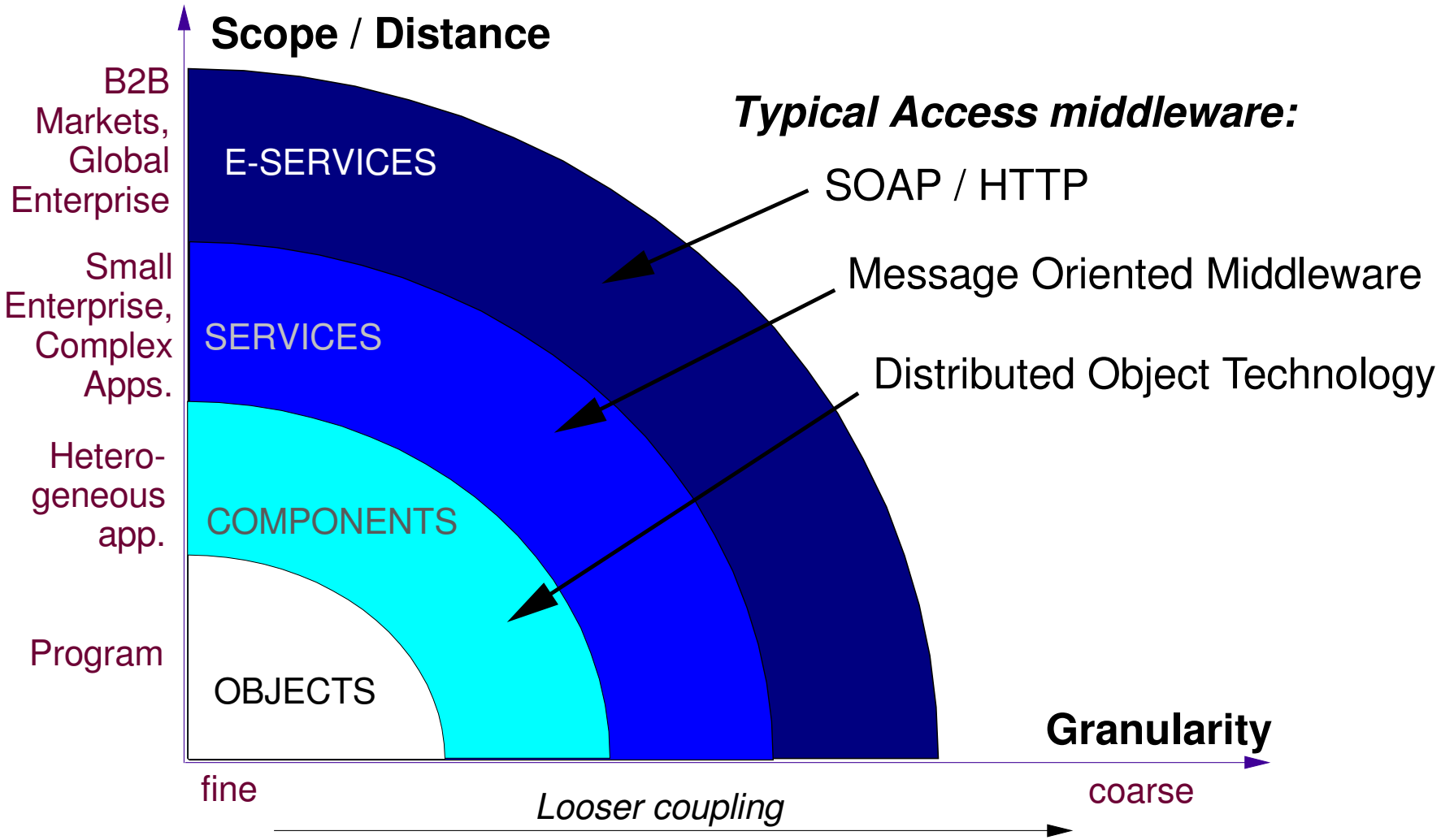
Sidestep 1: Evolution of Middleware



Sidestep 2: A timeline of Distributed Applications Technologies (1994 – 2001)



Application coupling – Gartner view



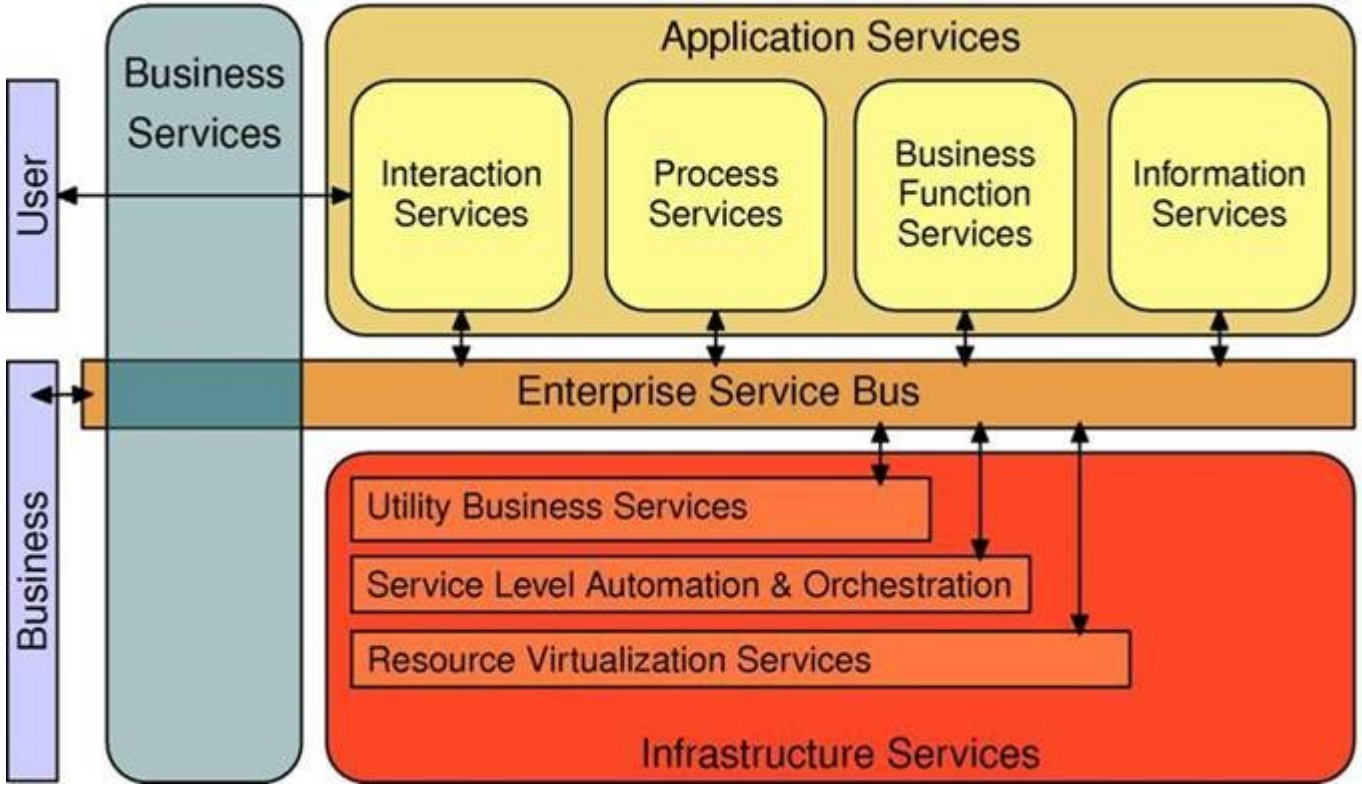
Three design flexibility watchwords to dance by

- Objectives in flexible system design

- Loose coupling (*arms out!*)
  - Meaning components are not tightly bound together (either logically or technically), giving freedom to alter component internals and implementations
  - The ‘interface’ or ‘service definition’ needs to stay the same in order to have zero impact on other components
- High cohesion (*elbows together!*)
  - Despite being loosely coupled, we still want components to ‘fit’ and work well together
  - The component model must still ‘make sense’, be logical
- Encapsulation (*arms above your head!*)
  - Components encapsulate (‘contain’, ‘capture’, ‘own’) a logical and consistent piece of functionality and/or data

(Semi-reprise from WDITADAD?) "Service Oriented Architecture"

- Shared logic & data, common services



What's really new?:

- service definition technology independent (excepting XML!)

- possible (if not always desirable) to perform runtime binding

The 'Buy' vs. 'Build' vs. 'Construct' debate

Strategy	Benefits (theoretical)	Implications and risks
Custom application development	• Applications can be built to meet exact requirements • Retain control of all technical standards, products and overall architecture • Flexibility is as good as your architecture	• Need to be able to capture requirements and develop efficiently • Require significant body of in-house or contracted skilled resource • Requires strong governance
Packages	• Exploit 'best of breed' functionality • Quicker / lower risk to implement (N.B. may be expensive to maintain …) • Fewer in-house skills required	• Must accept vendor 'view of the world' (e.g. data model, business process) • Need to integrate packages together • Flexibility dependent on vendor's architecture • Can become reliant on vendor
Frameworks & toolkits	• Construct applications flexibly from frameworks to achieve high flexibility • Potentially lower cost and risk then custom application development	• Still reliance on vendor • Flexibility limited by scope of vision of the framework / toolkit • More complicated than straight package implementation

Trends to watch in flexible business application construction

- Model Driven Architecture (MDA) and Model Driven Development (MDD)
  - Construct executable artefacts from design models
  - Create or alter behaviour by managing the models
  - Can include population through discovery of existing infrastructure and services
  - Particularly useful for integration purposes and ‘brownfield’ developments
- Business workflow and business rules oriented development
  - Specify the application functionality as a business workflow governed by rules
  - Business workflow and supporting rules are designed and specified by business people, not developers
  - Workflows and rules can be modified, simulated and applied in real time
  - A growing market

Summary

Summary of Topics

- Despite continuing advances in technology, IT Architects spend significant amounts of time engineering IT systems to account for Qualities and Constraints
  - Software and infrastructure designs need to be iterated together to achieve goals
- IT systems increasingly go hand in glove with business processes and business policy
- Security is a vital characteristic of any IT system managing valuable assets
- Customer requirements often include vague and difficult to measure statements such as “easy to extend”, “supports future business change”, “easy to maintain”
- IT Architects need to consider all of the following for each design challenge:
  - Motivation – Requirements – Technologies & Tools – Methods and Techniques – Architecture & Design – Implementation – Management and Change
- Regardless of the quality of design, the quality of implementation must be validated through testing
  - Architects must influence test strategy and planning

**** May your systems be secure,**
easy to use, and flexible in the face of change **