Dr. Hans-Peter Hoidn
Distinguished IT Architect (OpenGroup certified)

Enterprise IT Architectures

Key Topics of an Solution Architecture Some Aspects in more Depth

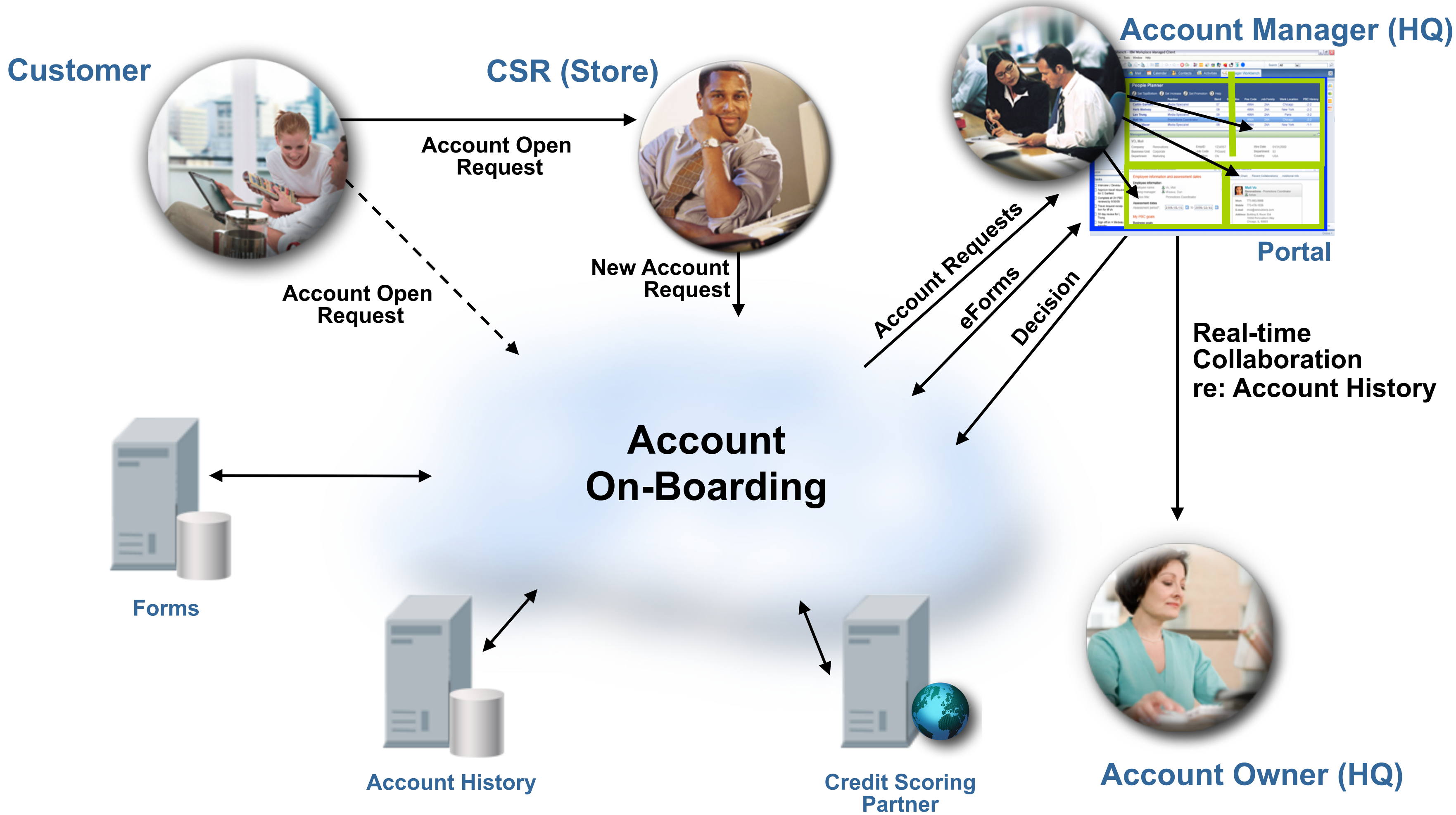
Agenda of this Session

- Q&A and Recap for Term Paper
  - Work Product (Scope, Decisions, Overview, Layered View)
  - Requirements (underpinning a statement of work)
- Security In Depth
  - Authentication and Authorization, Attacks
  - Operational Model with Zone concepts
  - Scenarios
- Enterprise Architecture (optional – will be continued December 7)
  - CBM (Component Business Model)

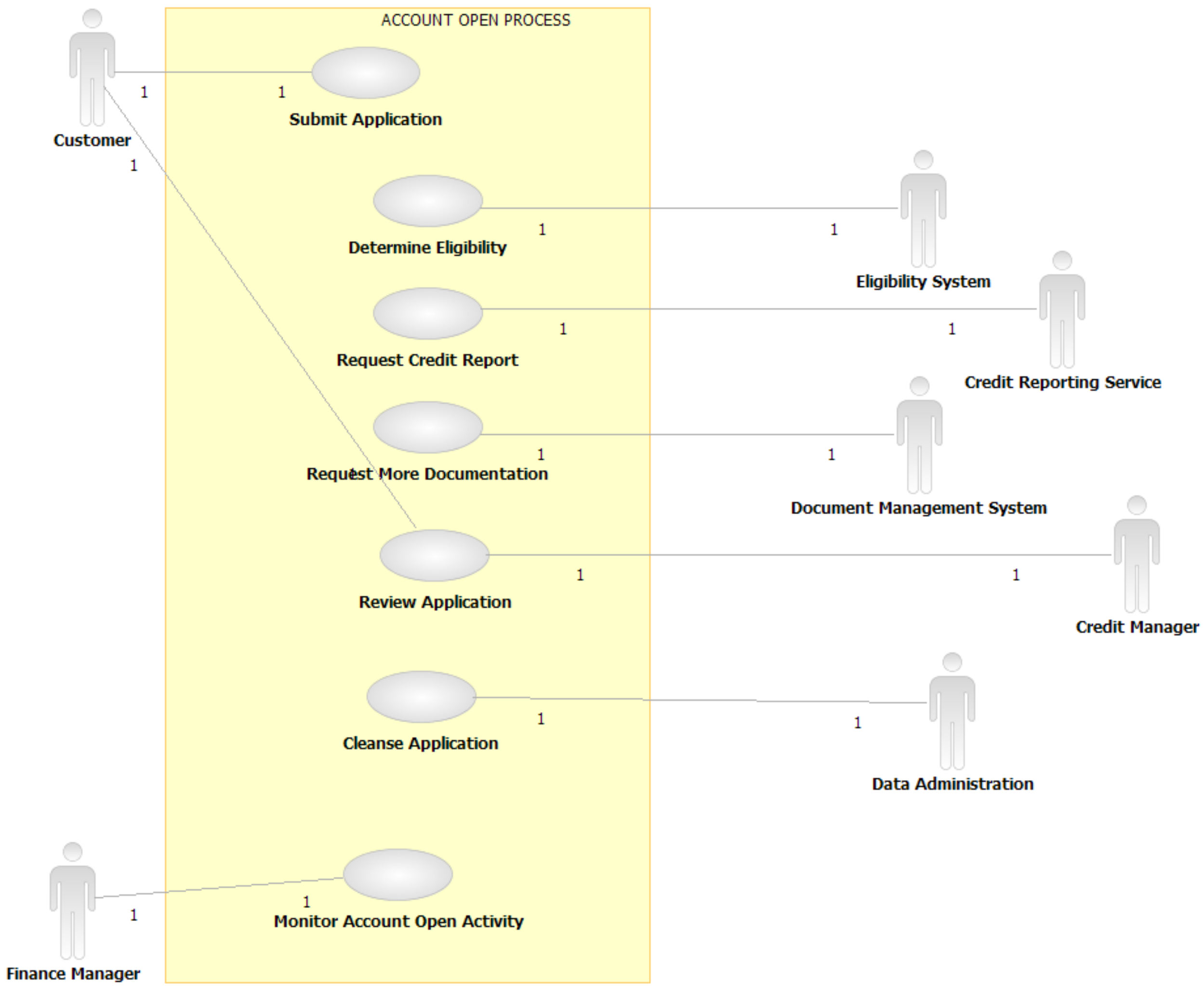

Questions

Q&A – Recap for Term Paper
Important Aspects of a Proposal

Business Context Diagram makes clear what is in scope and what is outside



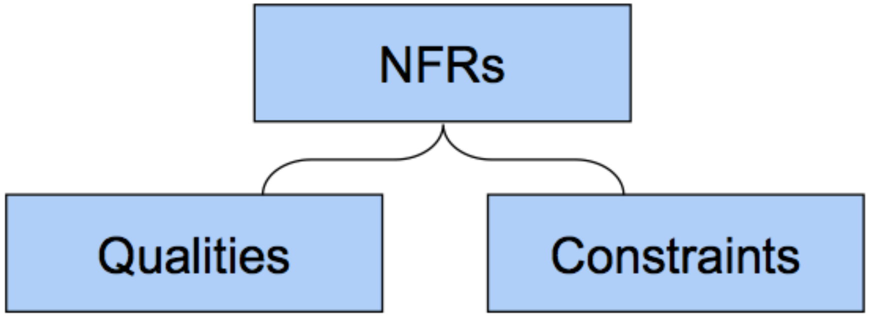
Use Case Diagrams show the Functional Requirements



NFRs (Non-Functional Requirements)

Non-functional requirements (or NFRs) define the desirable qualities of a system *and* the constraints within which the system must be built

- *Qualities* define the properties and characteristics which the delivered system should demonstrate
- *Constraints* are the limitations, standards and environmental factors which must be taken into account in the solution



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

Business	Technical
Regulatory	Legacy Integration
Organisational	Development Skills
Risk Willingness	Existing Infrastructure
Marketplace factors	Technology State of the art
Schedule & Budget	IT Standards

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.

Run-time

- Performance & Capacity
- Availability
- Manageability
- Security
- Usability
- Data Integrity

Non-Runtime

- Portability
- Reliability
- Efficiency
- Scalability
- Maintainability

Quality ! – Good versus Bad Qualities

Well specified are:

- *Correct*
- *Unambiguous*
- *Complete*
- *Consistent*
- *Measurable (verifiable)*
- *Traceable*
- *Actionable*
- *Design independent*

Note on feasibility:

- *It may not be possible to meet a particular NFR given other constraints – if so, this is a design/business issue*

They are *not* well specified if they are:

- *Misrepresentative of the true business need*
- *Open to interpretation*
- *High-level "principles" or "guidelines"*
- *Conflicting*
- *Not possible to test*
- *Implying a specific solution or technology*
- *Missing !*

Quality ! – Reducing Risk by consider the qualities from start

Build 'quality' into the solution starting with early design

- Understand the risks to the project
- Conduct quality of service engineering from the first elaboration of the architecture model
- Set guidelines for the developers (software & infrastructure)
- Test the application/system at each major stage of development
- Make sure that the live support teams will be able to manage quality

Fix it early, and save money and problems later …

Common Problems with Non-Functional "Requirements" (1)

Requirements are often vague or unactionable

- They need further elaboration, clarification, investigation (and possibly rejection)
- It may be possible to derive clear, actionable intentions from them

Requirements can be statements of principle or good intention but come with little enforcement

- The organisation's governance models are central

Once captured, requirements are often treated as "musts" or "givens" whereas in fact they are "tradable" and may need to be challenged

- Classic example is "given" technology standards (e.g. "all applications in .NET") or infrastructure constraints ("64kbps links to offices")

Common Problems with Non-Functional “Requirements” (2)

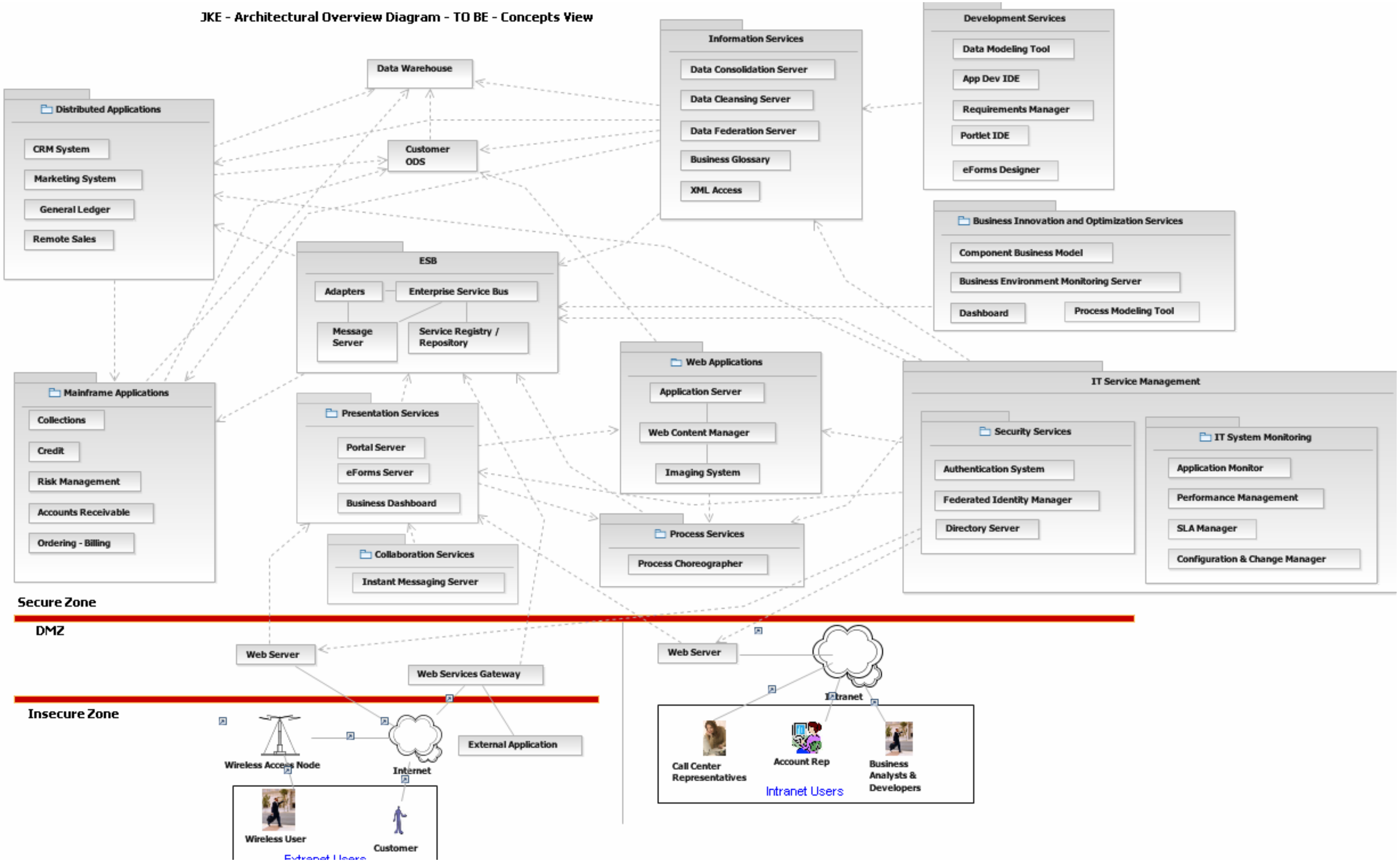
Requirements are often of poor quality

- Watch out for these issues:
  Unrepresentative, unclear, inaccurate, inconsistent, incomplete or unnecessarily constraining

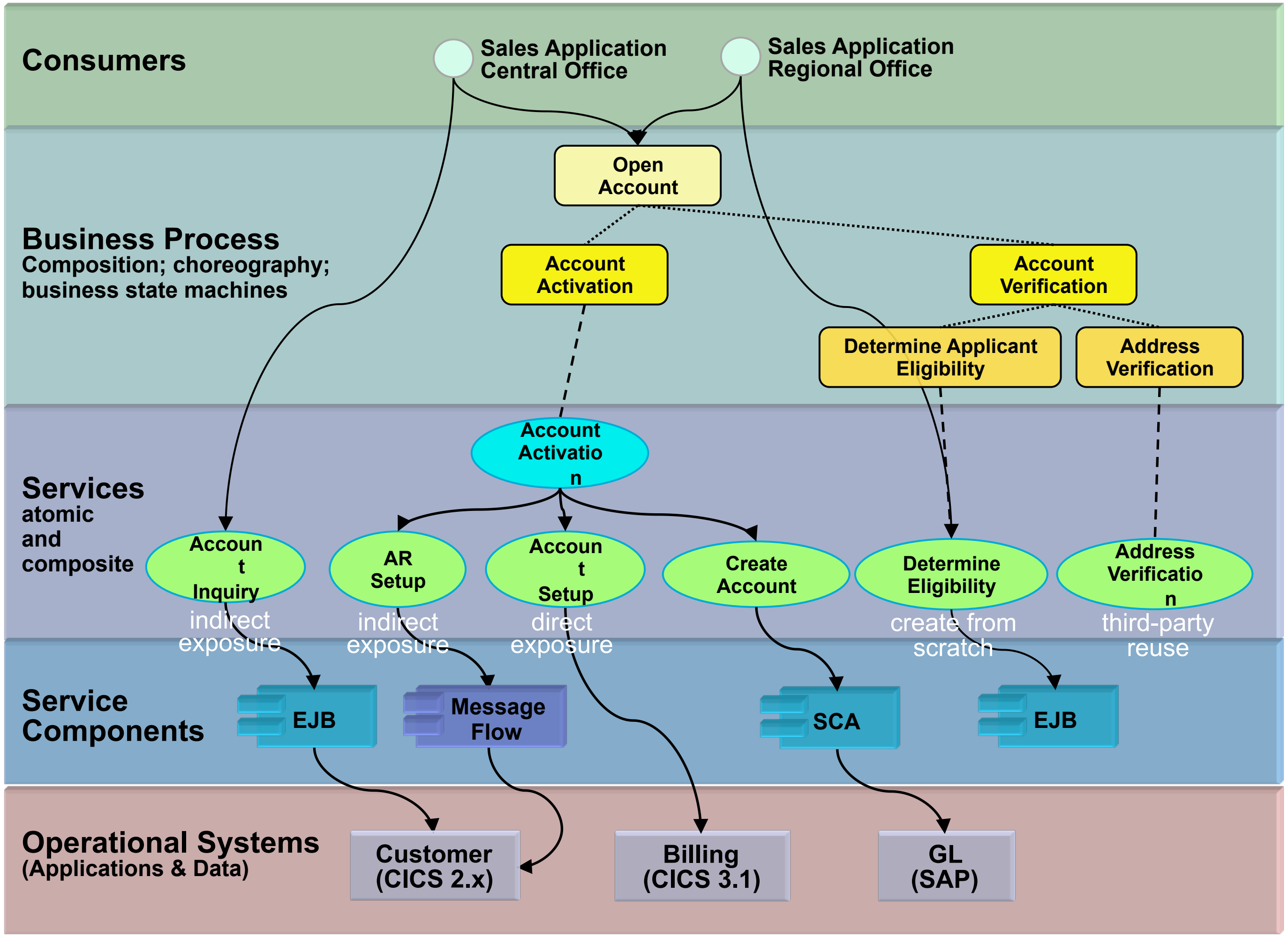
NFRs documents often become “dumping grounds” for things which don’t have another home

- (regardless of quality or suitability)

To-Be AOD (Architecture Overview Diagram) – Providing Planning Assumptions



The SOA Layered View illustrates the Presentation, Business Processes, Services and their Components



In more Depth: Security

Security is an example of run-time quality and covers these topics

Safety

- To reduce or eliminate danger
- To reduce or eliminate anxiety
- To reduce or eliminate risk or liability

Protection

- To defend against attacks (insider and outsider)
- To defend against fraud (misuse of assets or misrepresentation of identity)
- To defend tangible assets (IT systems and applications or stored information or information in transit)
- To defend intangible assets (reputation)

Assurance

- To ensure correct and reliable operation
- To enforce identity and ownership
- To promote trust

Broad scope of a "Security Governance, Risk Management, and Compliance"

- People and Identity (e.g. Authentication, Access Control)
- Data and Information (e.g. Cryptography, Data Loss Protection)
- Application and Processes (e.g. Security of Services)
- Network, Server, and Endpoints (e.g. Virtualization, Threat Protection, Malicious Traffic Detection, "Circle of Trust")
- Physical Infrastructure (e.g. Operational Management)

Security & Safety

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

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

Key Objectives of Security Engineering (1)

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)

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.

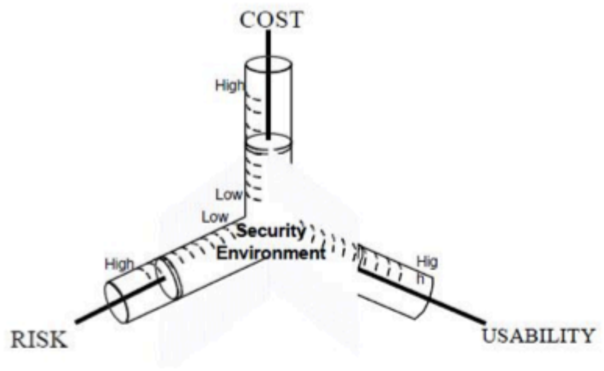
Non-Repudiation – proving what happened happened

- 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 Architecture is about answering the question "how much security is enough security"

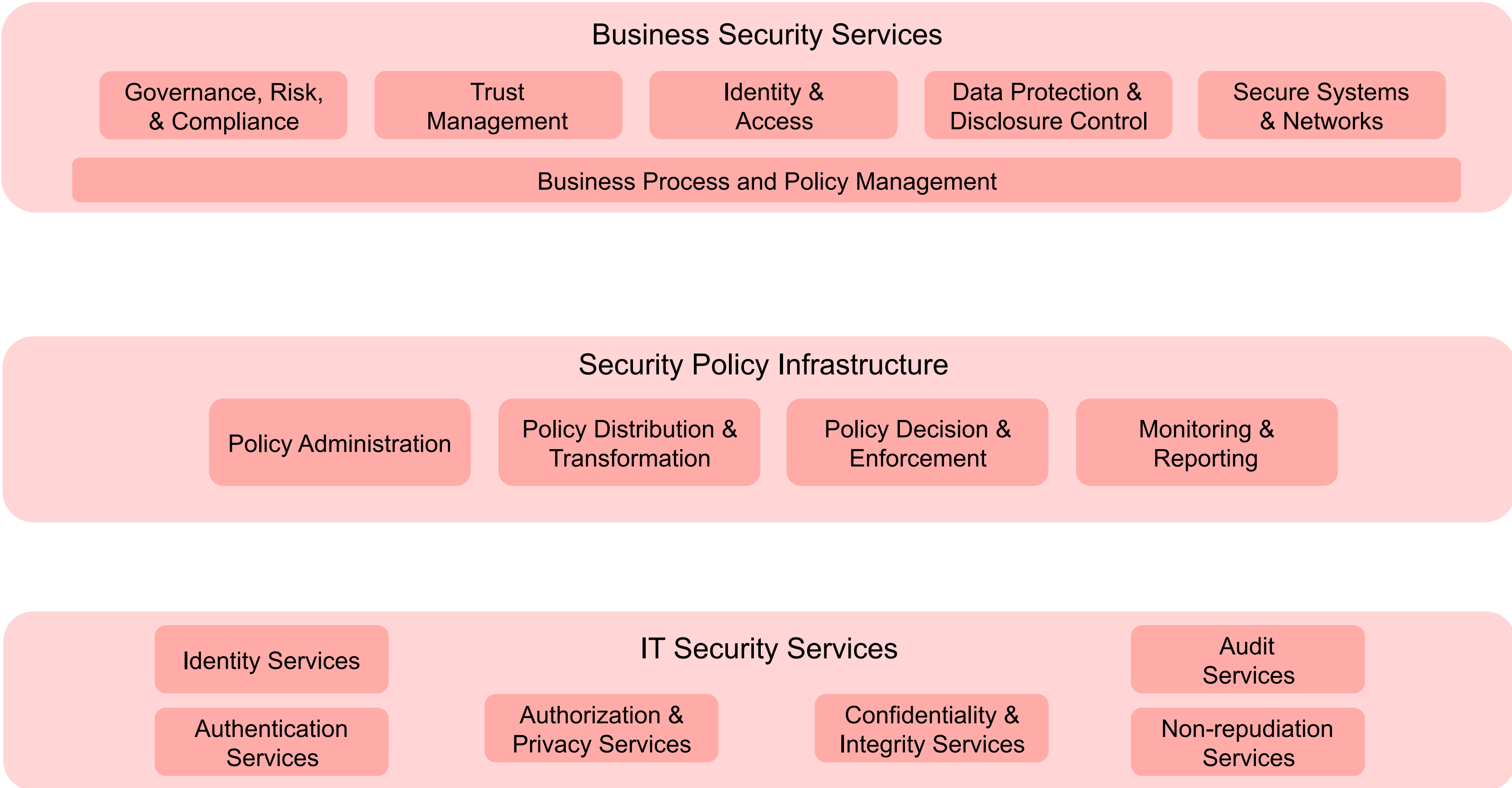
From a security perspective, all IT solutions must balance three conflicting factors:

- **The risk** – to the organisation of operating the IT solution
- **The cost** – of implementing and operating the security controls in general, the tighter the controls the lower the risk
- **The usability** – of the solution in general, the tighter the controls, the greater the impact on the users of the system



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

Security Reference Model (for SOA by IBM)



People and Identity: Today's *Identity and Access* Challenges

Trusting Identities

Customers
or criminals?

Partners or
competitors?

Employees
or hackers?

Managing Access

Securing Services

Payroll

Online banking

Loan applications

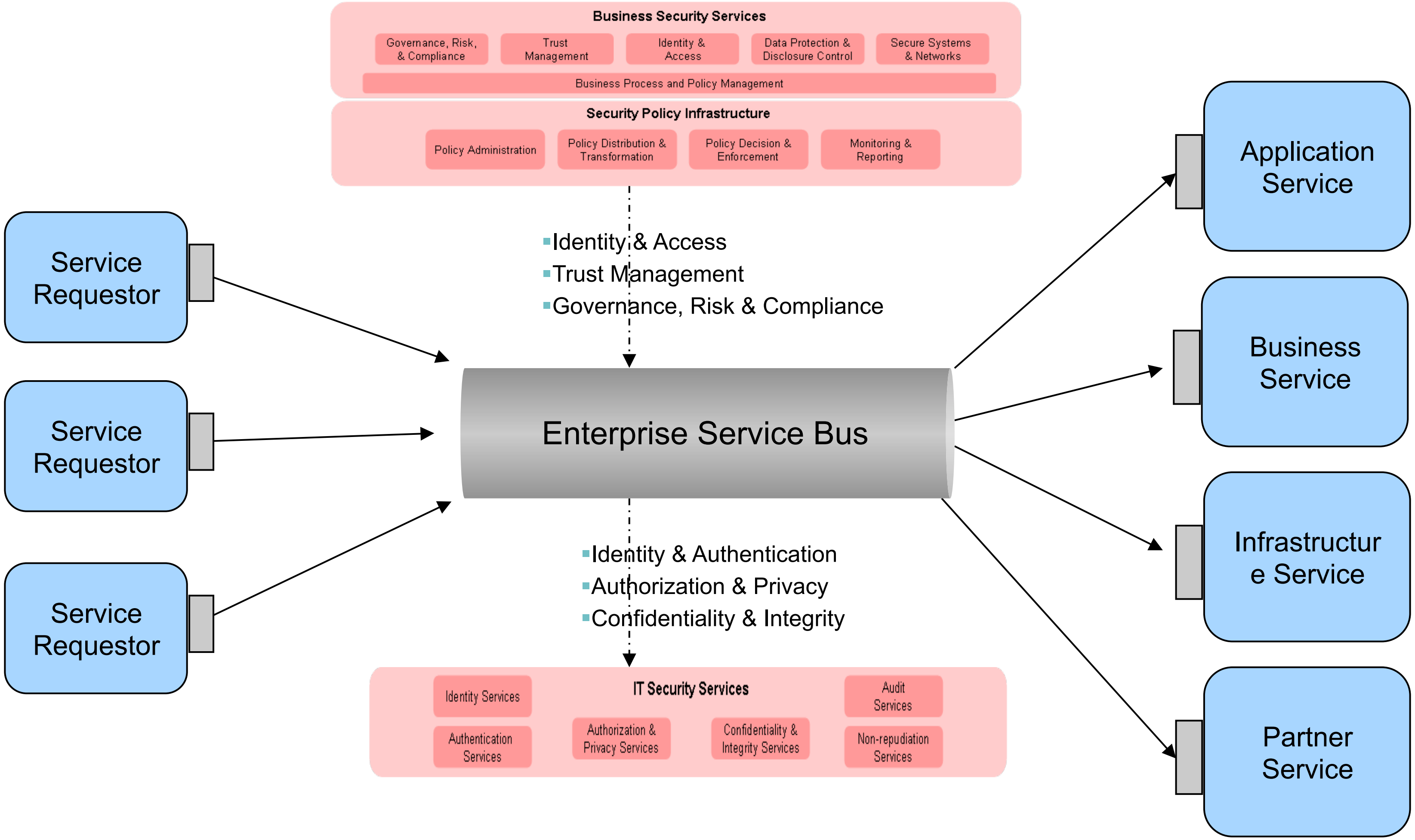
Retail sales

Inventory

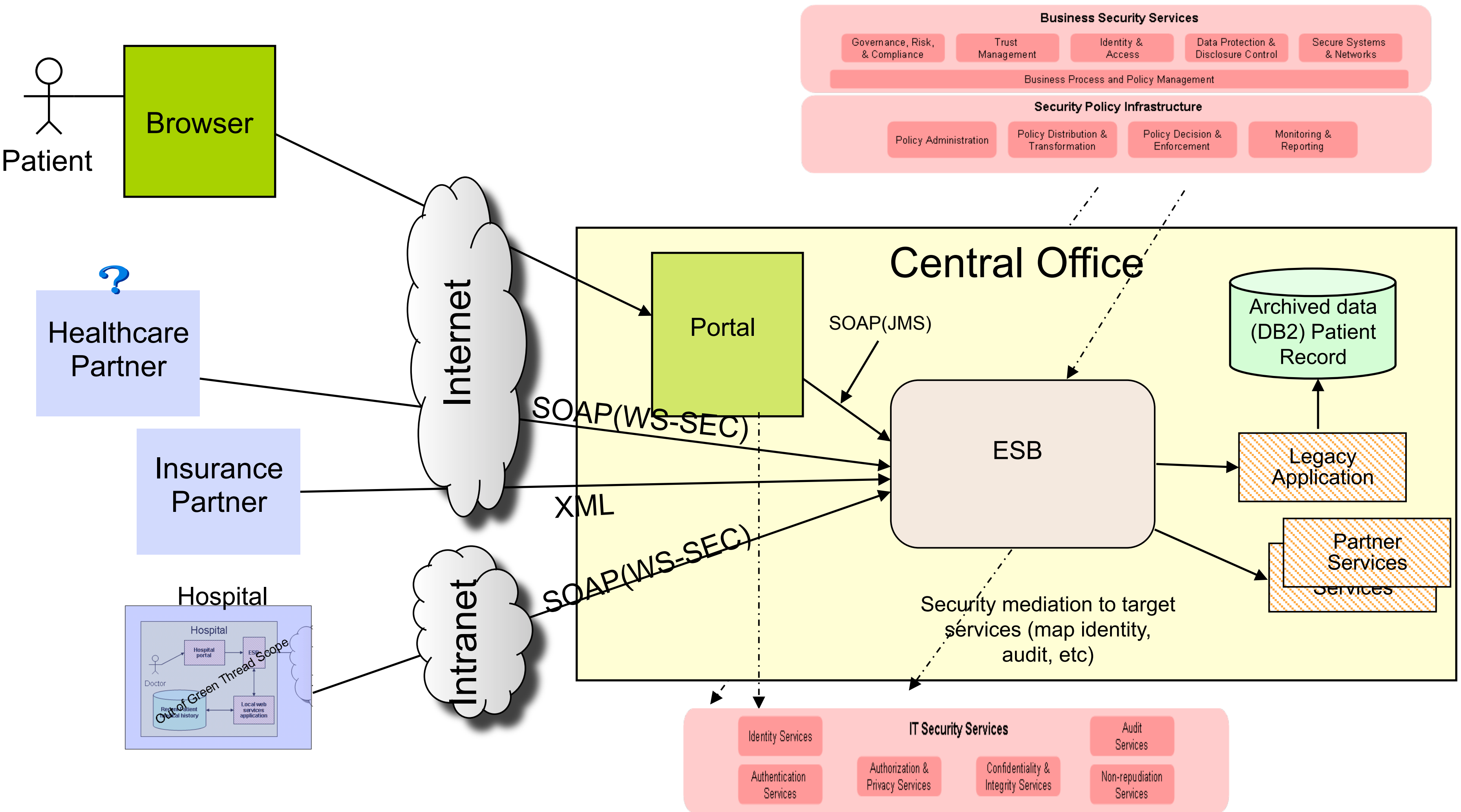
Protecting Data and Privacy

Security has to be applied within a Business context and fused into the fabric of business and not as a widget to solve the next security threat

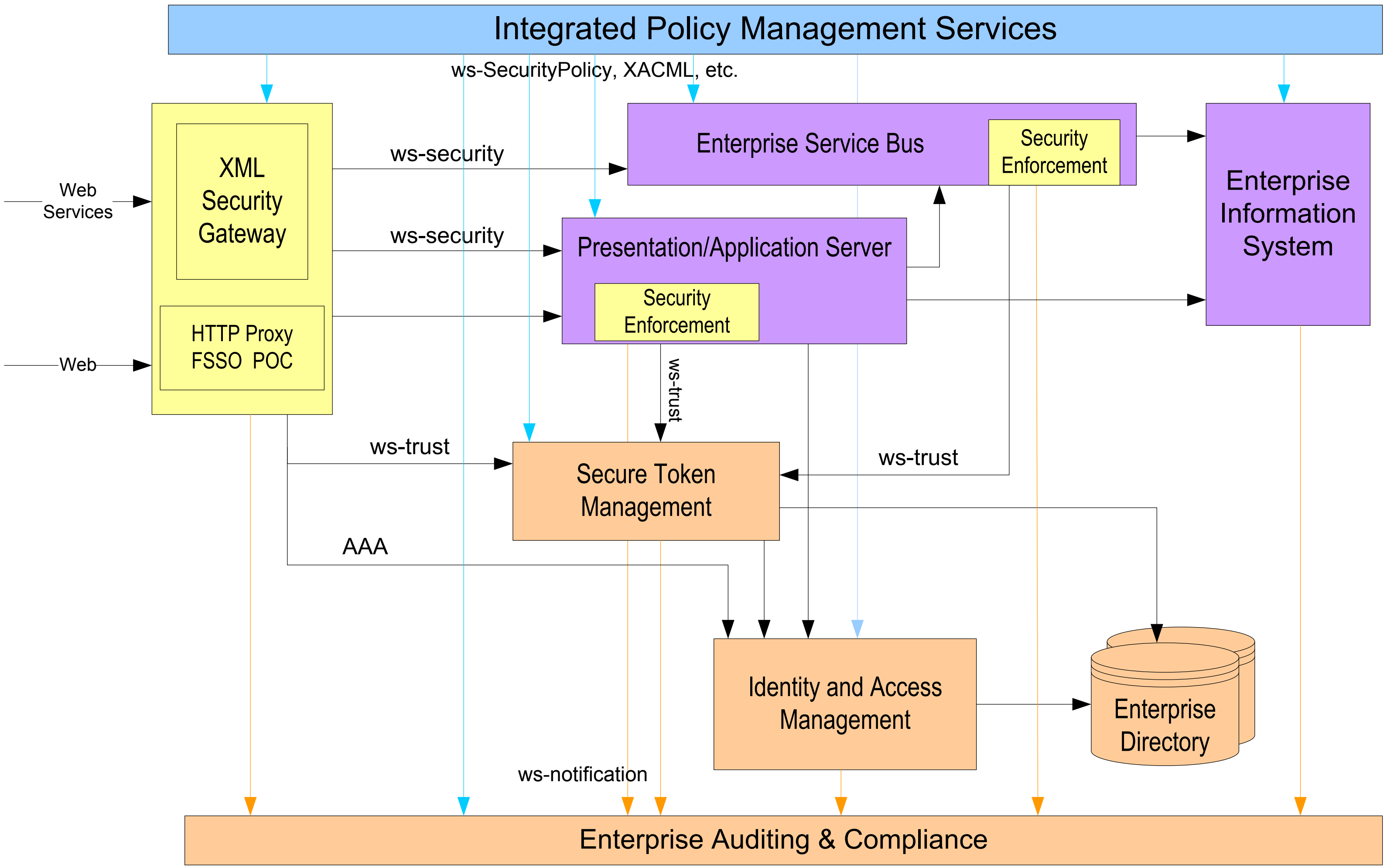
Services Integration Scenario



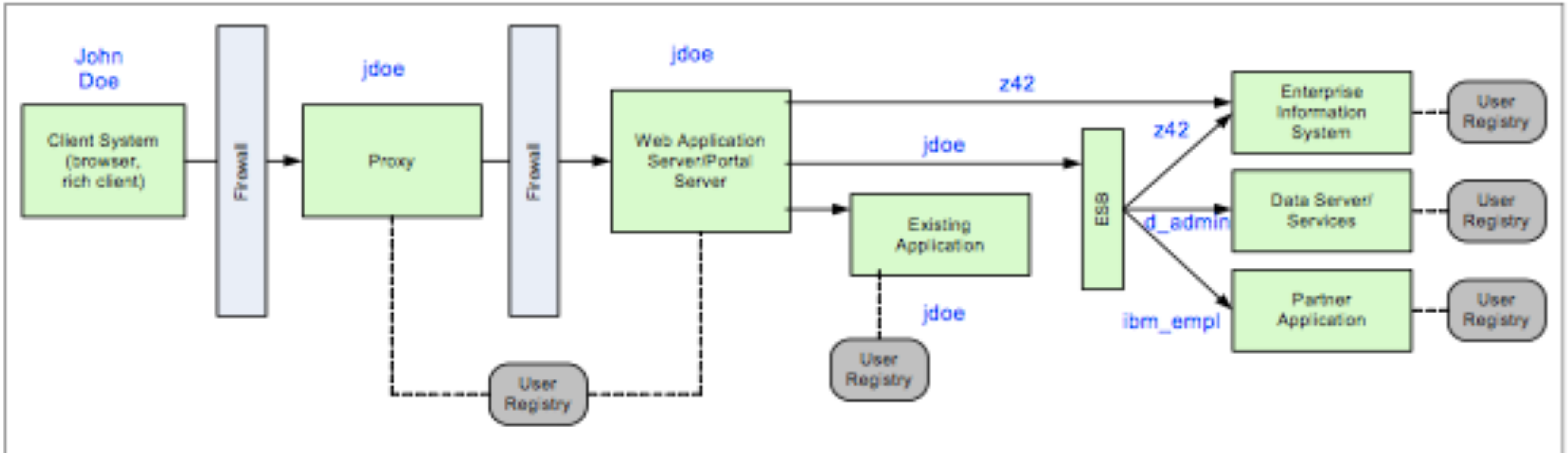
Service Integration Scenario with Security



Logical Architecture for Service Integration Scenario



Identity and Access: Various “Identities” for one person



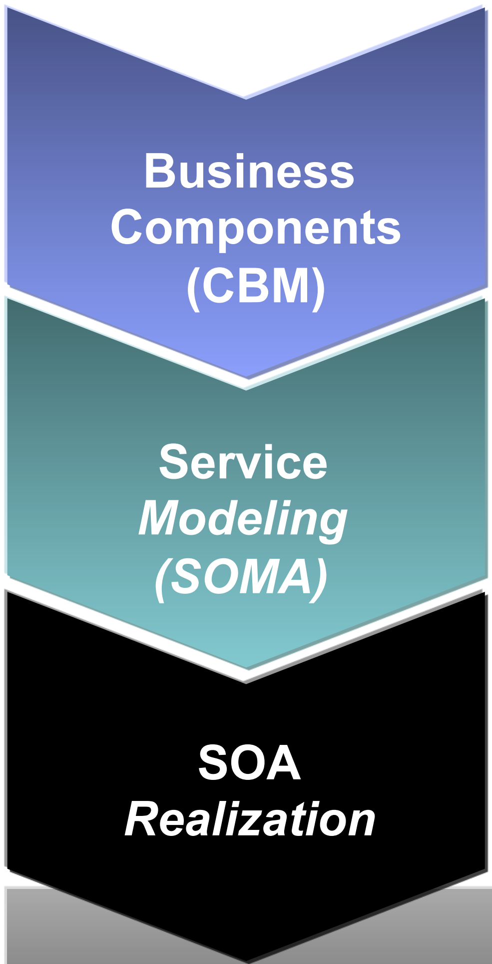
- John Doe with multiple “identities” and roles
- Applications with multiple user registries
- Challenges: Transferring identities, aligning registries, areas of trust
- Reference: http://www.redbooks.ibm.com/abstracts/sg247310.html?Open

Appendix: Security Services – Standards

Service	Relevant Standards
Identity Services	IdAS, SPML, SAML, WS-Federation
Authentication Service	WS-Trust, Kerberos, SAML, PKI
Authorization and Privacy Services	XACML, JACC, WS-Authorization, WS-Privacy, WS-Policy, IDEMIX
Audit Service	CBE extensions, Audit web service (in progress), WS-BaseNotification
Message Protection	WS-Security, WS-SecureConversation, PKI, XKMS, WS-SecurityPolicy, SSL/TLS, JSSE/JCE

Business View with CBM (Enterprise Architecture)

Approach for SOA



Step 1: Break down your business into components

- Decide what is strategically important, and what is just operations in the value chain domains
- Analyze the different KPIs attached to these components
- Prioritize and scope your transformation projects

Step 2: Define a Service Model

- Identify your services based on your business components
- Specify the services and components accordingly
- *Make SOA realization decisions based on architectural decisions*

Step 3: Implement a Service Model

- Develop a service-oriented architecture to support the Componentized Business
- Implement service based scoping policy for projects
- Implement appropriate governance mechanism

Business-Aligned IT Architecture

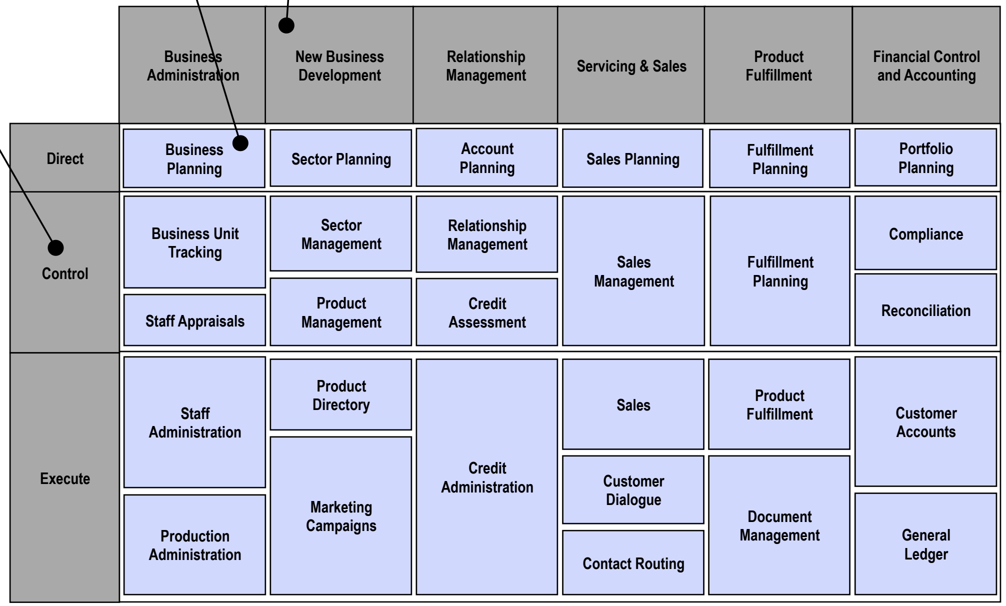
Component Business Model (CBM) – Definition (1)

A **Business Component** is a part of an enterprise that has the potential to operate autonomously, for example, as a separate company, or as part of another company.

Columns are Business Competencies, defined as large business areas with characteristic skills and capabilities, for example, product development or supply chain.

An **Operational Level** characterizes the scope of decision making. The three levels used in CBM are direct, control and execute.

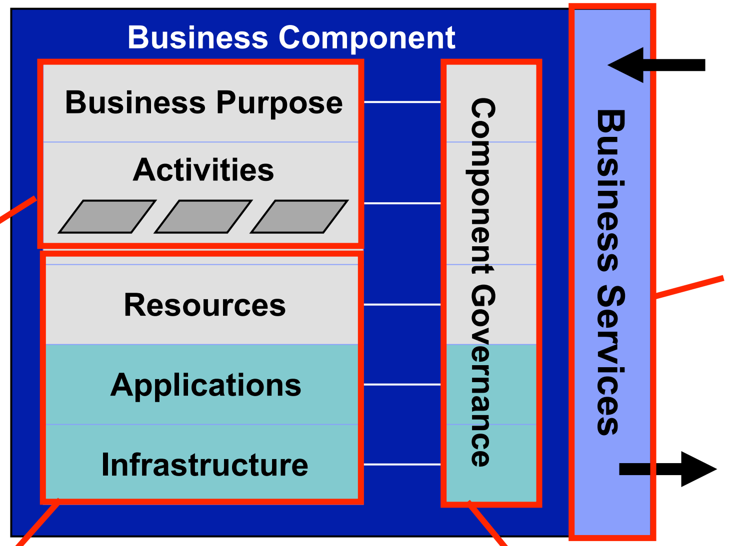
- Direct is about strategy, overall direction and policy.
- Control is about monitoring, managing exceptions and tactical decision making
- Execute is about doing the work



CBM – Definition (2): The building block of a component business model is a ‘business component’

A component is a business in microcosm. It has activities, resources, applications, infrastructure. It has a governance model. It provides goods and services (business services)

Business Component Elements



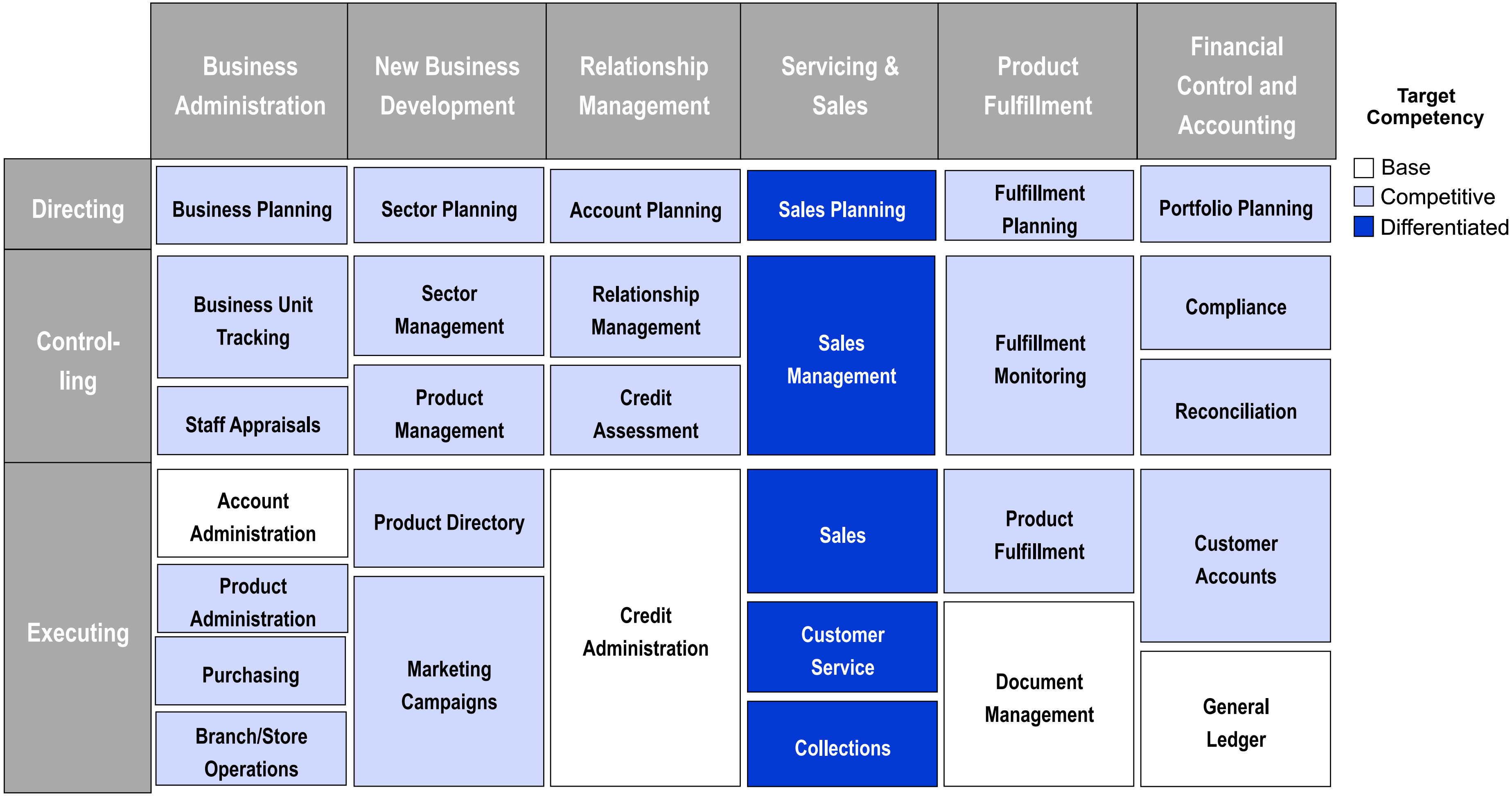
Each business component has differentiated capabilities

Each business component defines and decides on the use of all resources needed to perform the defined activities

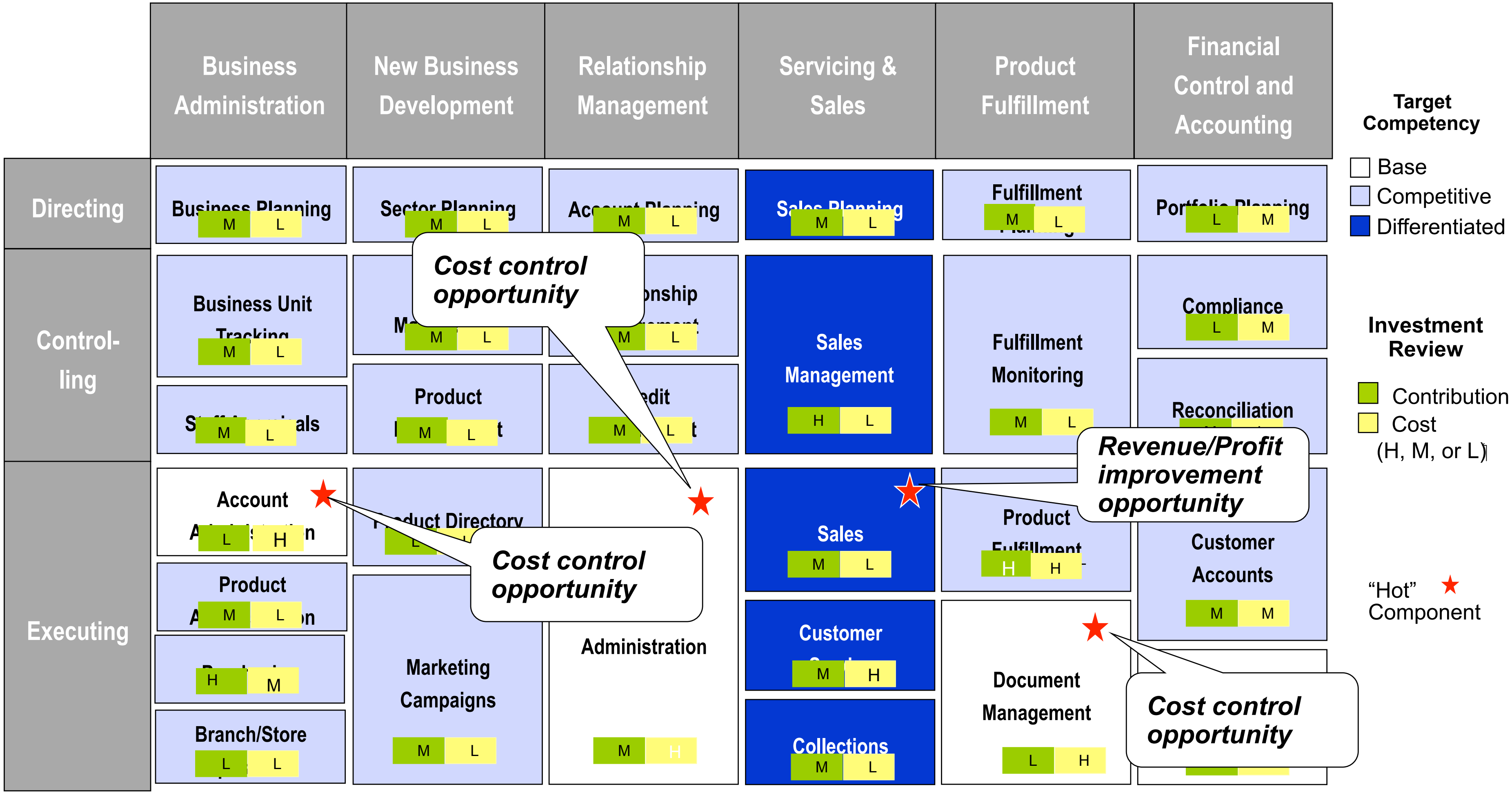
Each business component has a governance structure within which it manages its activities

Each business component has business services which form the interfaces to other business components

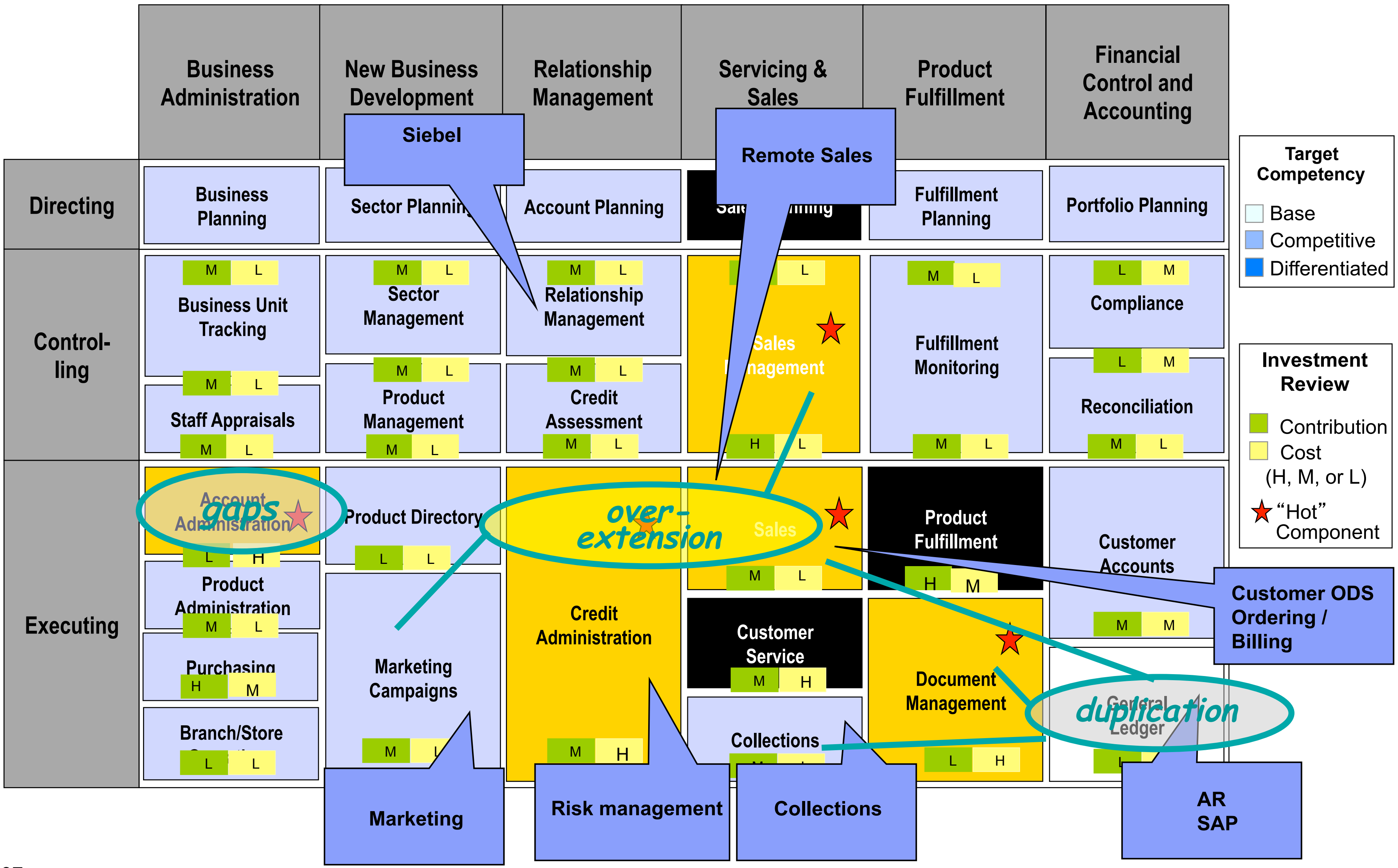
Domain Decomposition– Component Business Modeling for JKE



Domain Decomposition– Component Business Modeling for JKE



CBM and IT Systems Coverage for JKE




Questions