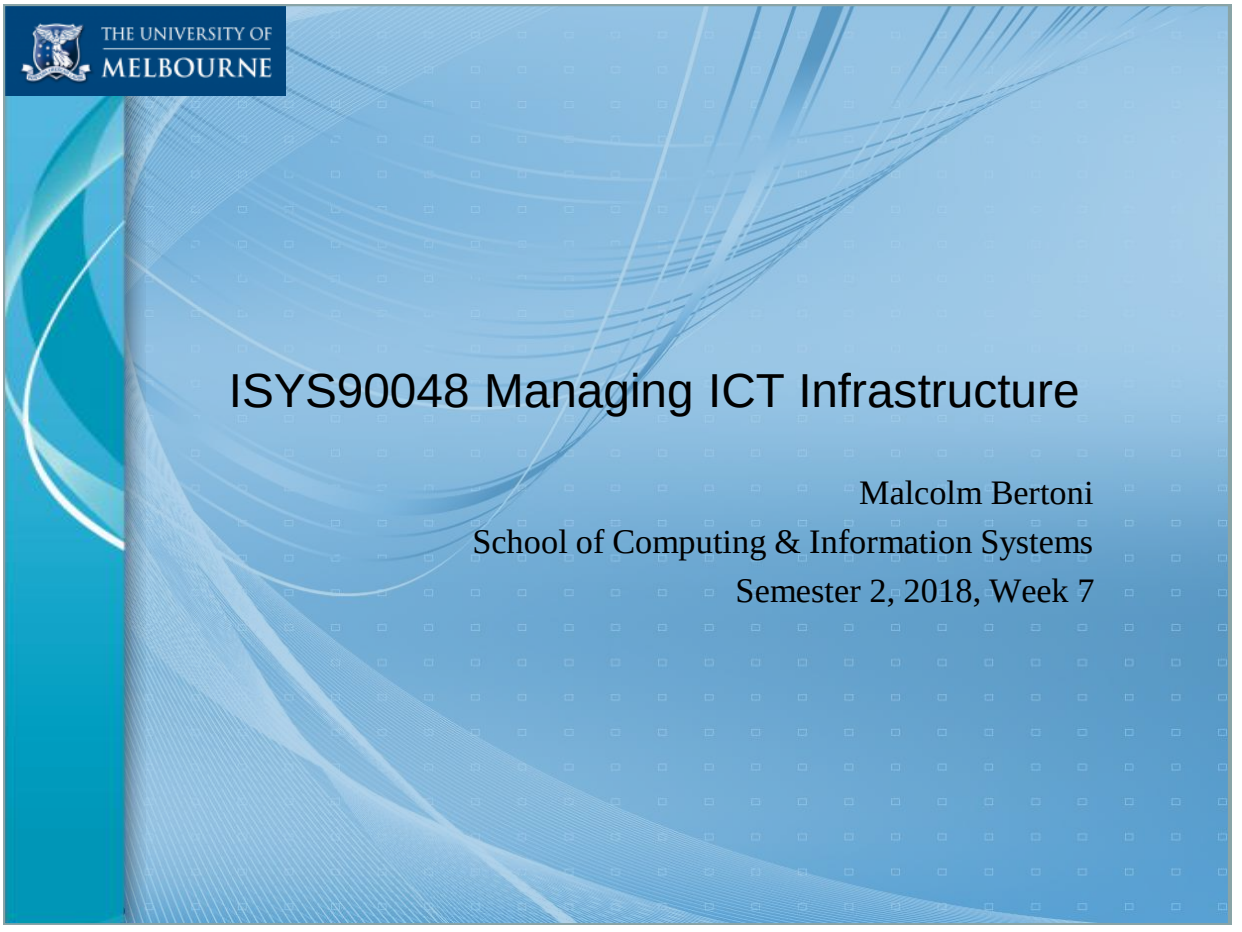

THE UNIVERSITY OF
MELBOURNE
ISYS90048 Managing ICT Infrastructure
Malcolm Bertoni
School of Computing & Information Systems
Semester 2, 2018, Week 7

THE UNIVERSITY OF MELBOURNE

Teaching Session 07

- Demonstrate an understanding of the role and importance of:
  - Tight vs loose coupling
  - Service Oriented Architectures (SOA)
  - Enterprise Service Bus (ESB)
  - Web service configuration and service-oriented architectures
  - Integration of SOAs and Cloud-based solutions

THE UNIVERSITY OF MELBOURNE

How it all began

- Tight and Loose Coupling
  - **Tight coupling** is a technique in which hardware and software components are highly dependent on each other
    - When we change one object in a tightly coupled application often it requires changes to a number of other objects
    - For example, an operating system would be considered tightly coupled as it depends on software drivers to correctly install and activate the system's peripheral devices
    - Costly to maintain
    - Complex
    - Slow and costly to change
    - Does not support reuse
  - **Loose coupling** allows us to reduce the inter-dependencies between components of a system with the goal of reducing the risk that changes in one component will require changes in any other component

THE UNIVERSITY OF MELBOURNE

How it all began cont

- Loose coupling was an elusive goal for the ICT industry
- It finally began happening through the development of services – Web Services is an example of a service
- Software components capable of communicating with each other over multiple networks using universally accepted open standards
  - Concept based on Extensible Markup Language (XML)
  - Web services enable disparate pieces of software to communicate and operate with each other regardless of the platform and programming language being used
  - The vision of Web Services is a world where systems can discover and utilise each other's capabilities without human intervention

THE UNIVERSITY OF MELBOURNE

Adoption of Web Services

Driving forces:

- Provides semantic, technical and organisational interoperability
- Industry-wide standards
- Reduced error rates due to data exchange to standards
- Reduced development times – through use of existing standards
- Reduced maintenance costs – through support of open standardised protocols
- Access to, and availability of, increased ranges of web-based application services
- Enables loose coupling

Inhibiting forces:

- Lack of agreed industry (de facto) or approved standards
- Evolving nature of standards
- Organisational changes – mergers, acquisitions, new entrants & substitutions

THE UNIVERSITY OF MELBOURNE

Service-Oriented Architecture

- Service-Oriented Architecture (SOA) ICT architecture model:
  - A framework for integrating business processes and supporting ICT infrastructure as secure, standardised components (services) that can be reused and combined to address changing business priorities
  - Initial integration efforts in SOA architectures were based on enterprise application integration (EAI) solutions
    - The main drawback was the tight coupling
  - In order to cope with the drawbacks of the EAI, the enterprise service bus (ESB) paradigm was proposed which follows the SOA approach
  - Within the SOA paradigm, an ESB performs as a mediator facilitating the provision and consumption of services
  - The use of an ESB increases the availability, reliability, performance and scalability and facilitates maintenance and changes

THE UNIVERSITY OF MELBOURNE

SOA Services

- Services in an SOA are activated by messages using a standards-based language over a standards-based communication protocol (eg: XML)
- It enables loose coupling and reuse of applications & services using standard-based interfaces and services, ensuring interoperability
- Enables large units of functionality to be built almost entirely from existing software service components
  - Eg: Middleware (Adapters, SOAP, etc)
  - Services are made available to multiple applications
  - Seamlessly connects separate technology systems

THE UNIVERSITY OF MELBOURNE

Service-Oriented Architecture

- SOA is often compared to Lego blocks:
  - It can be plugged, unplugged and interchanged with all other Lego blocks, so that it can be used whenever and wherever it's needed
  - The SOA turns your ICT systems into the equivalent of Lego blocks – you can move and reconfigure at will
  - Services are autonomous units
- SOA is an architecture, whereas Services is a technology that can be used to implement SOA
  - Essential components of an SOA are services, enabling technology, SOA governance and policies, SOA parameters, organisational and behaviour models
  - The architecture is the blueprint for the system and therefore high-level plan for the system
  - SOA can be used to integrate multiple applications that use different platforms

SOA Advantages/Disadvantages & Issues

Advantages:

- SOA improves interoperability
- The services are independent and do not depend on the context or state of the other services
  - They work within a distributed systems architecture and are reusable
- Business Process optimisation with seamless end to end process
- Information consistency
- Reducing impact of change
- Flexible Business Processes
- Simplifies the architects views of the systems world

Disadvantages/Issues:

- Introduce another level of complexity and specification
- Add another level of message handling
- Current web applications work okay
- Require a change in design philosophy and a focus on a new service paradigm
- SOAs require new tools and interfaces that may not yet been fully implemented for all systems
- Lack of standards
- Takes time to implement - an incremental approach is preferred
- Requires careful consideration of the SOA infrastructure components needed to ensure scalability, performance, governance & management

THE UNIVERSITY OF MELBOURNE

The Eight SOA Principles/Characteristics

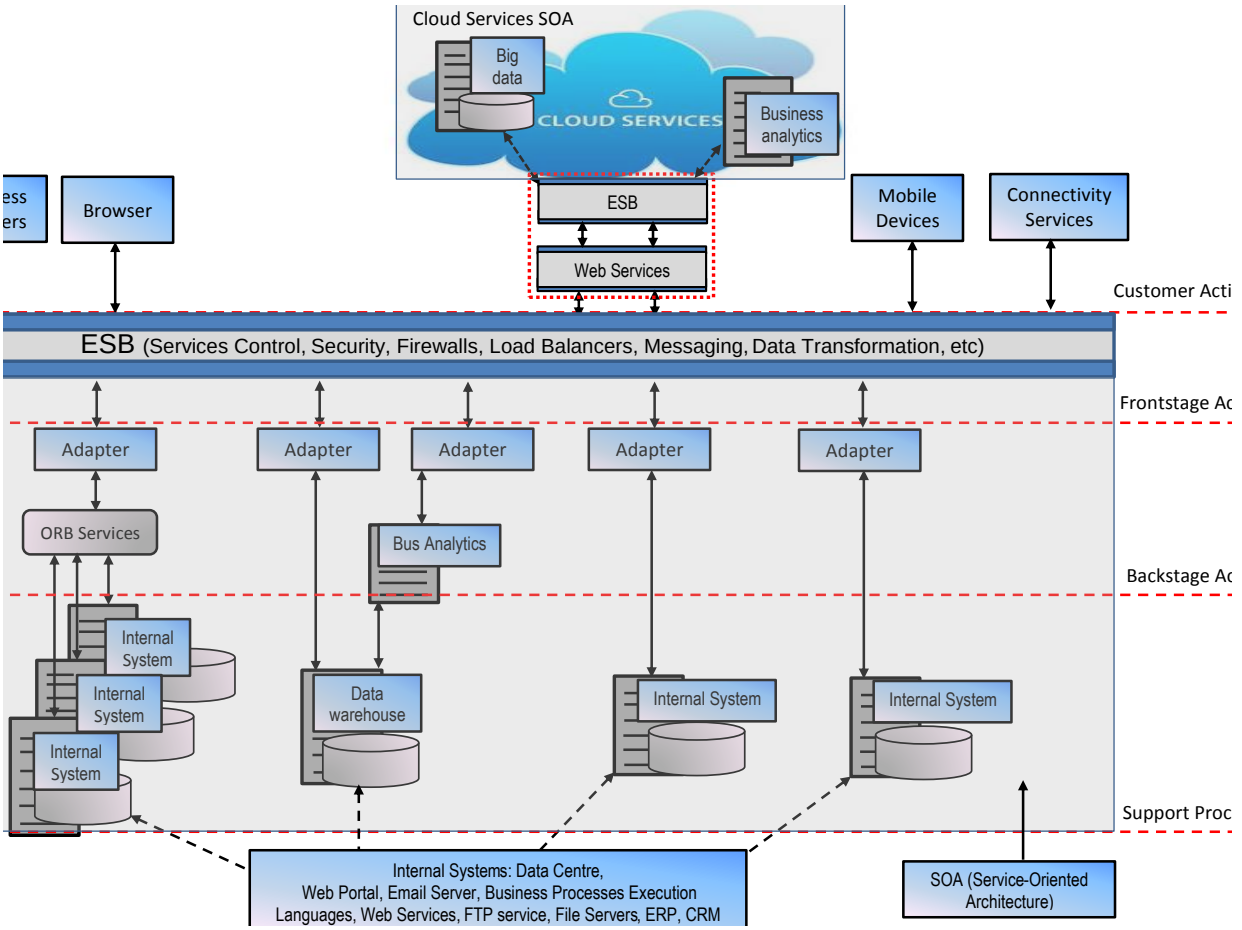
1. **Loose coupling** Services maintain relationship that minimises dependencies & only requires that they retain an awareness of each other
2. **Service contract** Services adhere to a communications agreement, as defined collectively by one or more service descriptions and related documents
3. **Autonomy** Services have control over the logic they encapsulate
4. **Abstraction** Beyond what is described in the service contract, services hide logic from the outside world
5. **Reusability** Logic is divided into services for the promoting of reuse
6. **Composability** Collections of services can be coordinated and assembled to form composite services
7. **Statelessness** Services minimise retaining information specific to an activity
8. **Discoverability** Services are designed to be outwardly descriptive so that they can be found and assessed via available discovery mechanisms

THE UNIVERSITY OF MELBOURNE

SOA summary

An SOA is then comprised of:

- Messages that are the units of communication between Service Handlers
  - All activities are triggered by the exchange of messages, and no activity can occur without a message exchange
  - Service Handlers group collections of related operations and are defined by the operations that they are comprised of
- Operations that send and receive messages to perform work, and are defined by the messages that they process
  - Operations are invoked by process instances to complete their automated work


Cloud Services SOA
Big data
CLOUD SERVICES
Business analytics
ESB
Web Services
Browser
Mobile Devices
Connectivity Services
Customer Acti
ESB (Services Control, Security, Firewalls, Load Balancers, Messaging, Data Transformation, etc)
Frontstage Ac
Adapter
Adapter
Adapter
Adapter
Adapter
ORB Services
Bus Analytics
Backstage Ac
Internal System
Internal System
Internal System
Data warehouse
Internal System
Internal System
Support Proc
Internal Systems: Data Centre,
Web Portal, Email Server, Business Processes Execution
Languages, Web Services, FTP service, File Servers, ERP, CRM
SOA (Service-Oriented Architecture)

THE UNIVERSITY OF MELBOURNE

SOA Implementation

- Having a business model is critical to the successful alignment of services with business goals and objectives, and consequently to the overall SOA implementation's success
  - Business drivers such as strategy, competition, market forces, regulatory forces
- Services that combine in a variety of different ways to support existing applications and systems
- SOAs can be designed and constructed according to a number of distinct development methodologies:
- Top down – align services with the business processes
- Bottom up – creates services on a needs basis
- Agile – combining top down and bottom up

THE UNIVERSITY OF MELBOURNE

SOA Governance

- The goal of applying governance to SOA is to get the most out of your SOA
- This requires taking the following steps:
  1. Define the policies you want to apply
     - People, Processes, Policies
  2. Apply these policies during design time and maintaining those policies
  3. Monitor and enforce the policies during runtime
     - Eg: Security Policy that ensures that all publicly provided services should be made over a secure channel
- SOA Governance keeps track of how services are used and keeps uniformity amongst services

THE UNIVERSITY OF MELBOURNE

Enterprise Service Bus

- An Enterprise Service Bus is the service interface of an SOA, extended to a whole of enterprise
  - An ESB is not required for SOA, but it does increase the power and flexibility of an SOA
- All service requests and responses go through the ESB
- The core concept of the ESB architecture is to integrate different applications by putting a communication bus between them & then enable each application to talk to the bus
- An ESB does not dictate whether components that use the bus are local to it or remote, nor does it enforce any specific requirements for programming languages
  - Instead, it acts to unify the various ways in which components can receive or send information to other applications

THE UNIVERSITY OF MELBOURNE

Enterprise Service Bus Functions

- Monitor and control routing of messages between services – from requester to responder, and from requester to intermediate services
- Control the deployment, update and versioning of services
- Monitor service usage, error rates, security attacks, redundant/disused services
- Intelligent routing
- Real time monitoring
- Service security
- Often needed to transform messages into a format that an application can interpret by using adapters
  - The adapter is responsible for talking to the backend application and transforming data from the application format to the bus format
  - This decouples systems from each other, allowing them to communicate without dependency on or knowledge of other systems on the bus

THE UNIVERSITY OF MELBOURNE

Problems with an ESB

- The challenge in the ESB concept is there is no single accepted standard for features or behaviour
- Middleware analysis skills needed to configure, manage, and operate an ESB
- ESB becomes a single point of failure
- Can be expensive

THE UNIVERSITY OF MELBOURNE

Benefits of an ESB

- Because an ESB architecture controls the way that work moves, it makes it easy to change components or add additional components
- Because an ESB sees everything, it also makes for a convenient place to enforce security and compliance requirements, log normal or exception conditions and even handle transaction performance monitoring
- An ESB also provides load balancing
- Cloud Services Platform-as-a-Service (PaaS) can complement an ESB
- It can also often provide failover support should a component or its resources fail
- An ESB integrates a variety of disparate technologies
- Allows for additional layers of abstraction

Failover: a procedure by which a system automatically transfers control to a duplicate system when it detects a fault or failure

THE UNIVERSITY OF MELBOURNE

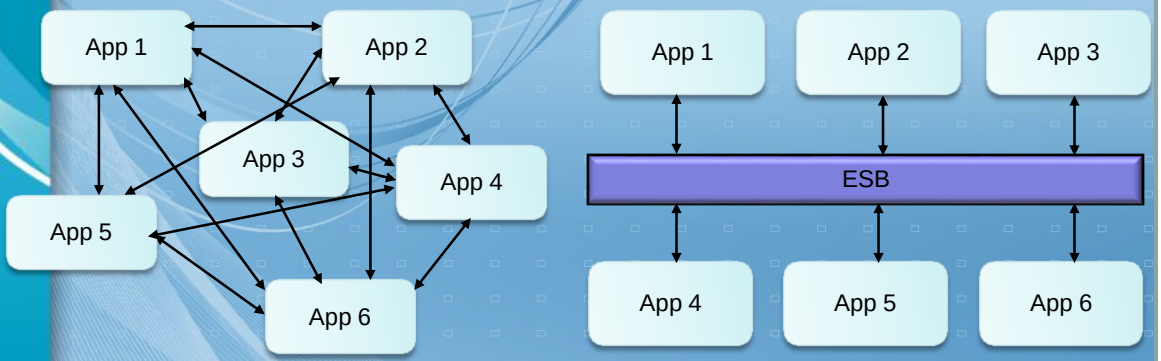
Without ESB

- Too many point-to-point links
- Multiple protocols, different qualities of service
- No clear picture of available services
- Many applications communicating with each other

THE UNIVERSITY OF MELBOURNE

Cleaning up the mess

- Now that you know that systems don't exchange information directly (ie, loosely coupled) and you understand what a service is, you can start making use of an ESB



THE UNIVERSITY OF MELBOURNE

Key Differences between SOA & ESB

- SOA is an architectural approach where we use ‘services’, whereas ESB is a technical implementation that aids in delivering a SOA
- SOA brings cost effective, reusable and low lead time solutions to an organisation whereas ESB enables low cost integration
- SOA is way of building the next generation of applications from ‘lego blocks’ called Services whereas ESB is a piece of infrastructure software that provides APIs for developers to create services and send messages between services
- SOA is just like a car and ESB is like a road on which this car runs

THE UNIVERSITY OF MELBOURNE

ESB Summary

- An enterprise service bus should be considered as an architecture style or pattern and not as a product
- There is no definition or specification for the ESB and therefore no standard
- An ESB can help achieve looser coupling between systems
- A service on an ESB is stateless:
  - No information is retained by either sender or receiver

Adoption of Standardised Data Interchange

Driving forces:

- Semantic (language), technical and organisational interoperability – easier exchange of data
- Reduced development times – through use of existing standards for database design and existing software modules
- Reduced maintenance costs – through use of standardised modules

Inhibiting forces:

- Costs to develop and propagate standard definitions
- Costs to change-over from existing to new, standardised systems
- Lack of clear industry-adopted standards
- Organisational changes – mergers, acquisitions, new entrants and substitutions

Adoption of Standardised Communication Protocols

Driving forces:

- Semantic, technical and organisational interoperability over networks – easier exchange of data
- Reduced development times – through use of existing standards for network protocols
- Reduced maintenance costs – though use of standardised protocols

Inhibiting forces:

- Different software solutions use different protocols
- Lack of agreed industry (de facto) or approved standards
- Lack of knowledge and training
- Need to translate the protocol to fully handle semantic interoperability
- Organisational changes – mergers, acquisitions, new entrants and substitutions

THE UNIVERSITY OF MELBOURNE

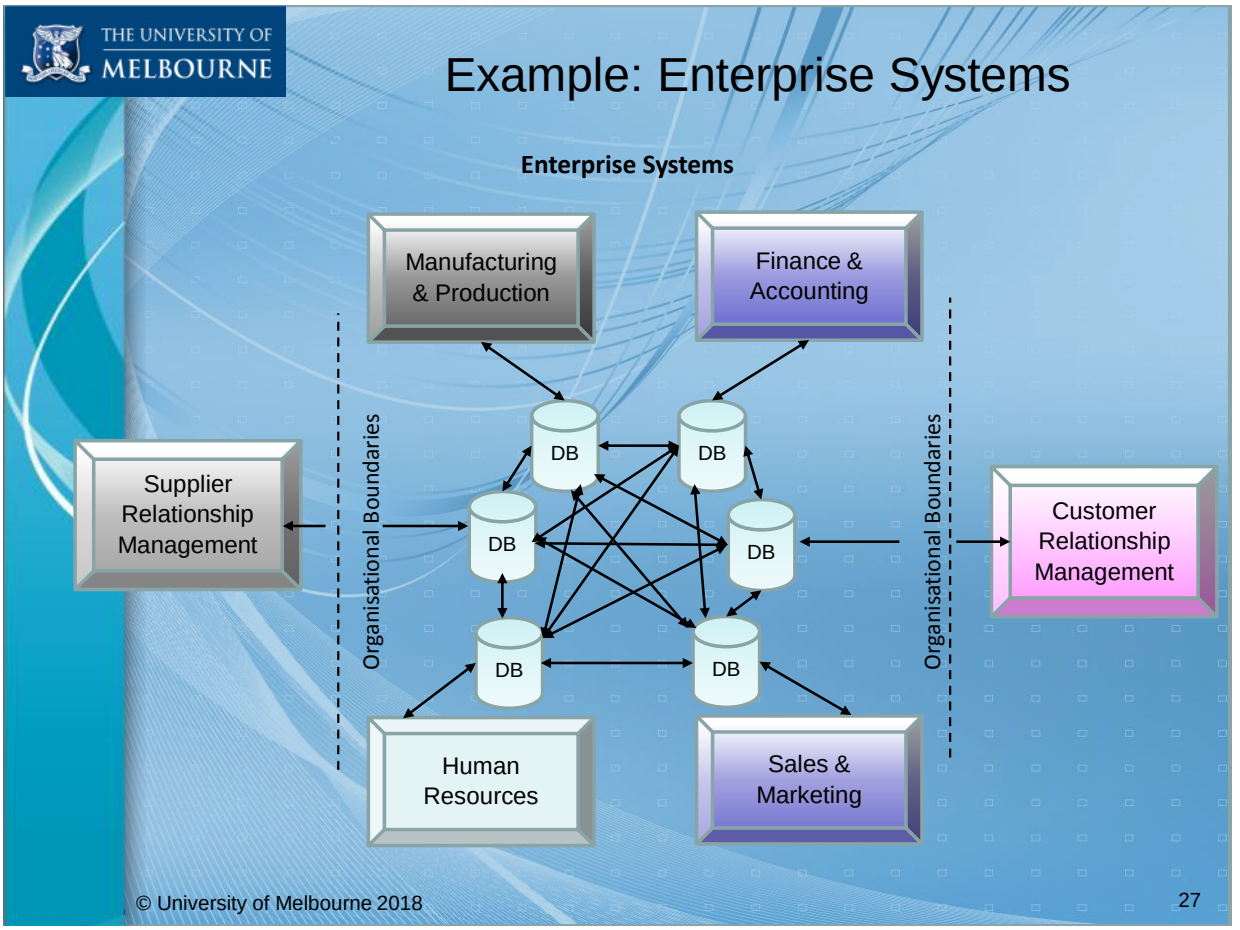
Example: Enterprise Systems

- Enterprise systems are large-scale, complex, expensive software packages that support business processes, (ERP-systems, HR, payroll, manufacturing/production, Sales/marketing)
- To implement these enterprise systems, organisations need to:
  - Translate their business processes into ICT-enabled processes
  - Select the systems and the functions they wish to use
  - Customise these systems to meet their specific requirements
  - Integrate these systems, so that they operate correctly together
    - In many organisations data are not kept in a single repository, but across dozens if not hundreds of separate computer systems each housed in an individual function, business unit, region, factory, or office

THE UNIVERSITY OF MELBOURNE

Example: Enterprise Systems

- Most Enterprise Systems have not been able to communicate with other systems in a transparent way
- Enormously complex pieces of software, and installing them requires large investments of money, time, and expertise
- Every organisation collects, generates, and stores vast quantities of data
  - In most companies the data are not kept in a single repository, but across dozens if not hundreds of separate computer systems each housed in an individual function, business unit, region, factory, or office


THE UNIVERSITY OF MELBOURNE
Example: Enterprise Systems
Enterprise Systems
Manufacturing & Production
Finance & Accounting
Supplier Relationship Management
Organisational Boundaries
DB
DB
DB
DB
DB
DB
Organisational Boundaries
Customer Relationship Management
Human Resources
Sales & Marketing
© University of Melbourne 2018
27

THE UNIVERSITY OF MELBOURNE

Enterprise Systems to Integrated Enterprise Systems

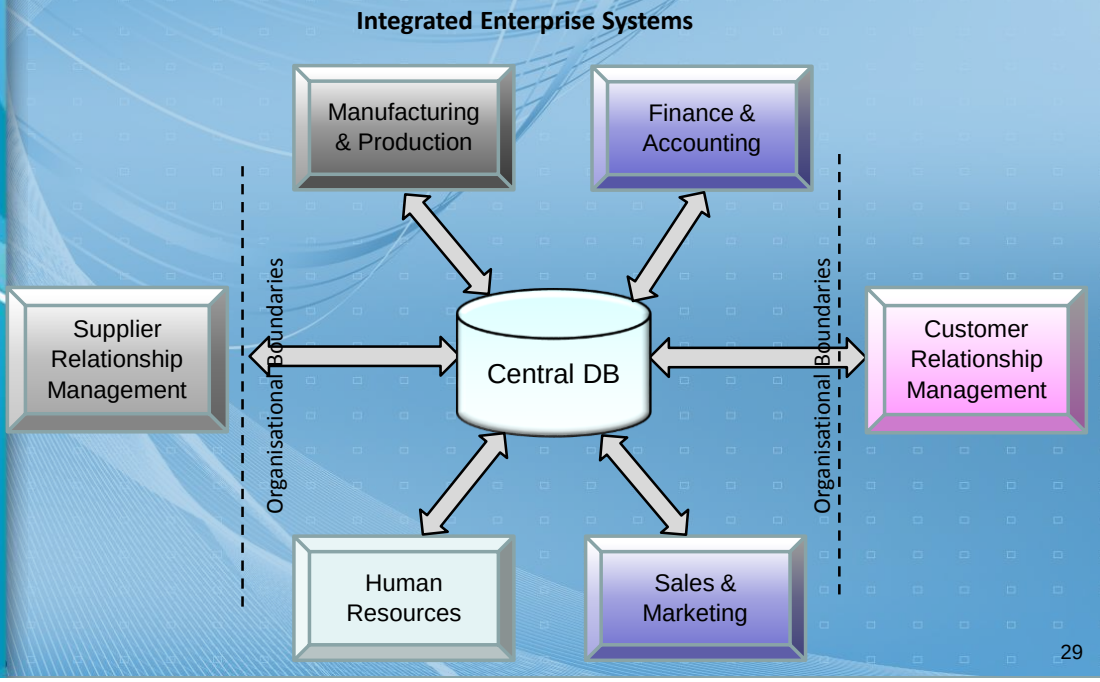
- Enterprise systems are now evolving from monolithic silos with tightly coupled processes to distributed applications with service-oriented architectures (SOA)
  - Today's business must be able to flexibly and quickly adapt to market needs
  - To achieve this, requires:
    - **Standardisation** of data and business processes
    - No **duplication** of separate systems
    - **Business modularity**
  - SOA allows organisations to use commercial off-the-shelf (COTS) software that can then be loosely coupled
- Now Integrated Enterprise Systems

Modularity: the degree to which a system's components may be separated and recombined

THE UNIVERSITY OF MELBOURNE

Integrated Enterprise System

- At the heart of an Integrated Enterprise System is a central database
- Using a single central database dramatically streamlines the flow of information



THE UNIVERSITY OF MELBOURNE

Integrated Enterprise Systems Benefits

- Organisation-wide integration of the key business processes
- Information can flow seamlessly across the enterprise
- Different business processes from sales, production, manufacturing, logistics, and human resources, can be integrated into one organisation-wide business processes
- Business processes to be automated
- Business performance information readily available
- Transaction processing, hardware, software, and IT support staff costs reduced
- Improved quality and efficiency of customer service, production and distribution by integrating the company's internal business processes in sales, finance, production, custom logistics, etc

THE UNIVERSITY OF MELBOURNE

Summary

- Service Oriented Architectures are an attempt to made networked services tangible in terms of composable and controllable Service Handlers
- SOA is a design philosophy for ICT architectures that spans both internal and cloud-based application services
- SOA is aligned with the general concepts of service level management
- Service Level Agreements form the basis of quality assurance of service delivery
- ESB provides a service interface for SOA
- Enterprise Systems are evolving into Integrated Enterprise Systems, focusing on interconnection, data interchange and distributed environments

THE UNIVERSITY OF MELBOURNE

References

- Channabasavaiah, K, Holley, K & Tuggle, E (2003). "Migrating to a Service-oriented Architecture, Part 1" IBM developerWorks, 8 pp
- Channabasavaiah, K, Holley, K & Tuggle, E (2003). "Migrating to a Service-oriented Architecture, Part 2" IBM developerWorks, 8 pp
- Computerworld (Sept 5, 2005). SOA and ESB Are More Than Different Answers to the Same Problem. Viewed 5 June, 2018, https://www.computerworld.com/article/2556953/it-management/soa-and-esb-are-more-than-different-answers-to-the-same-problem.html
- Erl, Thomas (2005). Service-Oriented Architecture: Concepts, Technology, and Design. Prentice Hall
- IMB Global Technology Services (May 2007). Infrastructure considerations for service-oriented architecture, 35 pp.

THE UNIVERSITY OF MELBOURNE

References cont

- Luthria, H & Rabhi, FA (2008) Building the Business Case for SOA: A Study of the Business Drivers for Technology Infrastructure Supporting Financial Service Institutions, Australian School of Business, UNSW, 15 pp
- Pisello, T (2006) "Is There Real Business Value Behind the Hype of SOA?" Computerworld Management, IDG, 4 pp
- Pulier, E and Taylor, H, (2006). Understanding Enterprise SOA. 242 pp
- Zaigham Mahmood (2007). Service Oriented Architecture: Potential Benefits and Challenges. International Conference on Computers, 497-501

THE UNIVERSITY OF MELBOURNE

Videos

- SOA (5 min) https://www.coursera.org/learn/python-network-data/lecture/0CpCx/video-service-oriented-architectures
- SOA (5.54 min) https://www.youtube.com/watch?v=L1tM0tMJdzY
- ESB (4 min) https://www.youtube.com/watch?v=VHzWswQNtgk