Application system reuse

- An application system product is a software system that can be adapted for different customers without changing the source code of the system.
- Application systems have generic features and so can be used/reused in different environments.
- Application system products are adapted by using built-in configuration mechanisms that allow the functionality of the system to be tailored to specific customer needs.
    - For example, in a hospital patient record system, separate input forms and output reports might be defined for different types of patient.

Benefits of application system reuse

- As with other types of reuse, more rapid deployment of a reliable system may be possible.
- It is possible to see what functionality is provided by the applications and so it is easier to judge whether or not they are likely to be suitable.
- Some development risks are avoided by using existing software. However, this approach has its own risks, as I discuss below.
- Businesses can focus on their core activity without having to devote a lot of resources to IT systems development.
- As operating platforms evolve, technology updates may be simplified as these are the responsibility of the COTS product vendor rather than the customer.

Problems of application system reuse

- Requirements usually have to be adapted to reflect the functionality and mode of operation of the COTS product.
- The COTS product may be based on assumptions that are practically impossible to change.
- Choosing the right COTS system for an enterprise can be a difficult process, especially as many COTS products are not well documented.
- There may be a lack of local expertise to support systems development.
- The COTS product vendor controls system support and evolution.

Configurable application systems

- Configurable application systems are generic application systems that may be designed to support a particular business type, business activity or, sometimes, a complete business enterprise.
  - For example, an application system may be produced for dentists that handles appointments, dental records, patient recall, etc.
- Domain-specific systems, such as systems to support a business function (e.g. document management) provide functionality that is likely to be required by a range of potential users.

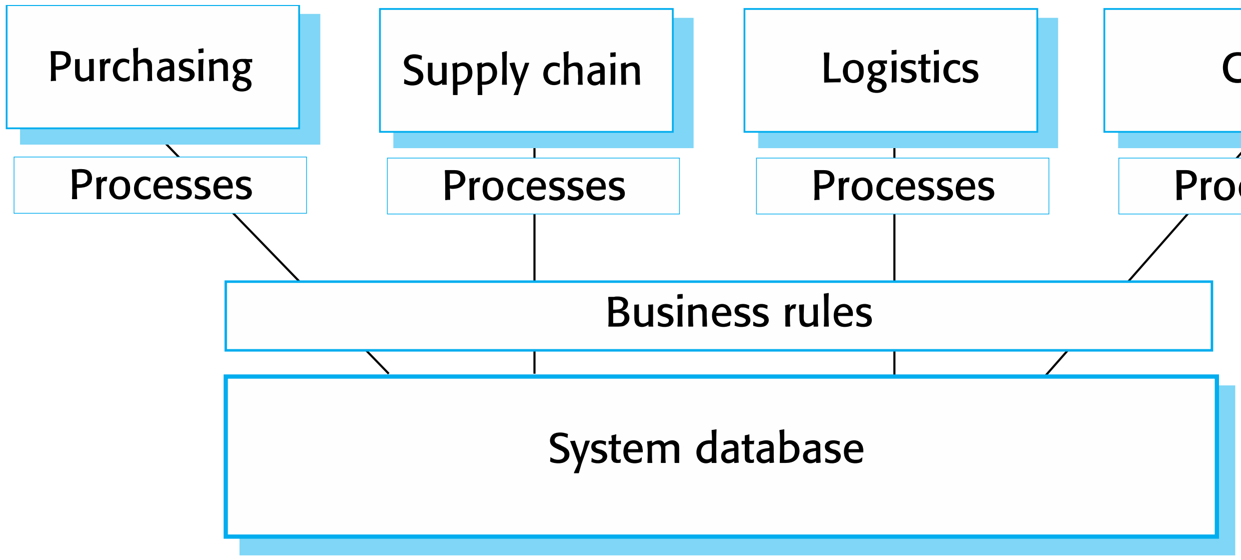
COTS-solution and COTS-integrated systems

Configurable application systems	Application system integration
Single product that provides the functionality required by a customer	Several heterogeneous system products are integrated to provide customized functionality
Based around a generic solution and standardized processes	Flexible solutions may be developed for customer processes
Development focus is on system configuration	Development focus is on system integration
System vendor is responsible for maintenance	System owner is responsible for maintenance
System vendor provides the platform for the system	System owner provides the platform for the system

ERP systems

- An Enterprise Resource Planning (ERP) system is a generic system that supports common business processes such as ordering and invoicing, manufacturing, etc.
- These are very widely used in large companies - they represent probably the most common form of software reuse.
- The generic core is adapted by including modules and by incorporating knowledge of business processes and rules.

The architecture of an ERP system



ERP configuration

- Selecting the required functionality from the system.
- Establishing a data model that defines how the organization’s data will be structured in the system database.
- Defining business rules that apply to that data.
- Defining the expected interactions with external systems.
- Designing the input forms and the output reports generated by the system.
- Designing new business processes that conform to the underlying process model supported by the system.
- Setting parameters that define how the system is deployed on its underlying platform.

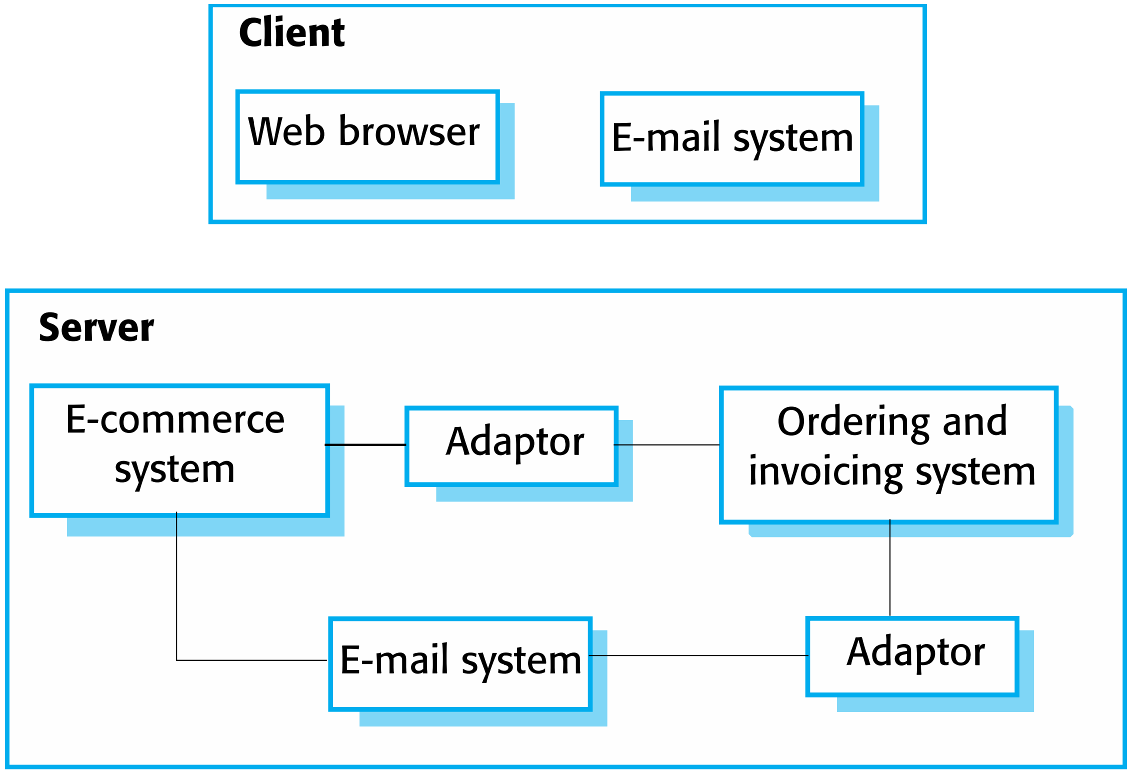
Integrated application systems

- Integrated application systems are applications that include two or more application system products and/or legacy application systems.
- You may use this approach when there is no single application system that meets all of your needs or when you wish to integrate a new application system with systems that you already use.

Design choices

- Which individual application systems offer the most appropriate functionality?
  - Typically, there will be several application system products available, which can be combined in different ways.
- How will data be exchanged?
  - Different products normally use unique data structures and formats. You have to write adaptors that convert from one representation to another.
- What features of a product will actually be used?
  - Individual application systems may include more functionality than you need and functionality may be duplicated across different products.

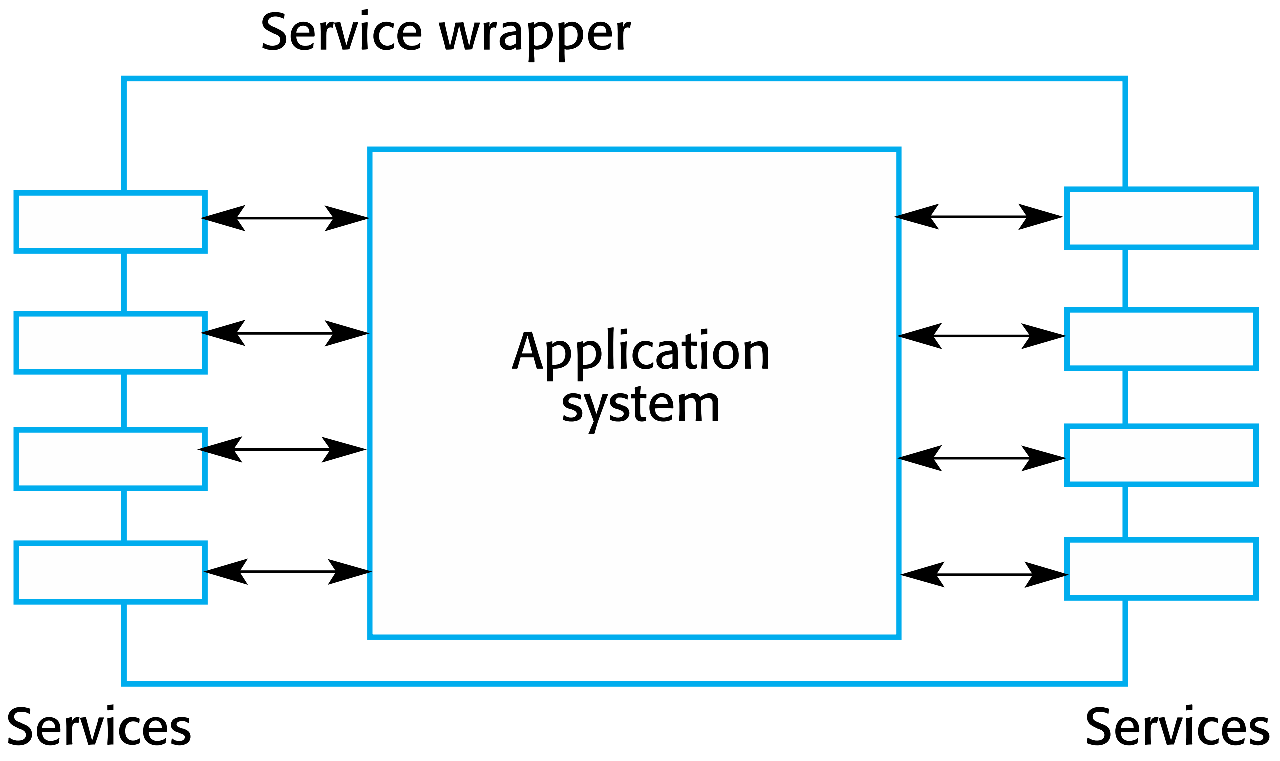
An integrated procurement system



Service-oriented interfaces

- Application system integration can be simplified if a service-oriented approach is used.
- A service-oriented approach means allowing access to the application system's functionality through a standard service interface, with a service for each discrete unit of functionality.
- Some applications may offer a service interface but, sometimes, this service interface has to be implemented by the system integrator. You have to program a wrapper that hides the application and provides externally visible services.

Application wrapping



Application system integration problems

- Lack of control over functionality and performance
  - Application systems may be less effective than they appear
- Problems with application system inter-operability
  - Different application systems may make different assumptions that means integration is difficult
- No control over system evolution
  - Application system vendors not system users control evolution
- Support from system vendors
  - Application system vendors may not offer support over the lifetime of the product

Open source development

- Open source development is an approach to software development in which the source code of a software system is published and volunteers are invited to participate in the development process
- Its roots are in the Free Software Foundation (www.fsf.org), which advocates that source code should not be proprietary but rather should always be available for users to examine and modify as they wish.
- Open source software extended this idea by using the Internet to recruit a much larger population of volunteer developers. Many of them are also users of the code.

Open source systems

- The best-known open source product is, of course, the Linux operating system which is widely used as a server system and, increasingly, as a desktop environment.
- Other important open source products are Java, the Apache web server and the mySQL database management system.

Open source issues

- Should the product that is being developed make use of open source components?
- Should an open source approach be used for the software’s development?

Open source business

- More and more product companies are using an open source approach to development.
- Their business model is not reliant on selling a software product but on selling support for that product.
- They believe that involving the open source community will allow software to be developed more cheaply, more quickly and will create a community of users for the software.

Open source licensing

- A fundamental principle of open-source development is that source code should be freely available, this does not mean that anyone can do as they wish with that code.
  - Legally, the developer of the code (either a company or an individual) still owns the code. They can place restrictions on how it is used by including legally binding conditions in an open source software license.
  - Some open source developers believe that if an open source component is used to develop a new system, then that system should also be open source.
  - Others are willing to allow their code to be used without this restriction. The developed systems may be proprietary and sold as closed source systems.

License management

- Establish a system for maintaining information about open-source components that are downloaded and used.
- Be aware of the different types of licenses and understand how a component is licensed before it is used.
- Be aware of evolution pathways for components.
- Educate people about open source.
- Have auditing systems in place.
- Participate in the open source community.

Key Points

- There are many different ways to reuse software. These range from the reuse of classes and methods in libraries to the reuse of complete application systems.
- The advantages of software reuse are lower costs, faster software development and lower risks. System dependability is increased. Specialists can be used more effectively by concentrating their expertise on the design of reusable components.
- Application frameworks are collections of concrete and abstract objects that are designed for reuse through specialization and the addition of new objects. They usually incorporate good design practice through design patterns.

Key Points

- Software product lines are related applications that are developed from one or more base applications. A generic system is adapted and specialized to meet specific requirements for functionality, target platform or operational configuration.
- Application system reuse is concerned with the reuse of large-scale, off-the-shelf systems. These provide a lot of functionality and their reuse can radically reduce costs and development time. Systems may be developed by configuring a single, generic application system or by integrating two or more application systems.
- Potential problems with application system reuse include lack of control over functionality and performance, lack of control over system evolution, the need for support from external vendors and difficulties in ensuring that systems