Software Reuse

Definition: Software reuse

Using existing software artifacts to construct a new software system

Definition: Artifact

A piece of formalised knowledge that can contribute to the software development process

There are two types of software artifacts

- Software products that are created as deliverables during the development process
- Development knowledge that is applied to the process

1 Reuse levels

Abstraction level

- Use the concepts, not actual code

Object level:

- Reuse objects or functions from code

Component level:

- Reuse a collection of objects and object classes

System level:

- Reuse the whole system

2 Reusable assets in the software lifecycle

Requirements:

- Analysis: Project models, histories, system simulations
- Requirements specifications, trade studies, cost models

Design:

- Designs, standards, frameworks, consumer reports, historical engineering costs, design simulations

Develop:

- Code modules, programmer's manuals, operating manuals, unit test cases, codes simulations, operating training programs

Integrate and test:

- System and subsystem test plans, procedures and cases; detailed interface simulations

Maintain:

- Problem/trouble report tracking systems, regression test plans, procedures, cases

3 Benefits of software reuse

- Reduced development costs - fewer components to spec, designed, implemented etc
- Accelerated development - reduction in development and validation time
- Increased dependability - tried and tested
- Reduced process risk - reduced margin of error in project cost estimation
- Effective use of specialists - encapsulation of their knowledge; not re-inventing the wheel
- Standards compliance

4 Problems with reuse

- Increased maintenance costs - source code of reused component etc not available
- ”Not-invented-here syndrome” - software engineers prefer to write own code as it has better quality
- Creating, maintaining and using a component library - populating and structuring can be expensive. The development processes used need to be adapted so they can use the library
- Finding, understanding and adapting reusable components- components in library need to be understood and adapted for new environment

5 COTS product reuse

Definition: COTS

Commercial off the shelf - a software system that can be adapted for different customers without changing the source code of the system

COTS systems have generic features and so can be used/reused in different environments

COTS products are adapted by using built in configuration mechanisms that allow the functionality of the system to be tailored to specific customer needs

5.1 Benefits of COTS 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

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

5.2 Problems of COTS 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 evaluation

5.3 Risks of using COTS

Vendor risks:

- Failure of vendor to provide support when required
- Vendor goes out of business or drops product from its portfolio

Product risks:

- Incompatible event/data model with other systems
- Inadequate performance when integrated with other systems
- Product is undependable in intended operating environment

Process risk

- Time required to understand how to integrate product is higher than expected

5.3.1 Risks reduction

- Only dealing with vendors that allow access to source code if they go out of business
- By extensive research and testing of product capabilities before using, discussion with other users etc
- In general though, because COTS are provided by external vendors, risk reduction is difficult

6 Reuse costs

- The costs of the time spent in looking for software to reuse and assessing whether or not it meets your needs
- Where applicable, the costs of buying the reusable software for large off the shelf systems can be very high
- The costs of adapting and configuring the reusable software components or systems to reflect the requirements of the system that you are developing
- The costs of integrating reusable software elements with each other and with the new code that you have developed