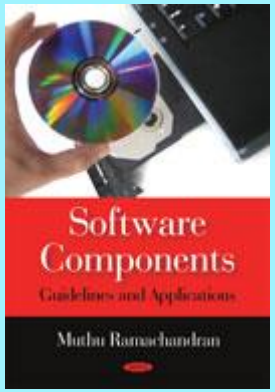
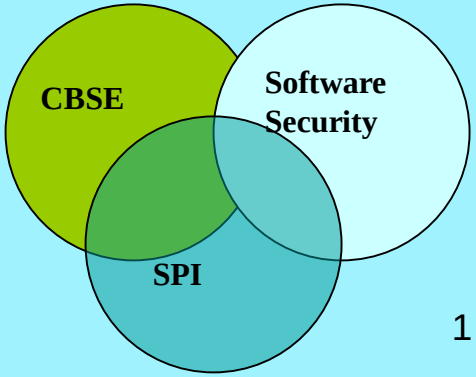
Enterprise Architecture for Large Scale Reuse



Prof Muthu Ramachandran

M.Ramachandran@leedsbeckett.ac.uk

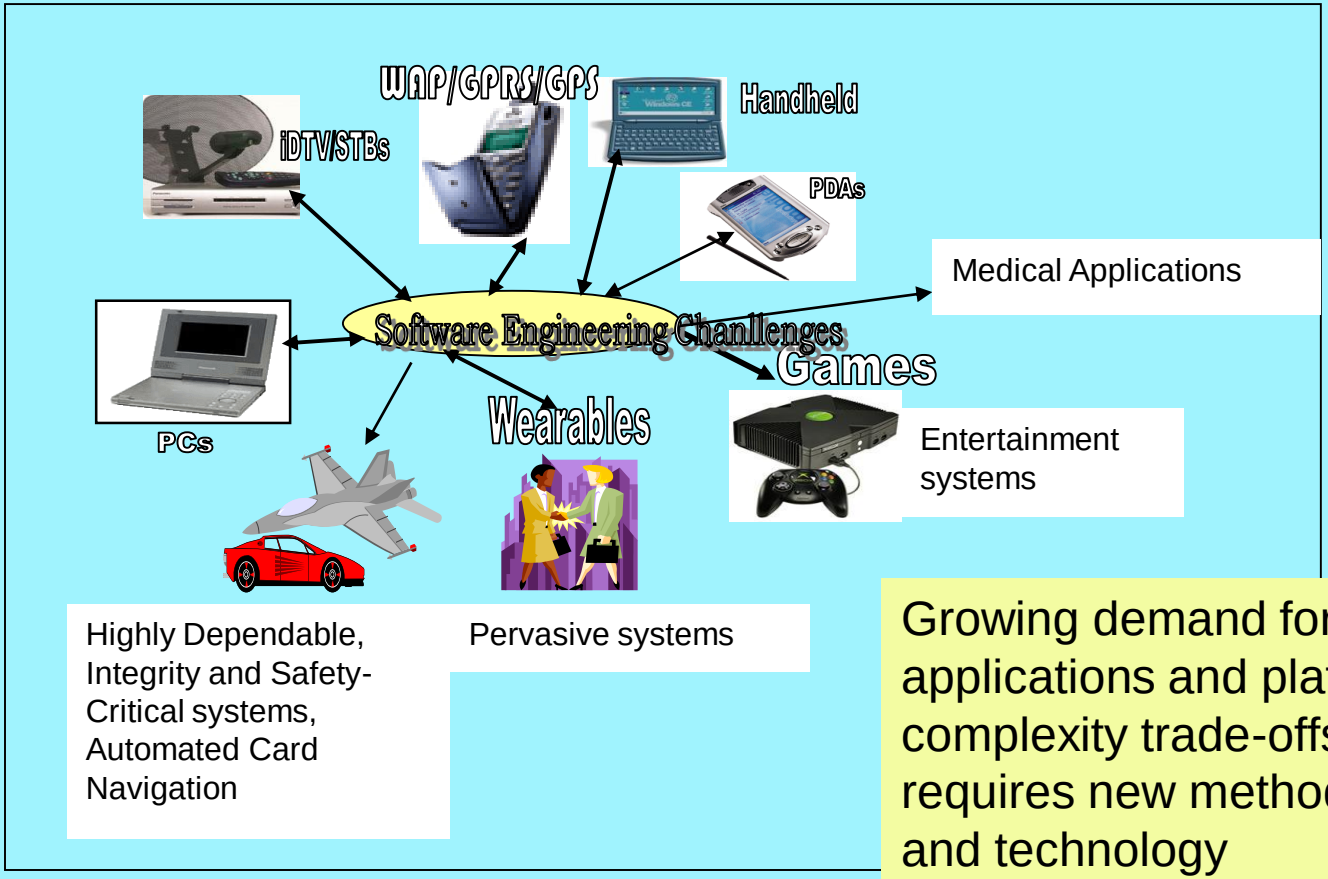


LEEDS BECKETT UNIVERSITY

Outline Today

- Outline of topics
  - Software Reuse and CBSE
  - Enterprise Architecture Design and Guidelines
  - Large scale software reuse

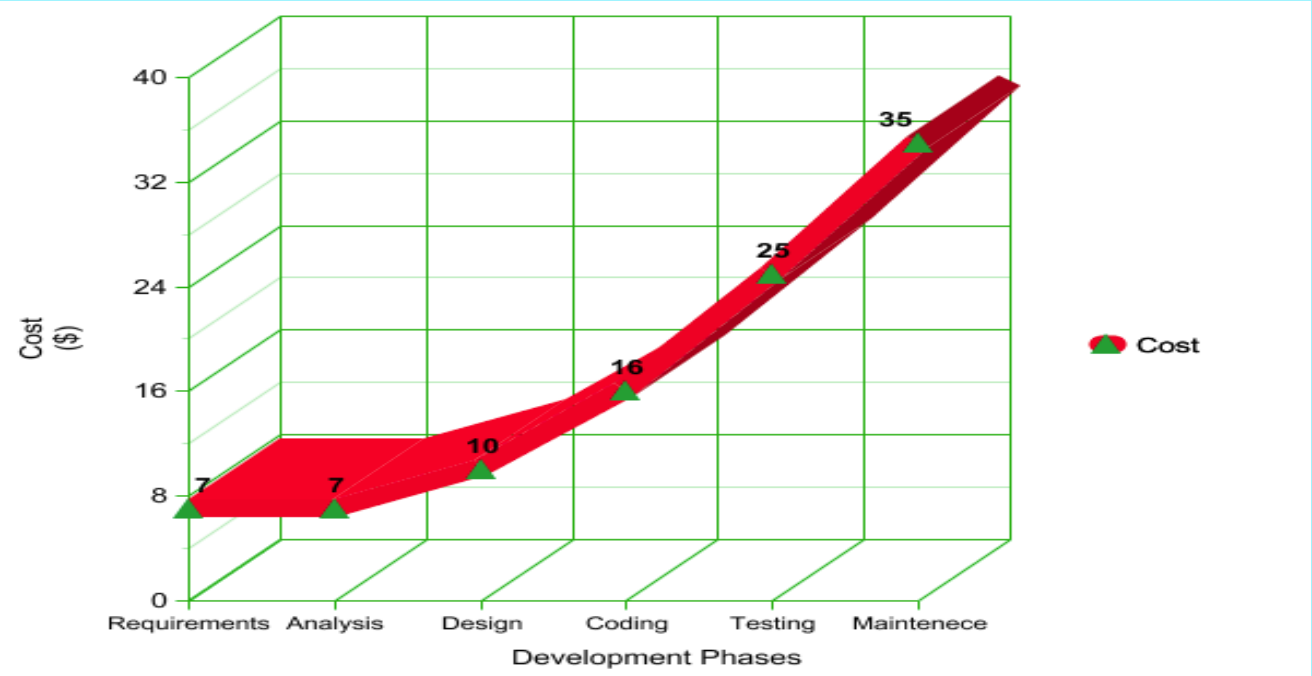
SE Role in Managing Complexity



Growing demand for applications and platforms: complexity trade-offs and requires new methods, design, and technology

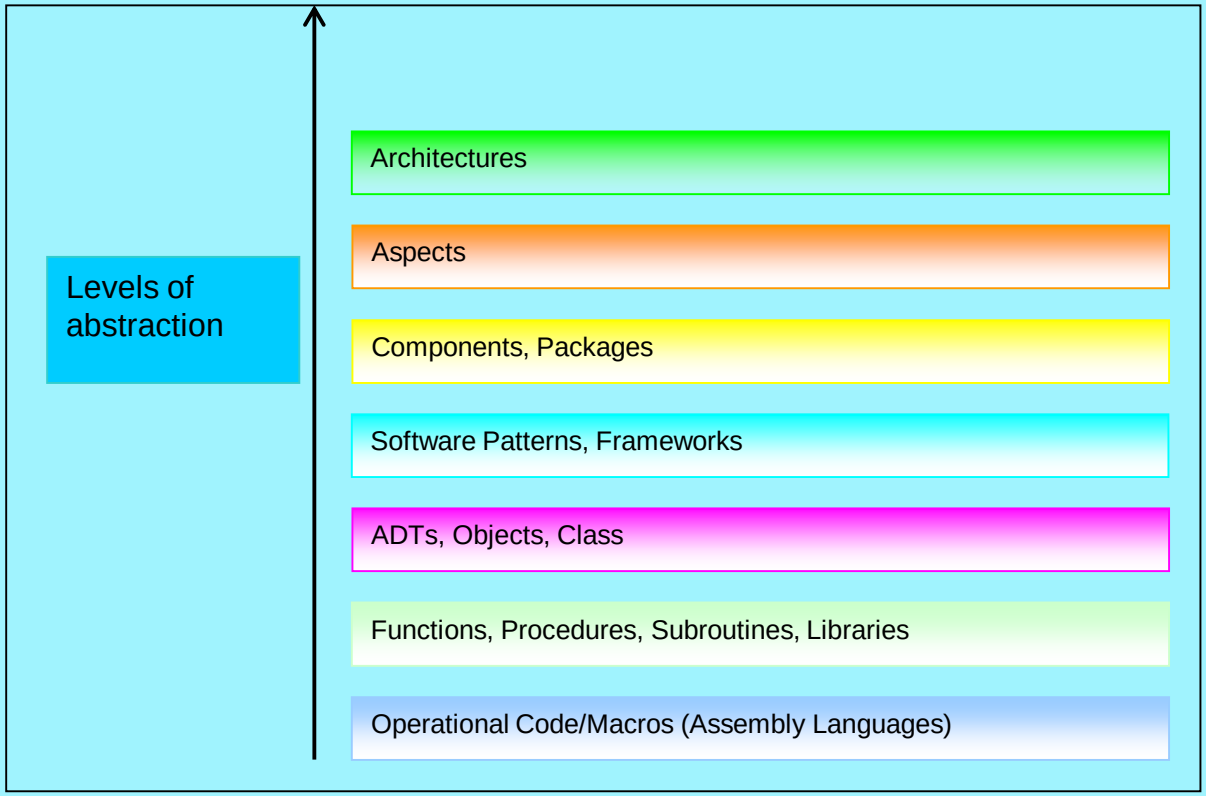
LEEDS BECKETT UNIVERSITY

Evolution of SE vs Development Cost: Software Crisis, No Silver Bullet, High Cost, Late Delivery



Ripple effect is the iron Law of SE: Change anything later part of the lifecycle (code) will cost you more to fix it.

Abstractions in Resolving Complexity



Evidenced from our experiences since the emergence of computing

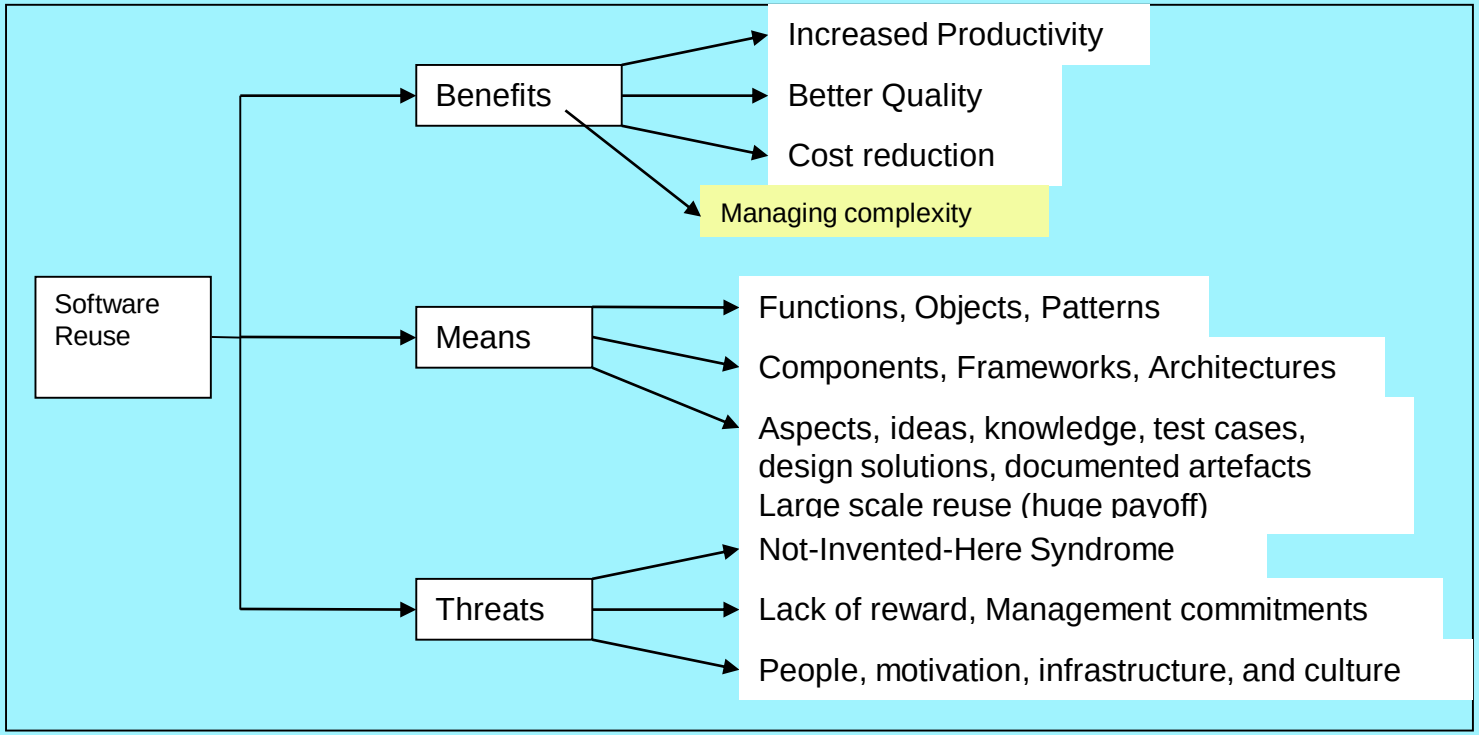
Object-oriented programming
A schematic paradigm for computer programming in which the linear concepts of procedures and tasks are replaced by the concepts of objects and messages. An object includes a package of data and a description of the operations that can be performed on that data. A message specifies one of the operations, but unlike a procedure, does not describe how the operation should be carried out. C++, C#, Java are examples of object-oriented programming language.

What is software reuse?

The concept of reuse can be compared with the activities such as constructing a building, designing a digital computer, and an automobile factory which manufactures thousands of Cars, Motors from a set of ready made parts. Software reuse means a process in which something (which may be a simple component or even a piece of information /specification) can be constructed/consulted using something that exists (which is called reusable). As a general term, reuse includes any of the following:

- A definition of software reuse is the process of creating software systems from predefined software components.
- Building systems on top of an existing O/S or DBMS.
- Building/assembling software systems by parts
- Using application generators.
- In software development, a piece of code used in more than one application.
- Making use of standard I/O facilities.
- Using (with modification, perhaps) parts of specification, design or code of one application to build another application.
- The process of creating potentially reusable components.

Why Reuse?



Reusable Components Video

http://www.softdevtube.com/2014/10/02/reusable-components/

LEEDS BECKETT UNIVERSITY

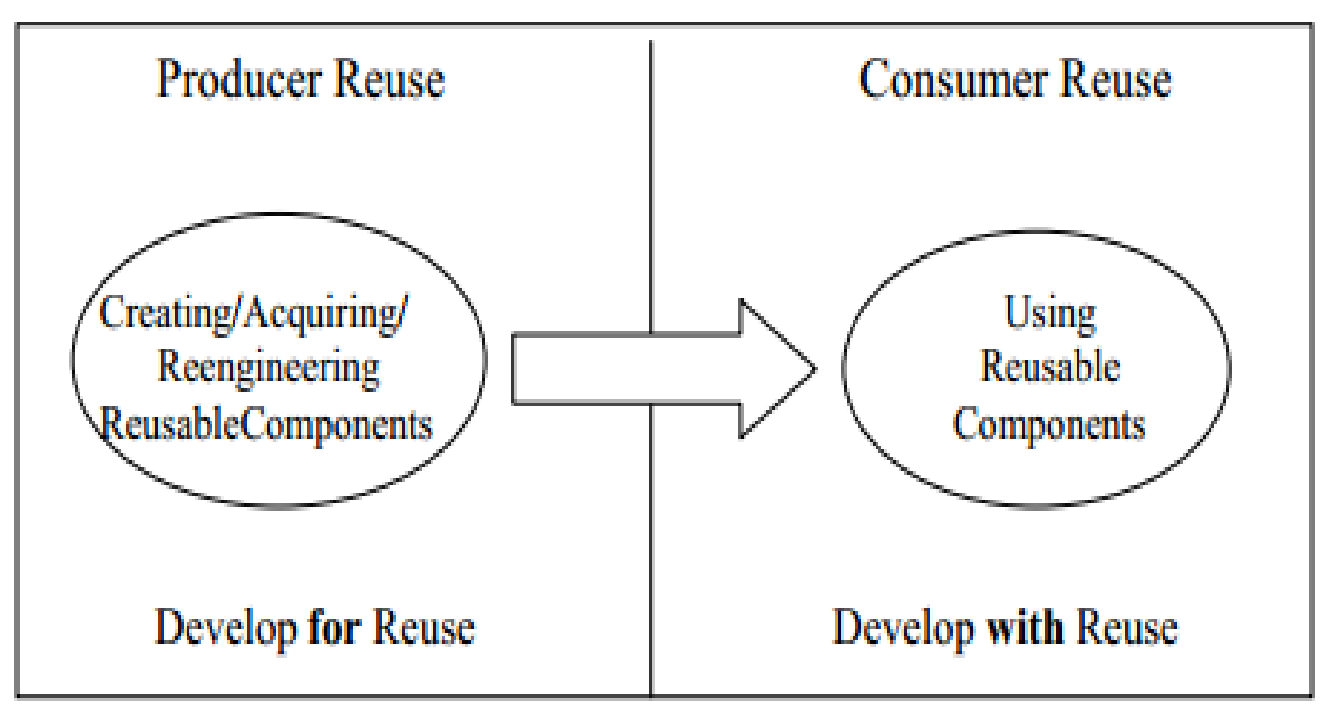
What are reusable?

Seven common types of reusable artifacts:

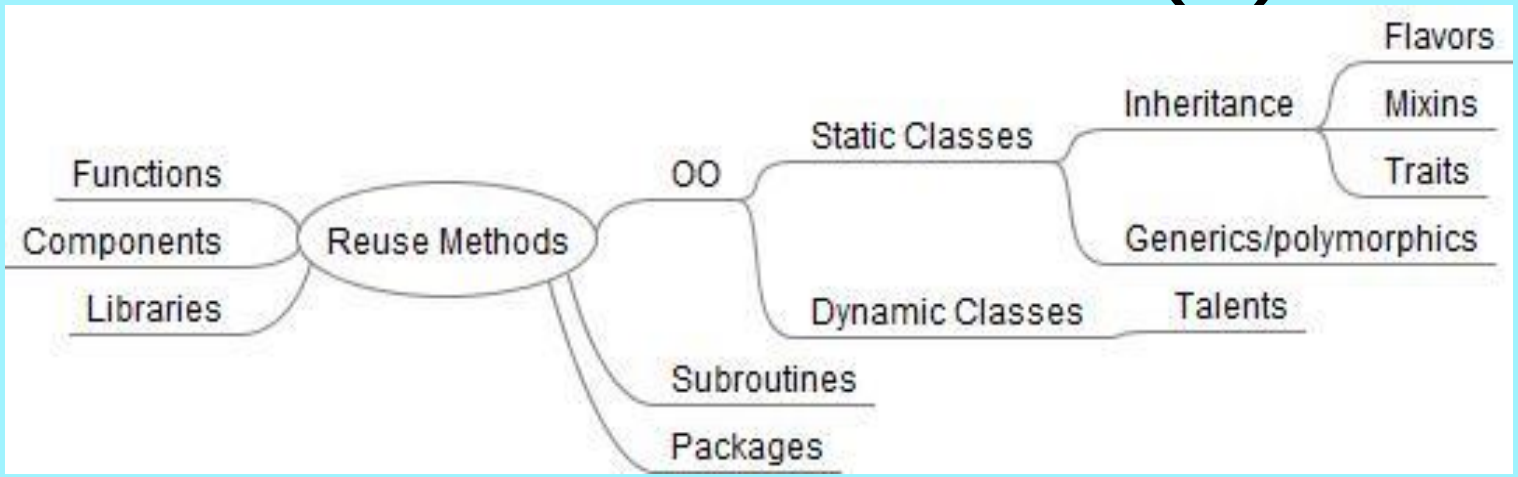
1. Executable Code
2. Source Code
3. Requirements Specifications
4. Designs, Patterns
5. Test Data
6. Documentation
7. Architectures (Layered, Enterprise Arch, SOA)
8. API
9. Web services and cloud services

Others: Knowledge, Experience, Skills, Tacit knowledge, Ideas, etc

Reuse Methods (1)



Reuse Methods (2)



Ressia, J. et al (2014) **Talents**: an environment for dynamically composing units of reuse, Software Practice & Experience,44:413–432

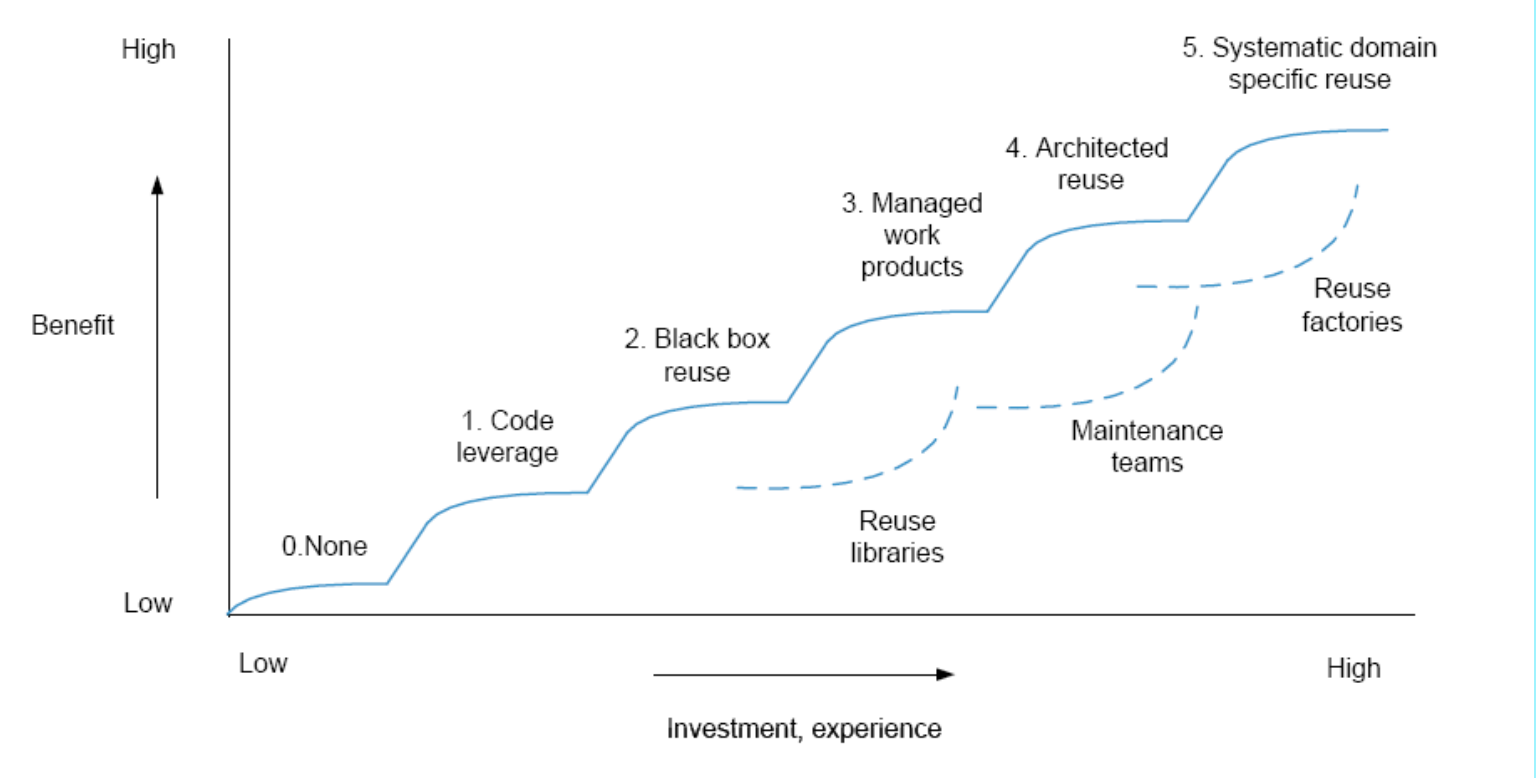
Reuse with Multiple Inheritance (MI)

Mixins has been introduced as an extension for the Lisp programming language being small incomplete implementations of classes that could be “mixed in” at arbitrary places in the class hierarchy.

https://courses.cs.ut.ee/MTAT.03.271/2012_fall/uploads/Main/UrmasTamm.pdf

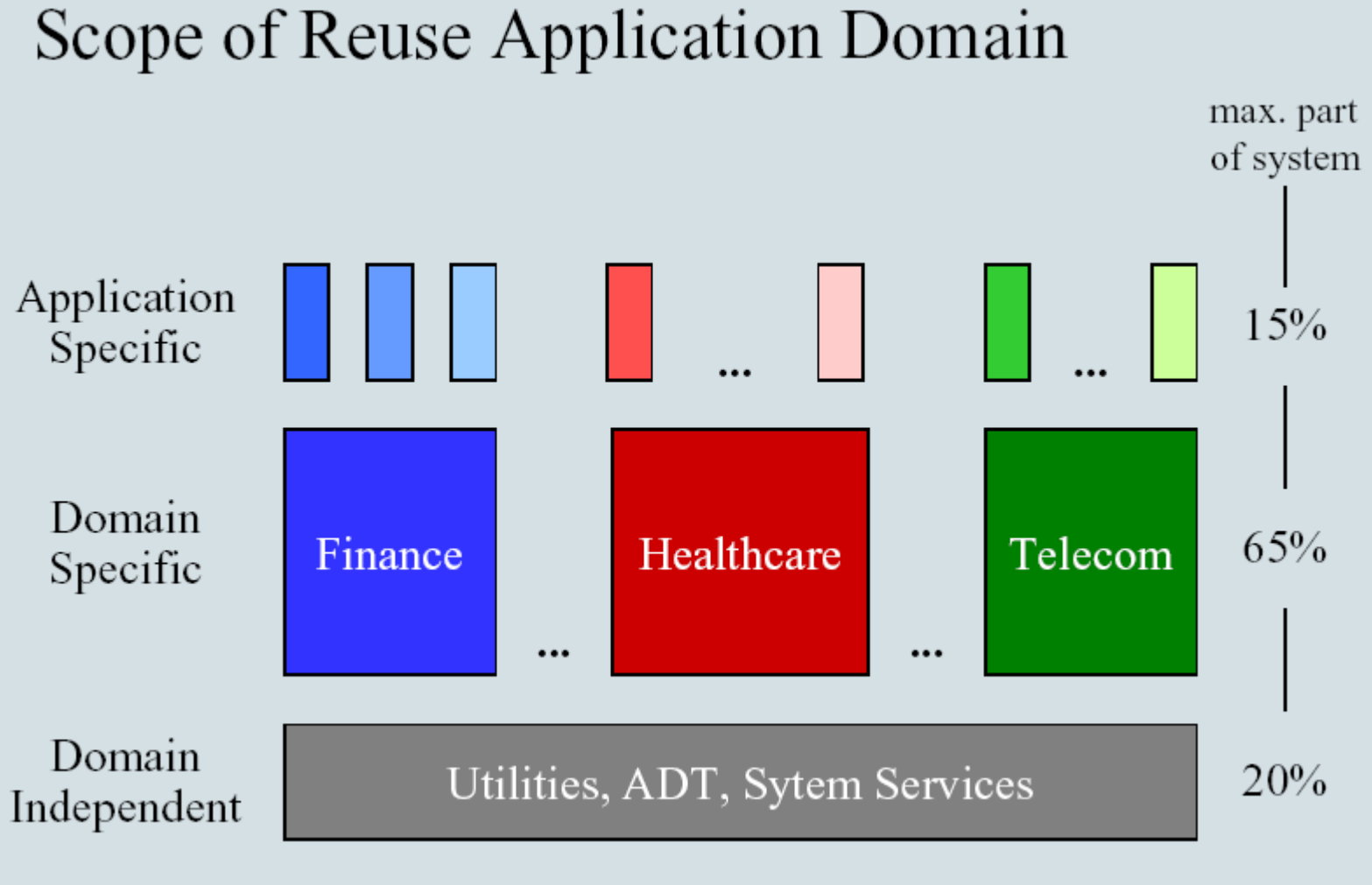
Traits are purely units of reuse that provide a way for unrelated classes to share code

Incremental Stage of Reuse: Gaining Experience towards large Scale Reuse



This reuse level is defined as level 0. The first real reuse level presents a form of code-leverage, where pieces of code are made available and can be reused by multiple parties. The pieces of code are made available through the use of a reuse library, providing a central place where the components are stored.

Scope for reusing application domains

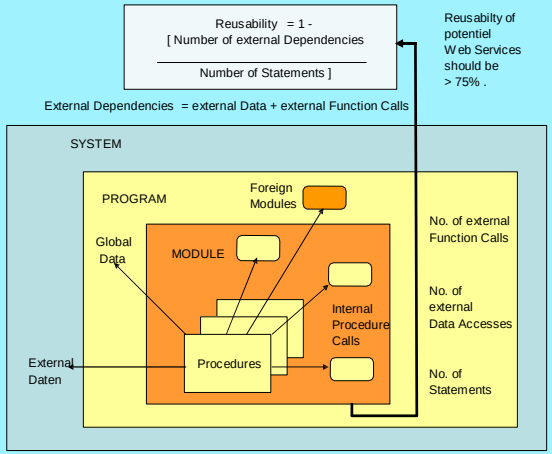


Reuse Metrics

- Reusability = 1 - [Number of external Dependencies/Number of Statements]
- External Dependencies = external Data + external Function Calls

Reusabilty of potential components/Web Services should be > 75% .

$$\text{Reuse \%} = \frac{\text{Reused Software}}{\text{Total Software}} \times 100\%$$



Evaluating Reusability of Code

A better metric that relates reuse to productivity is the notion of *Reuse Leverage for Productivity* (RL). The relative productivity of an organizational unit (project, department or company) that does not practice reuse is set at 1. If the productivity of the organization increases after the introduction of reuse processes, then the reuse leverage is greater than 1.

$$\text{Reuse Leverage for } \textit{Productivity} = \frac{\text{Productivity with Reuse}}{\text{Productivity without Reuse}} \times 100\%$$

LEEDS BECKETT UNIVERSITY

Current Examples

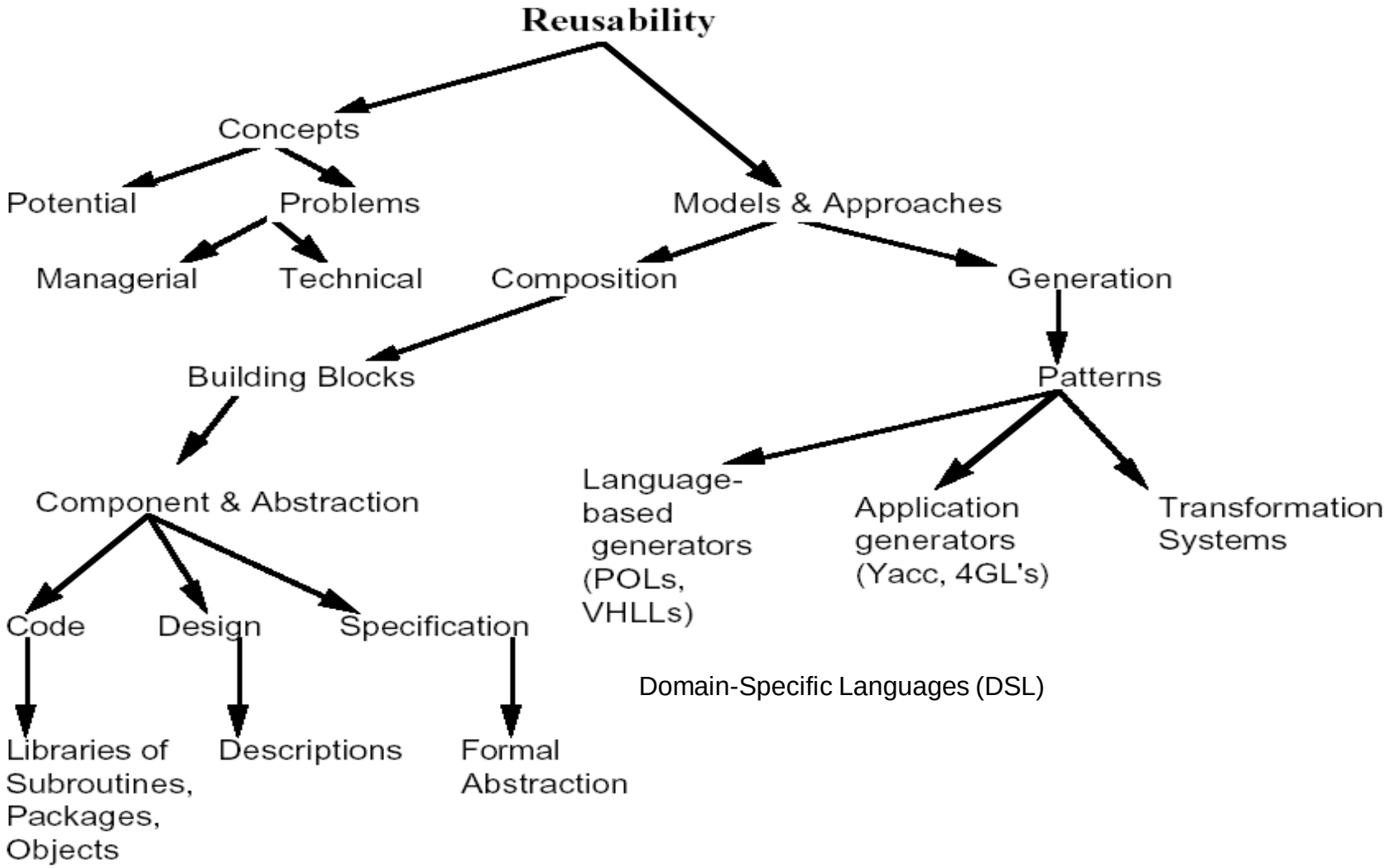
- HP 57% productivity increase has been reported
- Raytheon Missile Systems 60% reuse rate and 50% increase in productivity and their data service division provides 3200 COBOL source modules for reuse
- Japanese Software factory
- Philips developed a component based model (Koala) which has been widely used in their consumer electronics products
- Toshiba
- NASA has reported reduction of 75% in overall development effort and cost
- GTE Data Services (220 reusable components consisting of 960K LOC of COBOL, C and assembler available to 2000 developers
- European Space Agency
- Existing Technologies: Microsoft COM/COM+/.NET, CORBA, EJB, SOA, Web Services

Reuse Videos

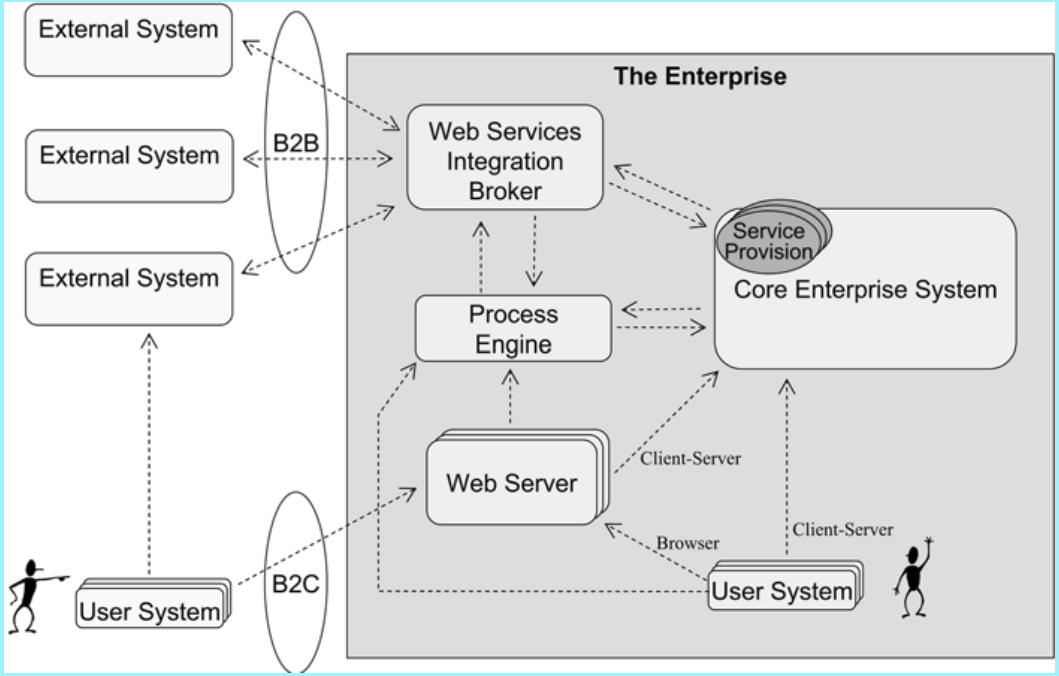
- http://videos.findtarget.com/videos/software_reuse/
  http://videos.findtarget.com/videos/off_the_shelf_software/
  http://videos.findtarget.com/videos/programming_language/
  http://videos.findtarget.com/videos/software_projects/
  http://www.videojug.com/
  http://videos.findtarget.com/

LEEDS BECKETT UNIVERSITY

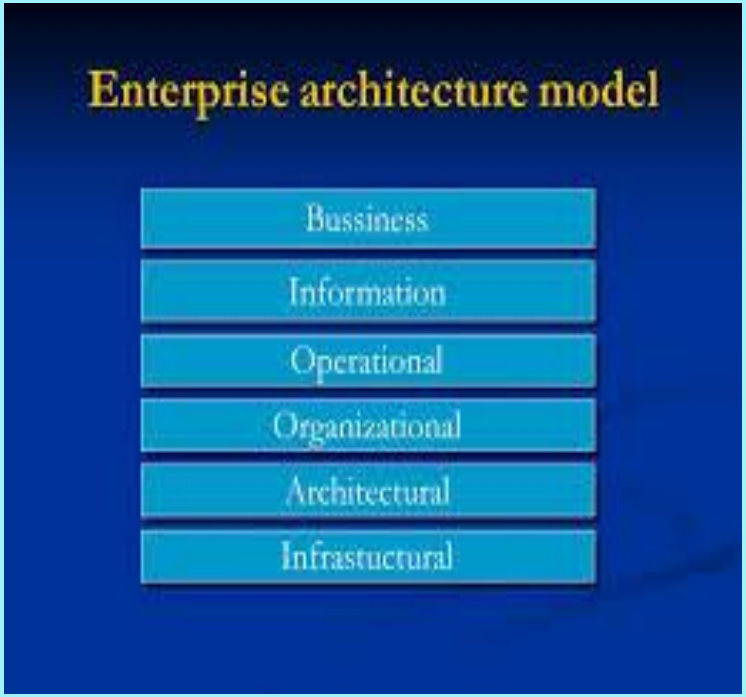
Research Directions/Reuse Attributes



Why SOA & Enterprise Architecture?



An IT View of the Enterprise and Its Partners

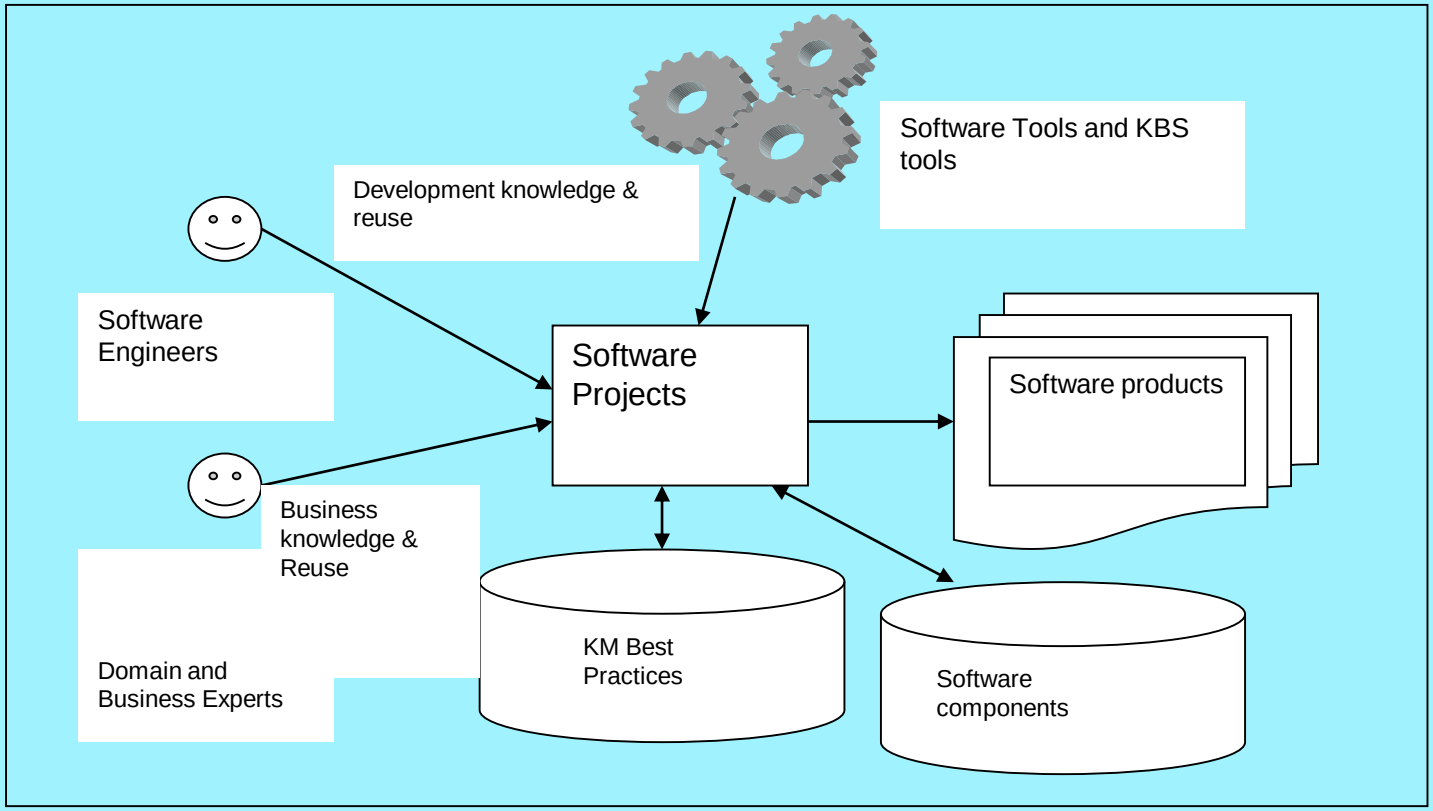


SOA is a formal way of integrating businesses, existing applications, legacy systems into an enterprise architecture and also to a cloud

Enterprise Architecture Frameworks

- A framework provides a generic problem space and a common vocabulary within which individuals can operate to solve specific problems. Thus, there are many frameworks or architecture models , e.g.:
- Zachman ’s Framework [5, 6] :This is a widely used approach for developing enterprise-wise IS architectures and is considered as a reference model against which other frameworks can map themselves.
- • RM-ODP [23– 26] : This uses a well-understood object-modelling technique (OMT) and is developed by highly reputable agencies such as ISO and International Telecommunications Unit.
- • TOGAF [7] :This is an industry standard generic framework and is freely available.
- • C4ISR /DoDAF [7 , 27, 28] :These are frameworks developed mainly for the use of US Department of Defense.
- For a comparison and review, refer to [29– 32] .

Generic Reuse Technology

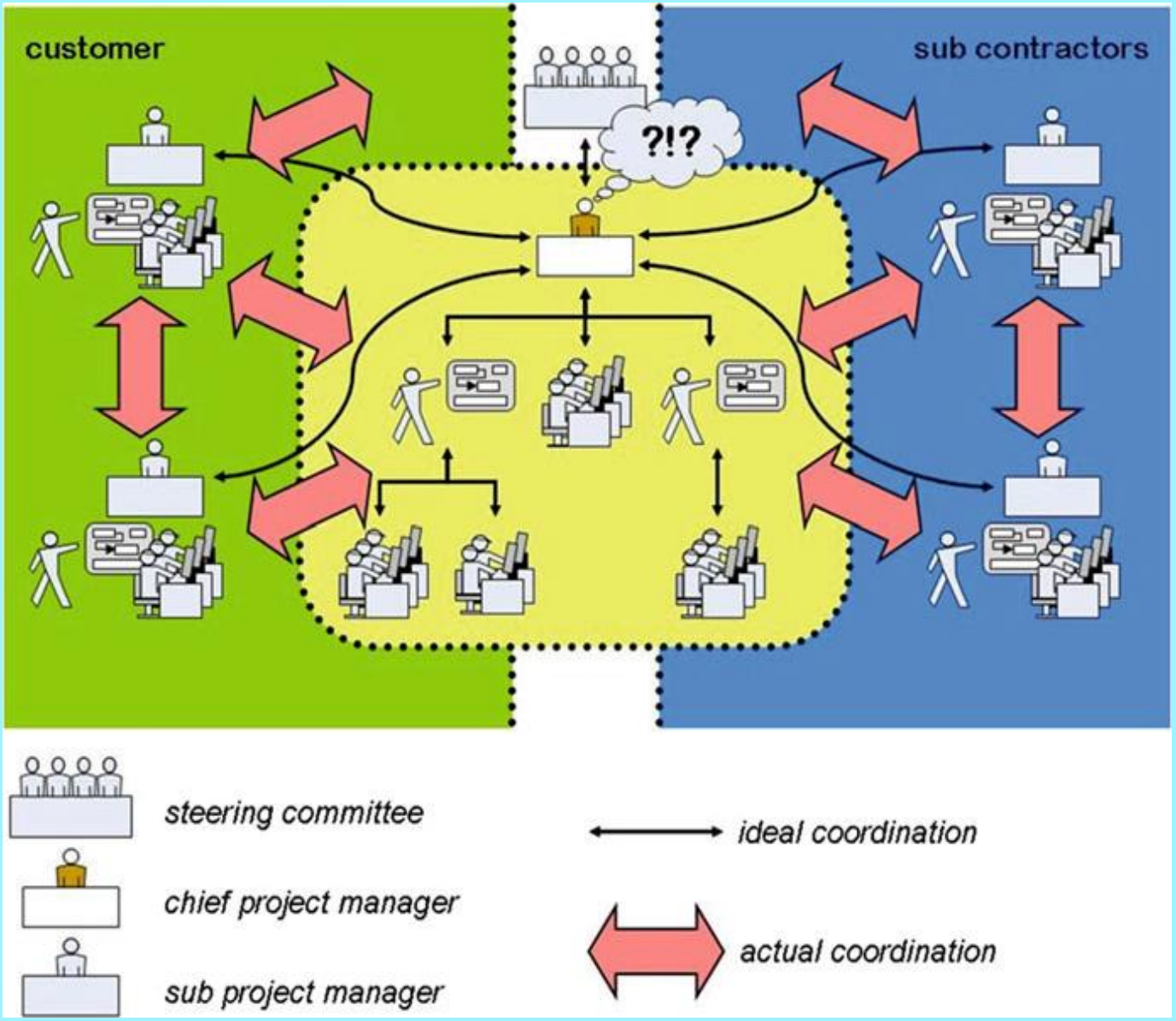


Reuse Challenges

- Integration
  - iWay System Integration
  - www.youtube.com/watch?v=XDv6SUXus4U&feature=related
  - http://www.iwaysoftware.com/
- Managing complexity
- Large scale reuse
- Emerging technologies (.NET/EJB, AoP, SOA, SaaS, Web services) and applications (cloud computing, middleware & distributed systems, mobile devices, wearables, games & entertainment systems)

Complexity of Large Scale Projects: Organisational Structure

What is the role of a chief program mer concept?



Key Points

- Software reuse isn't just reusing code
- Design for reuse is the basic design principle embedded in current developments such as Objects, Patterns, Frameworks, Architectures, Aspects, SOA, Web Services, etc
- Key design principles include abstraction levels, separation of concerns, design views, managing complexity and adaptability, extensibility, integration and composition (product line – reusing whole systems)

Tutorial Exercises

- Identify and define constructs that support reuse explicitly in C#, Java, EJB, C/C++, Ada 95/05 or any of your favourite programming languages
- Identify a set of guidelines for introducing software reuse into your own project
- Identify and improve a code for reuse which was developed by you previously
- Identify and provide rationale for a set of well known reuse techniques
- Identify and provide a documenting structure for a reusable code for other users
- Research into reuse success & failure stories
- Research & Discuss Technical Savings vs Technical Debt
- http://effectivesoftwaredesign.com/2013/10/18/avoiding-technical-debt-how-to-accumulate-technical-savings/#!
- Online Quiz on Component-Level Design Chapter 10, SE by Pressman
- http://highered.mheducation.com/sites/0073375977/student_view0/chapter10/multiple_choice_quiz.html
- Another Quiz on Reuse
- Multimedia waterfall model quiz

Lab Sessions 1-2

- Install Visual Paradigm and NetBeans with UML Plug-in. Compare ease of use.
- Identify a set of requirements for Qbay system
- Draw a set of use case models for Qbay system
- Identify a set of classes
- Identify and model a set of software components with appropriate interfaces

References

- Ramachandran, M (2008), Software components: Guidelines and Applications, Nova Publishers, New York, https://www.novapublishers.com/catalog/product_info.php?products_id=7577
- Krueger, C (1992) Software Reuse, ACM Surveys, Vol. 24, No. 2, June 1992.
- Oshri, I., Newell, S., and Pan, S. L. (2007) Implementing component reuse strategy in complex products environments, CACM, December, Vol. 50 No. 12
- Heineman, G. T. and Councill, W. T (2001) Component-Based Software Engineering, Addison Wesley
- Szyperski, C . Gruntz, D., and Murer, S (2002) Component Software; beyond OO, Addison Wesley, 2002
- Sommervile, I (2007) Software Engineering, Addison Wesley, 8th Edition, 2007.
- Pressman, R (2006) Software Engineering: A Practitioner's Approach, 5th Edition, McGraw Hill
- CBSE current trends, http://www.deeperweb.com/business-research/management/component-based-software-engineering-research-trends-surveys.html
- Ramachandran, M (2010) Handbook of research in SE, IGI Global, http://www.igi-global.com/bookstore/titledetails.aspx?titleid=505
- Ramachandran, M (2011) KESDLC, IGI Global, http://www.igi-global.com/bookstore/titledetails.aspx?titleid=46170
- SEI (2000) Technical aspects of CBSE, http://www.sei.cmu.edu/reports/00tr008.pdf
- SEI (2000) Market assessment of CBSE, http://www.sei.cmu.edu/reports/01tn007.pdf
- Crnkovic and Larsson (2000) Component-Based Software Engineering – New Paradigm of Software Development, http://www.mrtc.mdh.se/publications/0293.pdf
- Reuse and CBSE, http://www.win.tue.nl/~mchaudro/cbse2007/Managing%20CBSE%20and%20Reuse.pdf
- Ramachandran, M (2011) Software security engineering: Design and applications, https://www.novapublishers.com/catalog/product_info.php?products_id=26331
- **Programming with software components**
- Lowy, J (203) Programming .NET components, O'Reilly
- Monson-Haefel, R (2001) Enterprise JavaBeans, O'Reilly

LEEDS BECKETT UNIVERSITY