Poseidon House
Castle Park
Cambridge CB3 0RD
United Kingdom

TELEPHONE: **Cambridge (01223) 515010**
INTERNATIONAL: **+44 1223 515010**
FAX: **+44 1223 359779**
E-MAIL: **apm@ansa.co.uk**

Training

ANSAwise - Architecture for Open Distributed Systems

Chris Mayers

Abstract

Organizations cannot develop IT systems and services as fast as they would like.

One approach to tackling this problem is to practice software reuse.

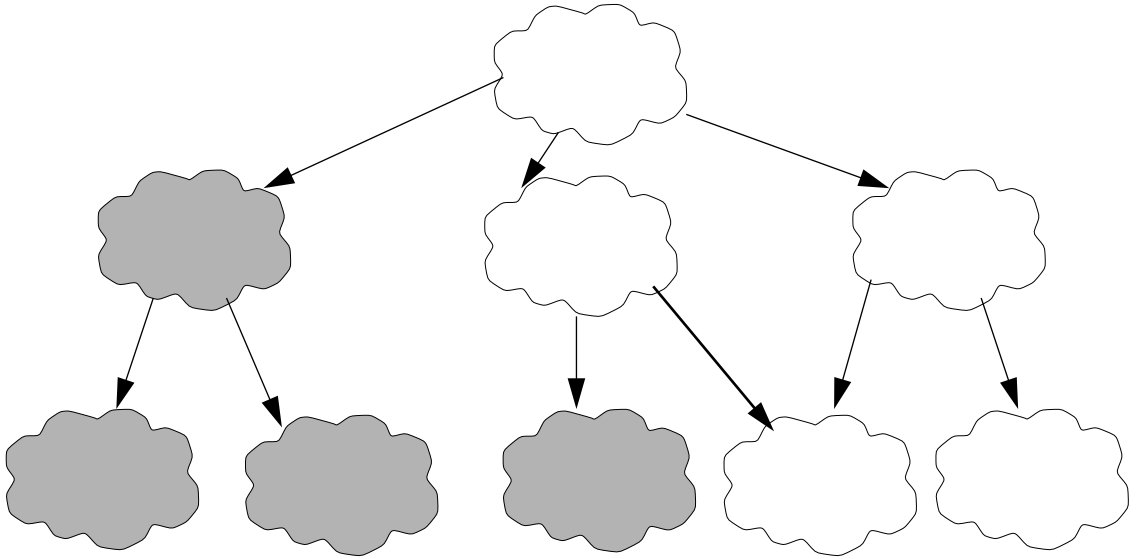
The solution being offered is the use of an architecture for distributed systems, enabling reuse within a controlling framework of models, problem-solving techniques, and templates.

This module of the ANSAwise training programme explains what an architecture for open distributed systems is. It explains how distributed systems architectures relate to other IT architectures (for example network architectures), and gives pointers towards specific architectures (ODP, CORBA, DCE) covered in other modules

APM.1323.01 **Approved** 24th October 1994
Briefing Note

Distribution:
Supersedes:
Superseded by:

Architecture for Open Distributed Systems



In this session

- *Explain what an architecture for open distributed systems is*
- *Explain the benefits of such an architecture*
- *Explain how such an architecture relates to other IT architectures*

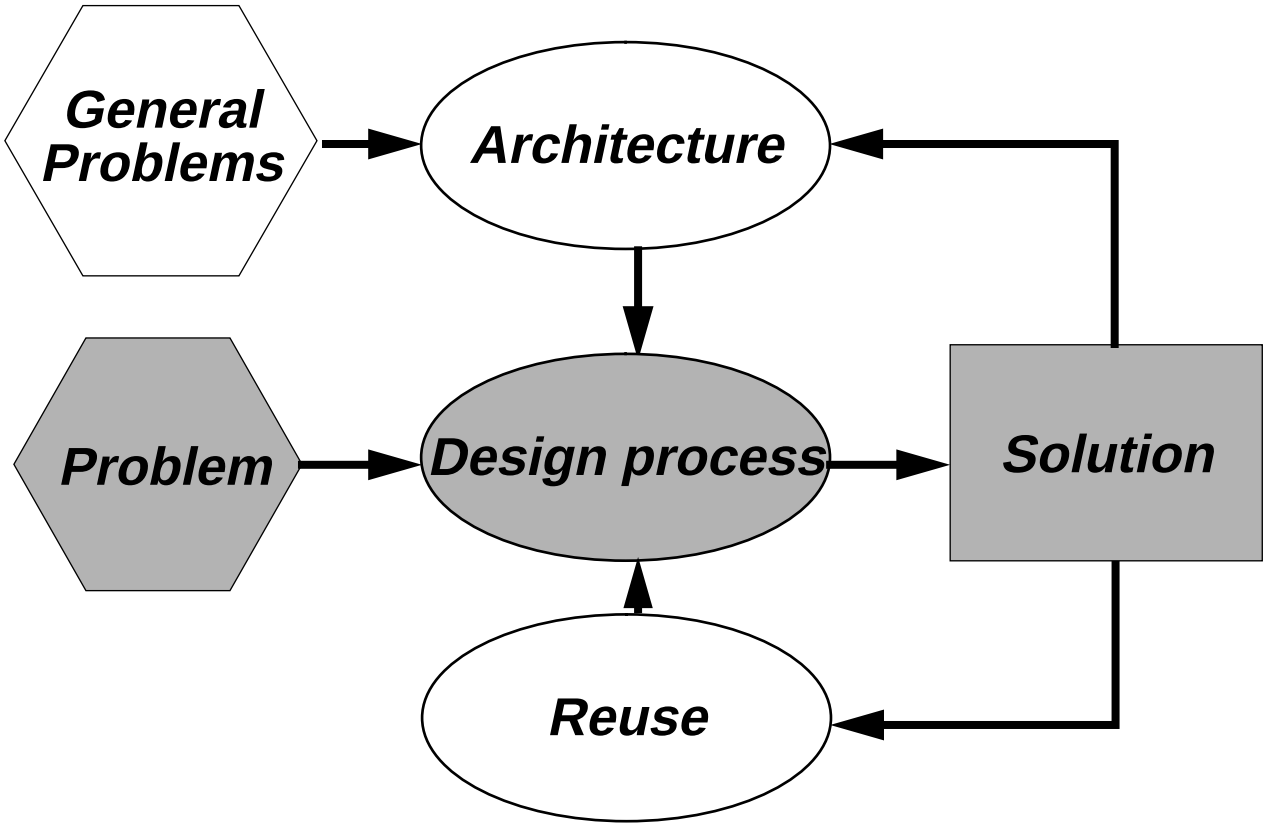
Systems are complex

- *We are dealing with complex problems*
- *We are applying complex solutions*
- *The development process is complex*
  - we are dealing with changing problems
  - we expect flexible solutions

More problems of complexity

- *Too much detail*
- *Human limitations*
- *Critical timing*
- *Problems can change faster than solutions*

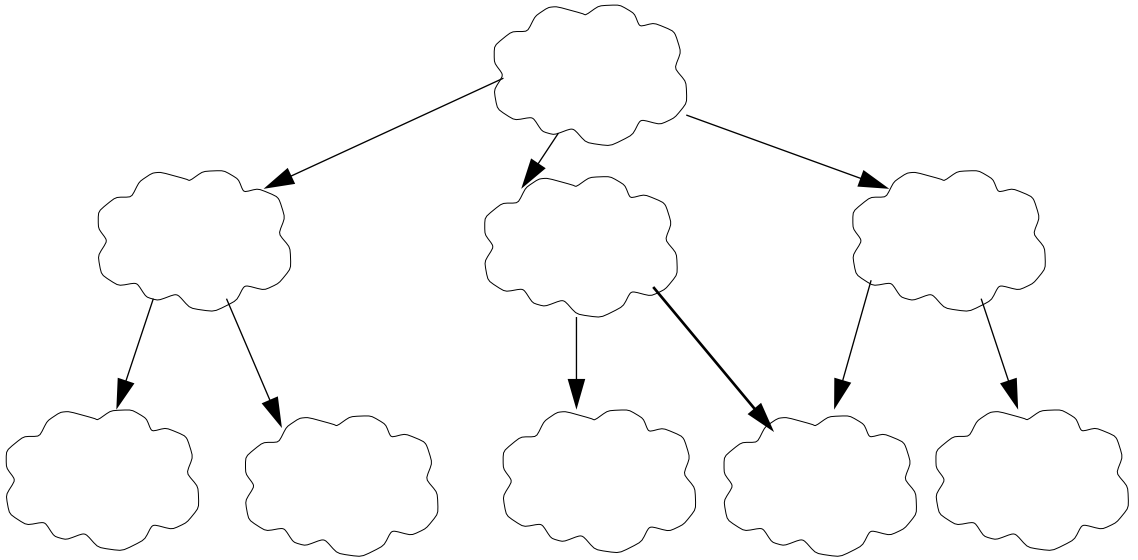
The design process



Solving complex problems

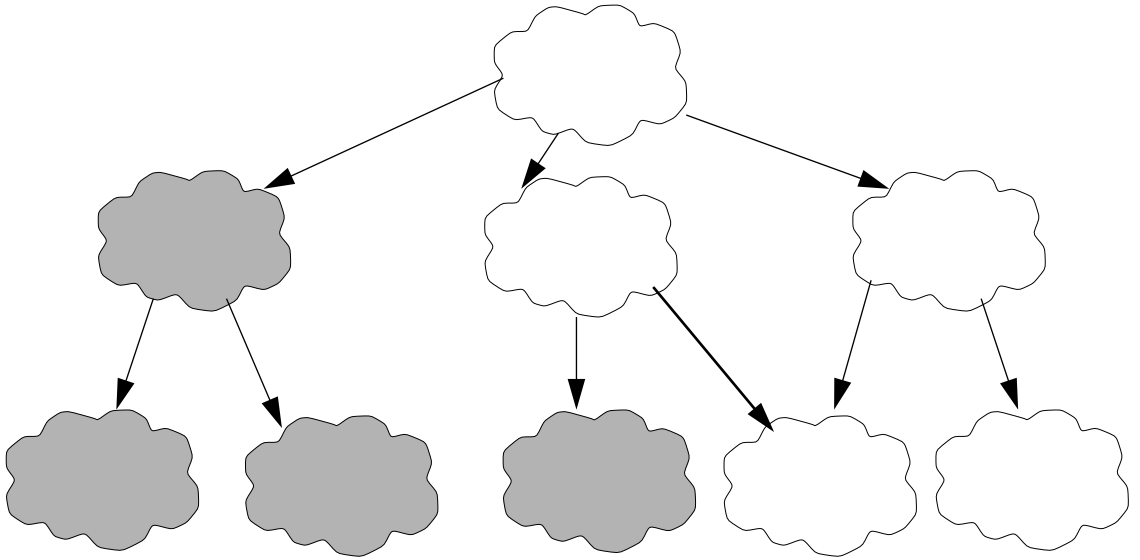
- *Techniques for solving problems include*
  - Decompositition
  - Abstraction
  - Modelling
  - Generalization
  - Evolution

Decomposition - splitting the problem up



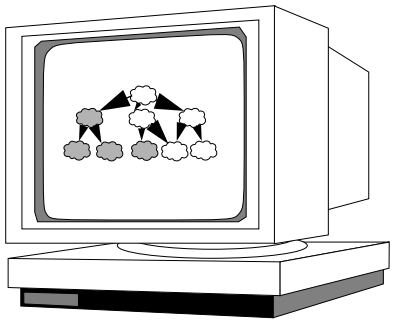
- *Also known as ‘divide-and-conquer’*
- *Use for separation of concerns, as well as for structure*
- *Decomposition allows you to give the problem to someone else*

Abstraction - suppressing detail

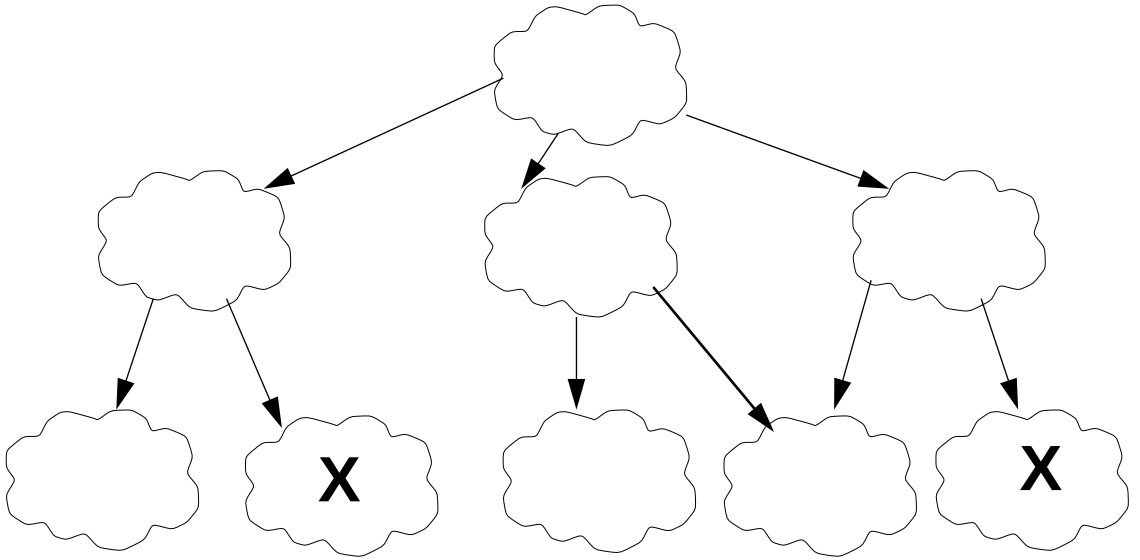


Modelling - communication via models

- ***Models allow shared understanding***

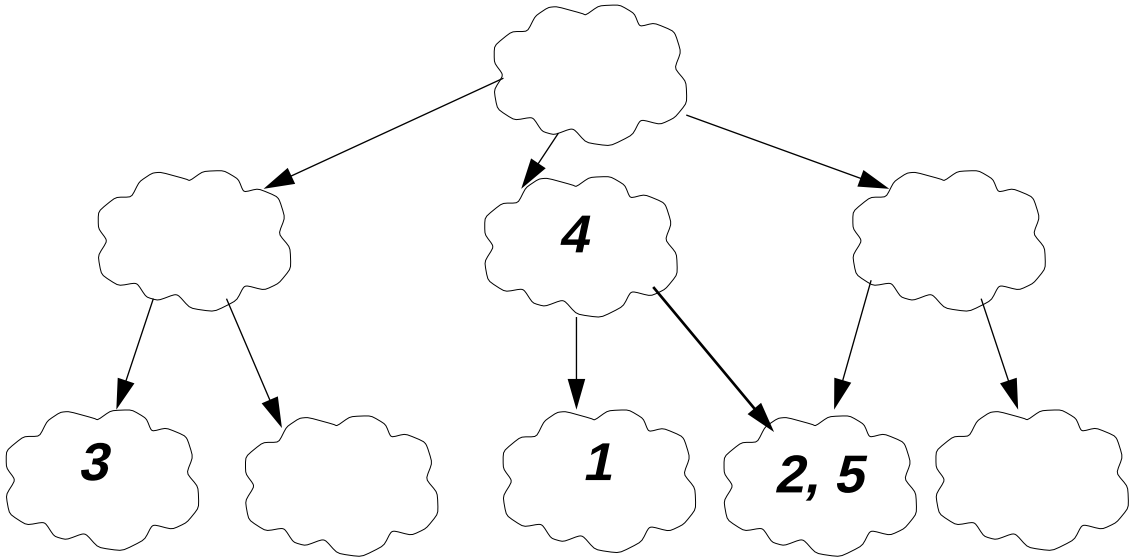


Generalization - using a solution more than once



- ***The solutions may be similar, not identical***

Evolution - solving the problem gradually

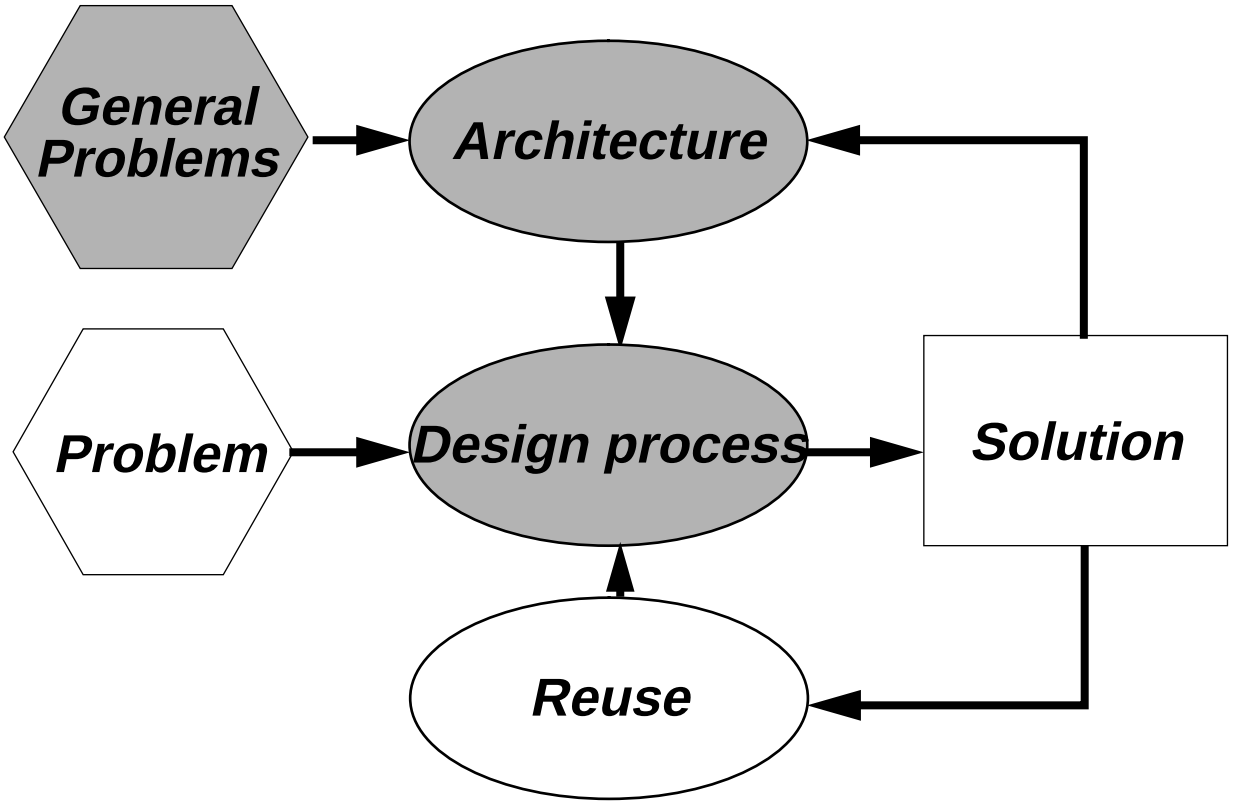


- *This also allows you to cope with changing problems*

“We use these techniques already”

- ***The real benefits only come when these techniques are applied***
  - **consistently**
  - **universally**

The design process



Learning from others

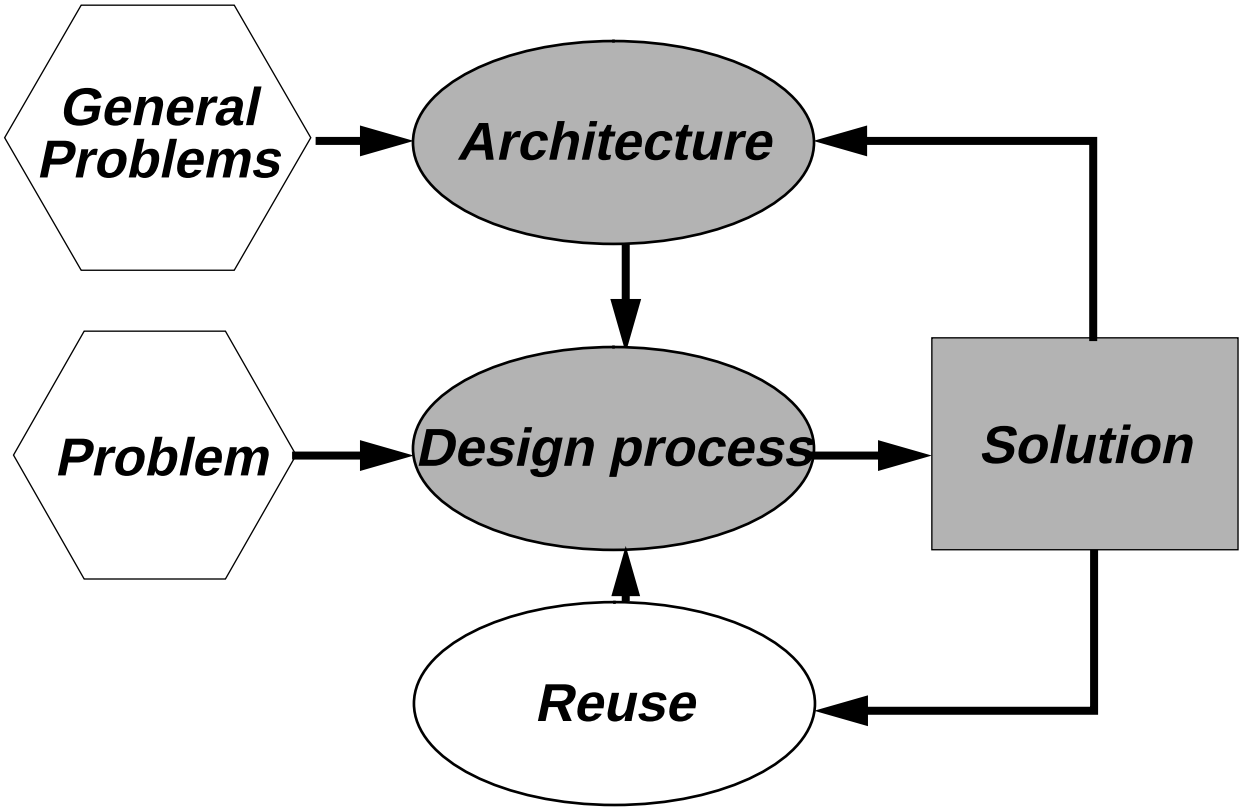
- ***We don't solve problems from scratch***
  - **we use our own previous knowledge...**
  - **...and that of experts**
- ***This knowledge is codified in***
  - **existing components (building blocks)**
  - **rules and standards**
  - **guidelines**
- ***Architecture is the framework for capturing and using this knowledge***

Architecture - an expert's view

- *"Computer architecture, like other architectures, is the art of determining the needs of the user of a structure and then designing to meet those need as effectively as possible within economic constraints."*

  - **Fred Brooks**

The design process



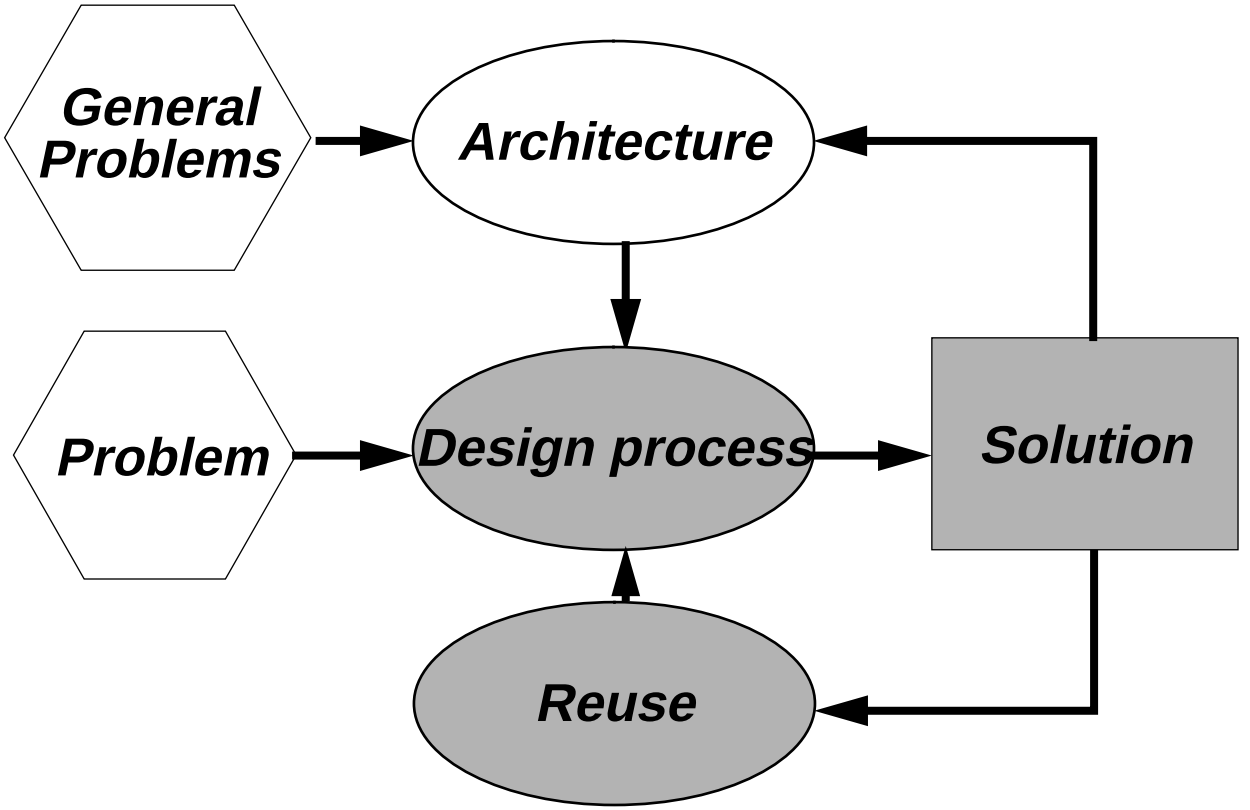
Architecture for controlling complexity

- ***The problem-solving techniques are powerful, but insufficient***
  - **the main challenge is maintaining conceptual design integrity**
- ***Architects must understand***
  - **the business needs (the problem at hand)**
  - **the field of application (the problem domain)**
  - **the limitations of existing systems, custom, and practice**
  - **the possible classes of solution, and the risks and costs involved**
- ***Architects must provide***
  - **a simple framework for solutions by implementors**
  - **an explicit framework (it may already exist, but its implications must be made explicit)**

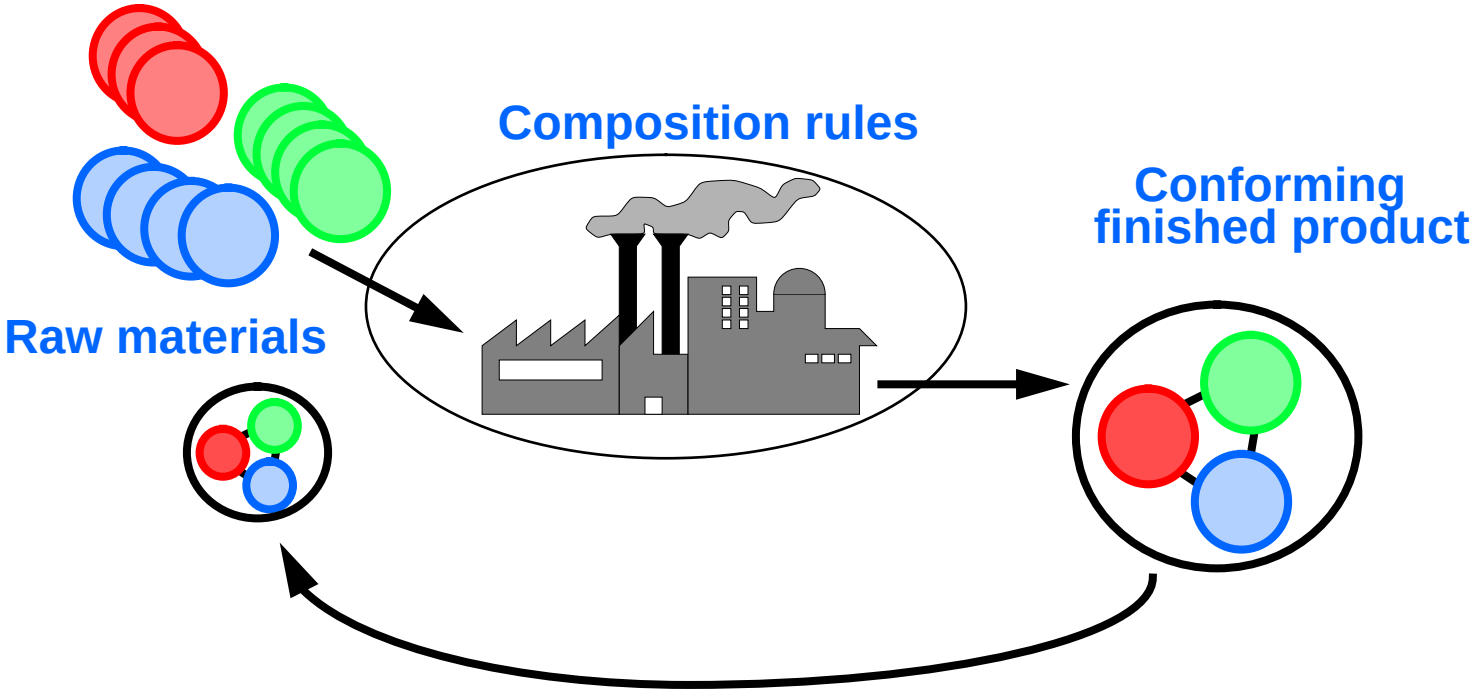
Architecture as a framework for reuse

- ***The biggest savings are by reusing...***
  - **...the most complex components**
  - **...the components that are needed most often**
- ***Reusing complex components requires a strategic framework***
  - **they cannot be 'bent to fit'; they must be designed with reuse in mind...**
  - **... hence the need for rules, standards, and guidelines**

The design process



Architecture for reuse

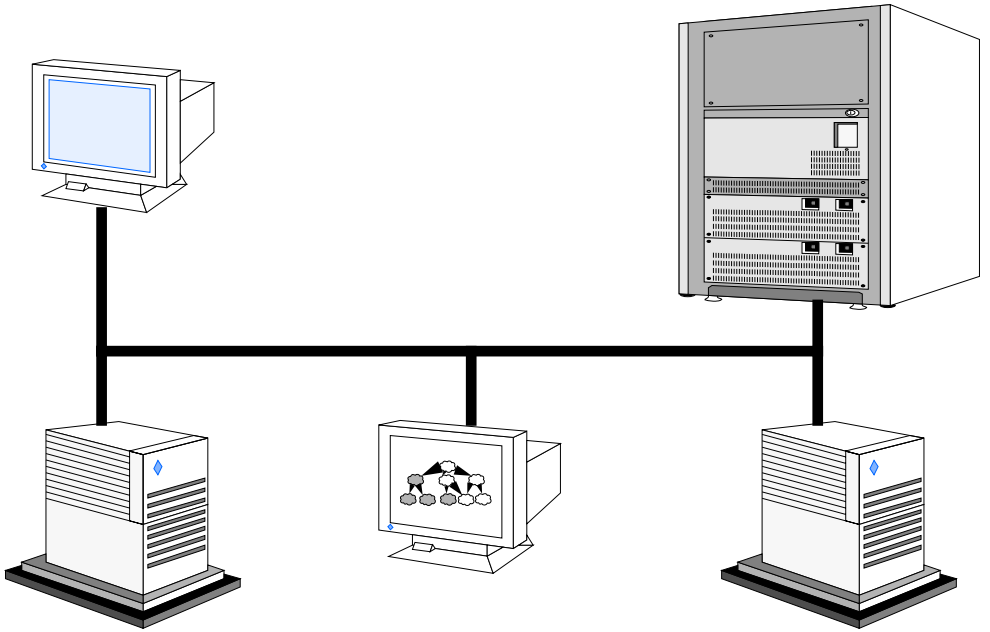


Architecture and reuse

- ***Reuse requires the components themselves***
  - **components that are built or bought**
- ***Efficient reuse requires tools***
  - **to repackage, reconfigure, recombine,...**

Self-describing systems and reuse

- ***A system that contains a description of itself is easier to evolve...***



- ***... the tools can operate on the real system***

Architecture - the aims

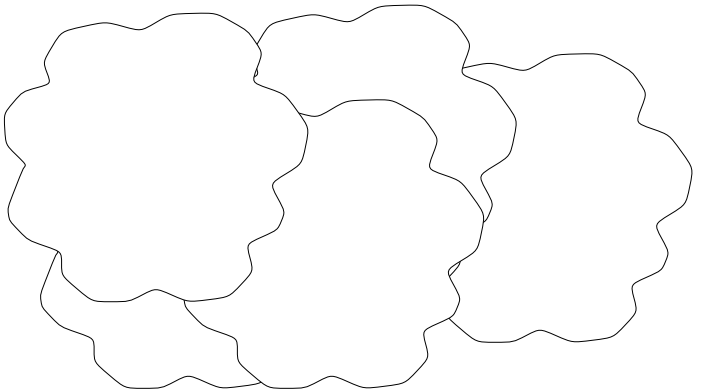
- ***The system should survive longer than any of its component parts***
  - **long-term evolution coping with planned (and unplanned) change**
- ***This means a strong focus on interoperability***
  - **allowing independent evolution of the parts**

Distributed systems and other IT architectures

- *Other IT disciplines have their architectures*
  - Hardware architecture
  - Operating system architecture
  - Network architecture
  - Management architecture
  - Security architecture
  - Application architecture
  - User interface architecture

Distributed systems and the IT architectures

- ***Some of their concerns intersect...***



  - **... this is discussed when we cover the specific topics**

Distributed systems and IT architectures

- ***Hardware architecture***
  - **no intersection**
- ***Operating system architecture***
  - **distributed systems make demands on operating systems**
- ***Network architecture***
  - **distributed systems make demands on network architecture**

Distributed systems and IS architecture

- ***Management architecture***
  - **distributed systems make demands on management architecture, and vice-versa**
- ***Security architecture***
  - **distributed systems make demands on security architecture and vice-versa**
- ***Application architecture***
  - **distributed systems recommend a particular partitioning**
- ***User interface architecture***
  - **no significant intersection**

Distributed systems architectures

- *Although having simple aims, architectures are themselves complex*
  - much work has already been done
  - many organizations have been involved
- *Architectures are a framework for existing standards*
  - addressing different problems in different ways

Standards relevant to distributed systems architecture

- *There is a very wide range of standards-making bodies*
  - National and international bodies
  - Industry bodies
  - Individual organizations

ISO/ITU-T (previously CCITT)

- *Examples of “de jure” standards*
  - OSI TP service and protocol
  - OSI Management information model and protocols (GDMO, CMIS/CMIP)
  - OSI Remote Data Access, SQL
  - Open Systems Environment (POSIX)
  - Basic Reference Model for Open Distributed Processing (RM-ODP)

OSF, OMG, and X/Open

- *Examples of industry consensus standards*
  - OSF Distributed Computing Environment, Distributed Management Environment
  - OMG Combined Object Request Broker Architecture (CORBA) Object Services
  - X/Open XA

Microsoft

- ***Examples of industry "de facto" standards***
  - **Microsoft Object Linking and Embedding (OLE)**
  - **Microsoft Windows NT**

Summary

- ***Architecture is the solution to systems complexity, and the key to reuse***
  - **it allows judgement on designs, based on experience and best practice**
- ***The available distributed systems architectures are incomplete***
- ***For more on architecture in system design***
  - **on architecture for distributed systems, see *An Overview of ANSA* (AR.000)**
  - **on architecture in system design, see *The Mythical Man-Month* by Fred Brooks (Addison-Wesley)**