Poseidon House
Castle Park
Cambridge CB3 0RD
United Kingdom

TELEPHONE: **Cambridge (0223) 323010**
INTERNATIONAL: **+44 223 323010**
FAX: **+44 223 359779**
E-MAIL: **apm@ansa.co.uk**

ANSA Phase III

2nd Feature Interaction Workshop slides (Keynote speech)

Rob van der Linden

Abstract

This document contains the slides for the keynote speech at the second day of the 2nd International Workshop on Feature Interaction in Telecommunications Systems, 9-10 May, 1994, Amsterdam, The Netherlands.

APM.1208.00.01 **Draft** 6 May 1994

External Paper

Distribution:
Supersedes:
Superseded by:

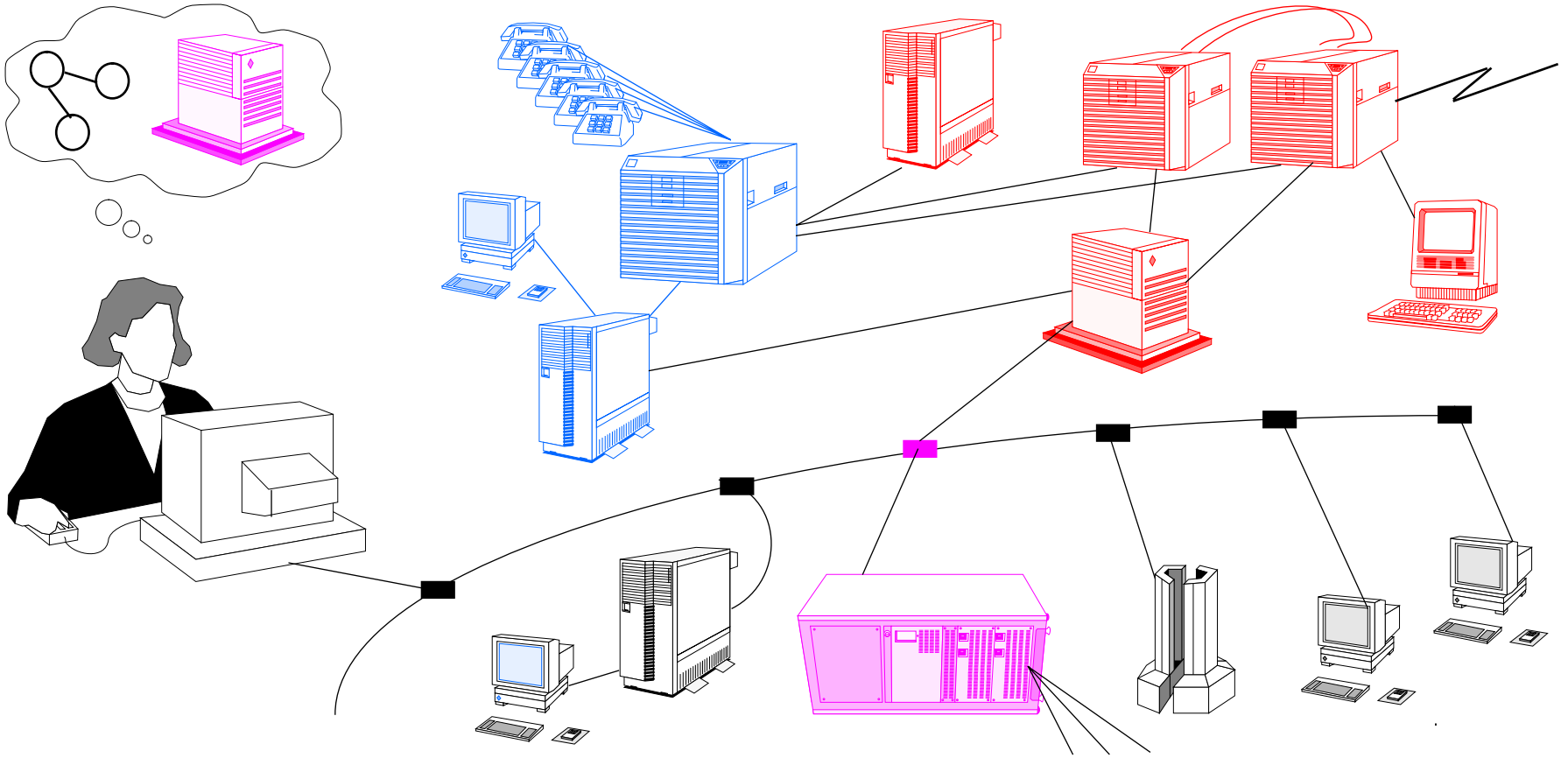
Using an Architecture to Help Beat Feature Interaction

Rob van der Linden

ANSA Project Manager

rvdl@ansa.co.uk

Background



Main Themes

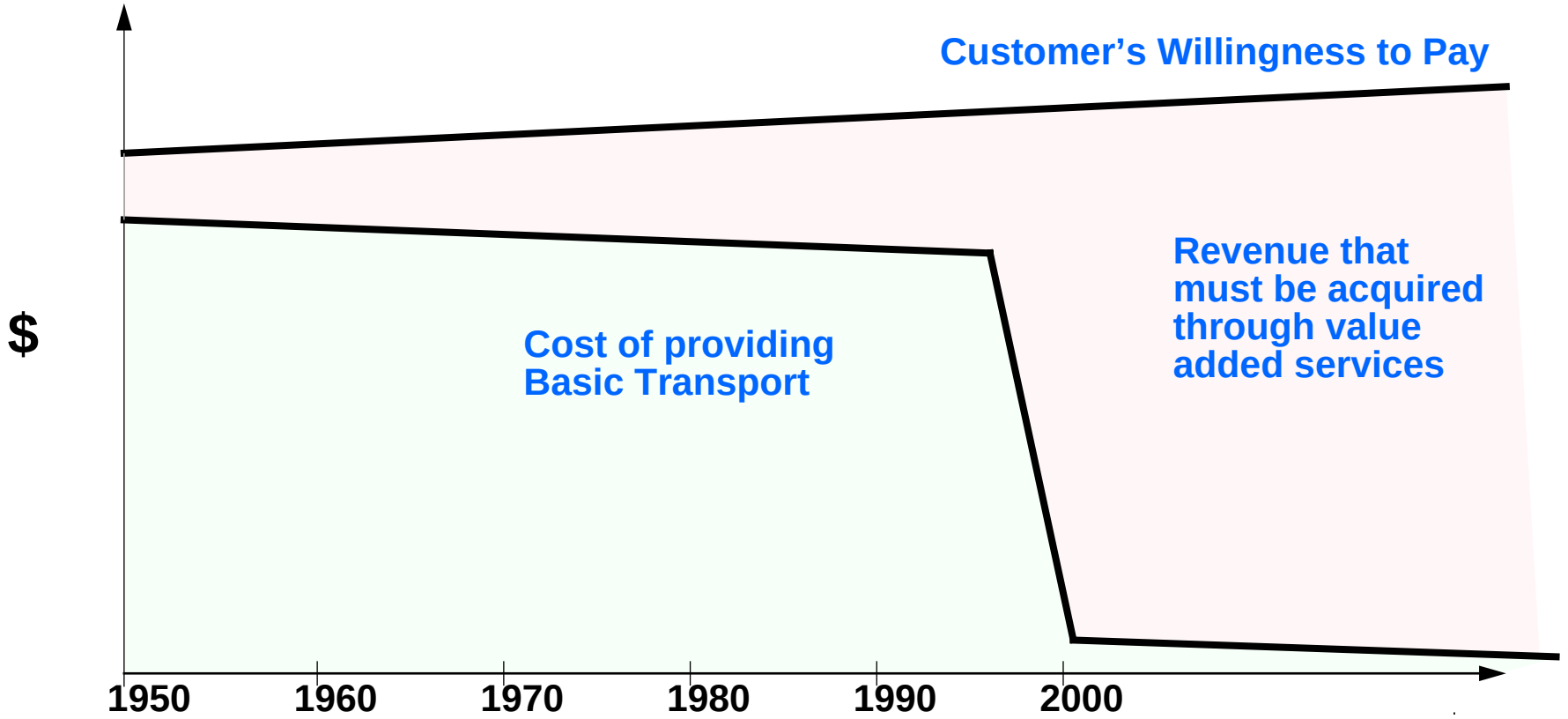
- **separation and substitutability**
- **available and accessible descriptions**

Meet the customer's needs

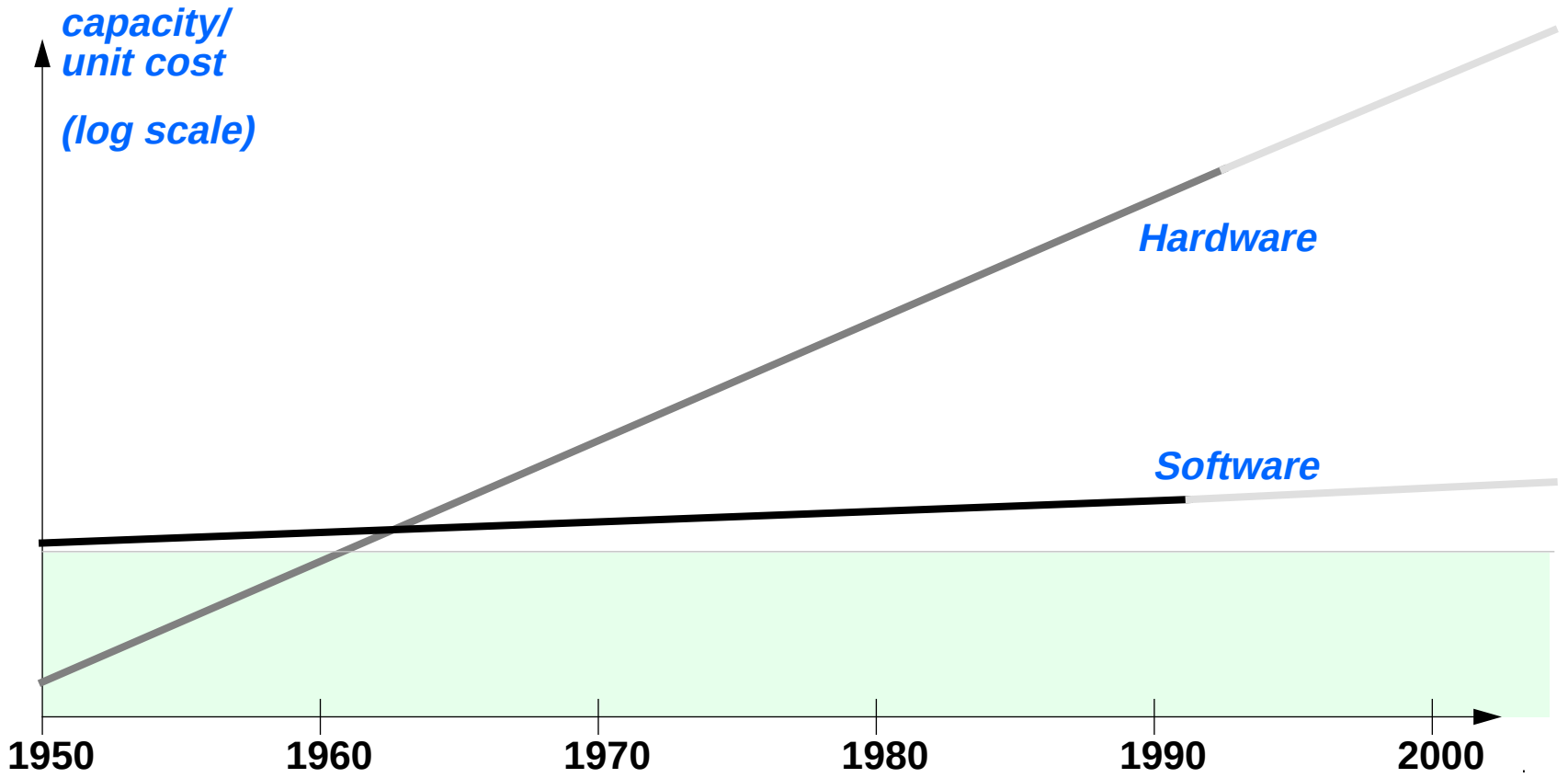
- **timely:** I want it NOW! (well, yesterday)
- **personalised:** I want it just the way I like it!
- **competitive:** I don't want to pay over the odds!
- **dependably predictable:** It should be there when I need it!
- **integratable:** Why can't I connect it to my PC?

. . . . . before your competitor does

The business challenge



Software cost

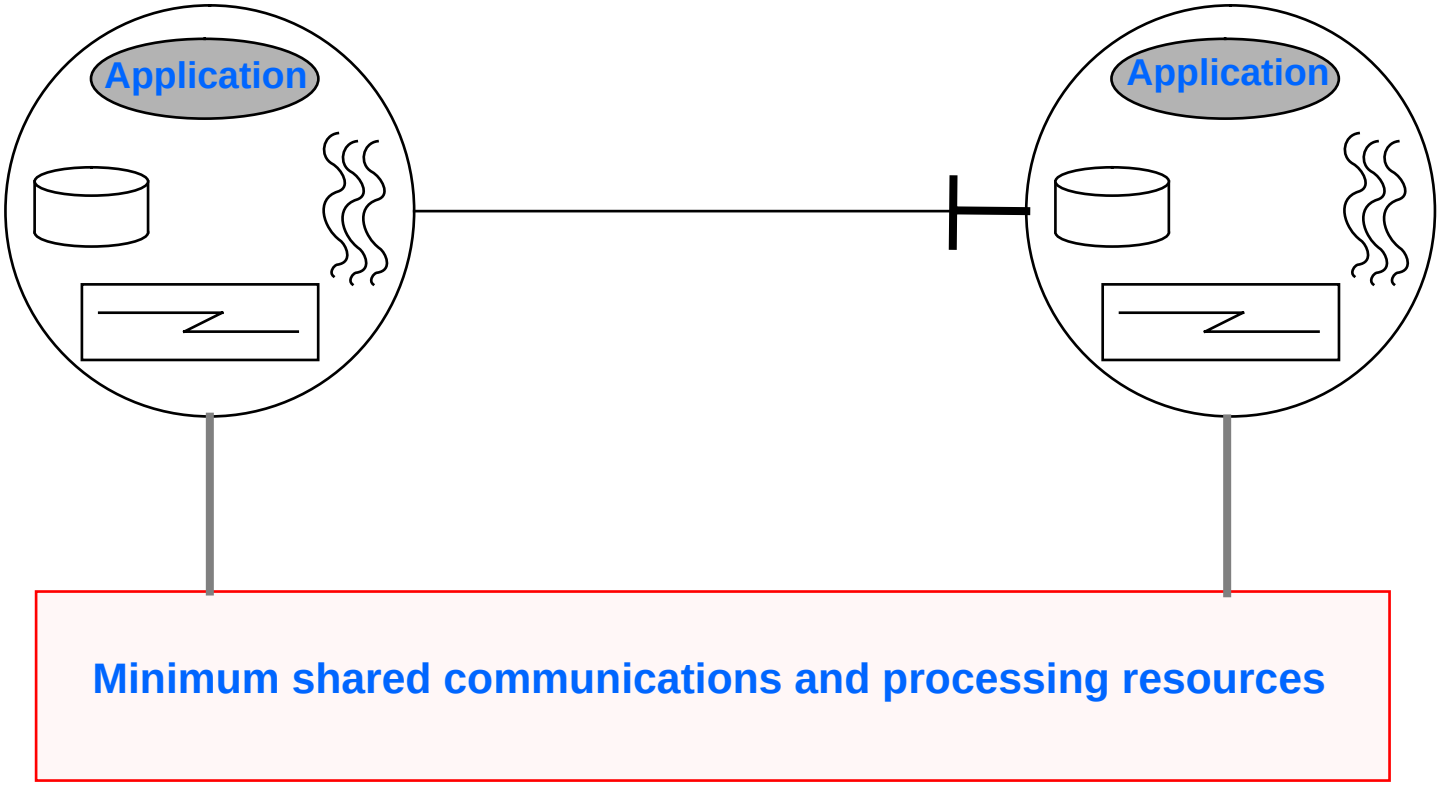


A new problem?

- **project management**
- **requirement analysis**
- **analysis and design methodologies**
- **informal and formal approaches**

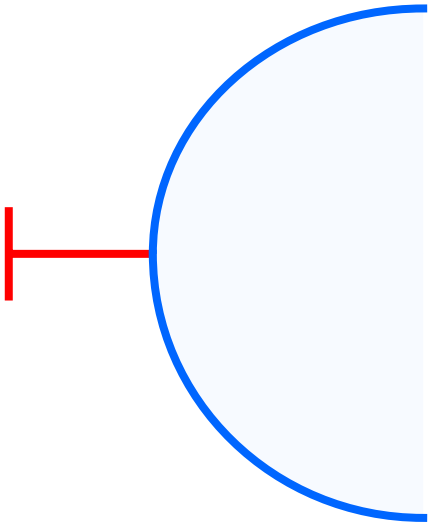
No, but none of these SCALE very well

System structure: separation



System structure: Substitutability (1)

- **operational Interfaces**



Interface {
operation$_1$Name (< typed parameter >)
-> outcome$_1$Name (< typed result >)
:
-> outcome$_n$Name (< typed result >);
:
operation$_n$Name (< typed parameter >)
-> outcome$_1$Name (< typed result >)
:
-> outcome$_n$Name (< typed result >)
};

System structure: Substitutability (2)

- **stream Interfaces**

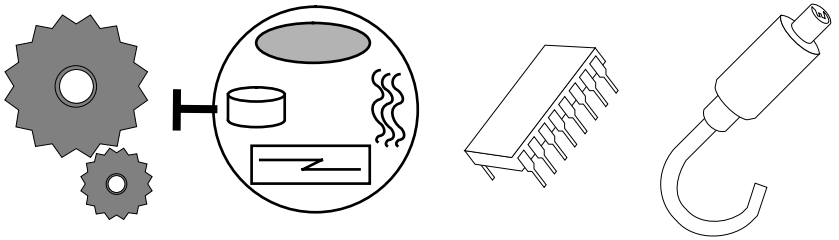
  - **for computationally unstructured streams of information**

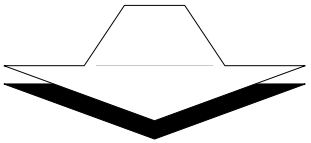
    audio, video, telemetry

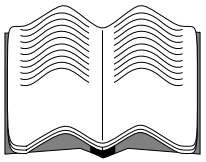
  - **for information streams with “irrelevant” structures**

Architecture

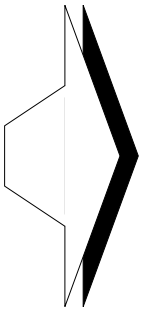
Basic building blocks

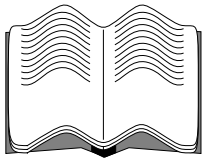




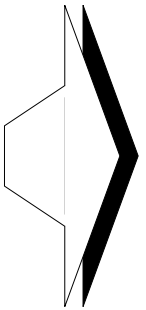


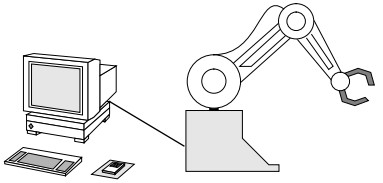
Combination rules



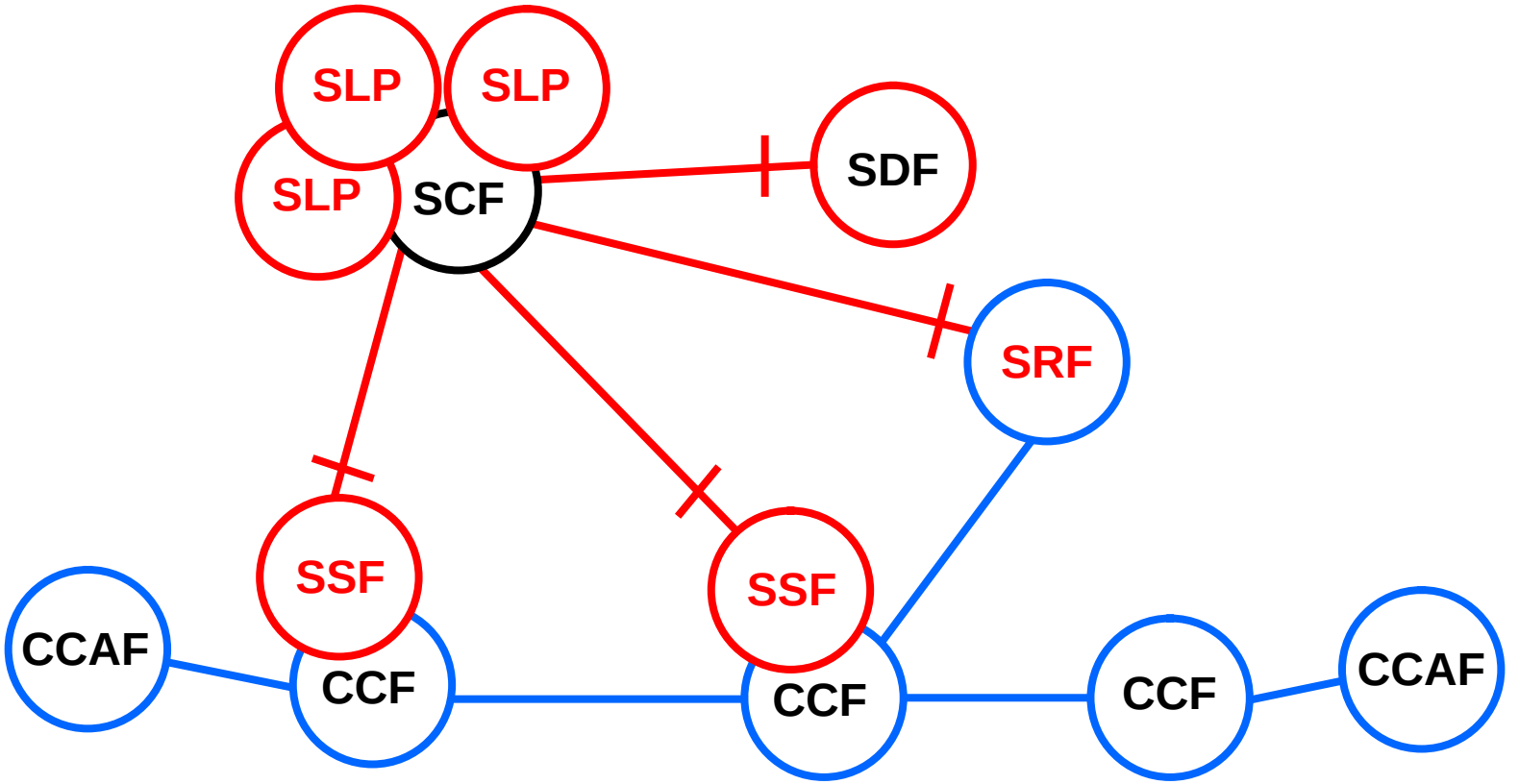


Recipes





IN CS-1

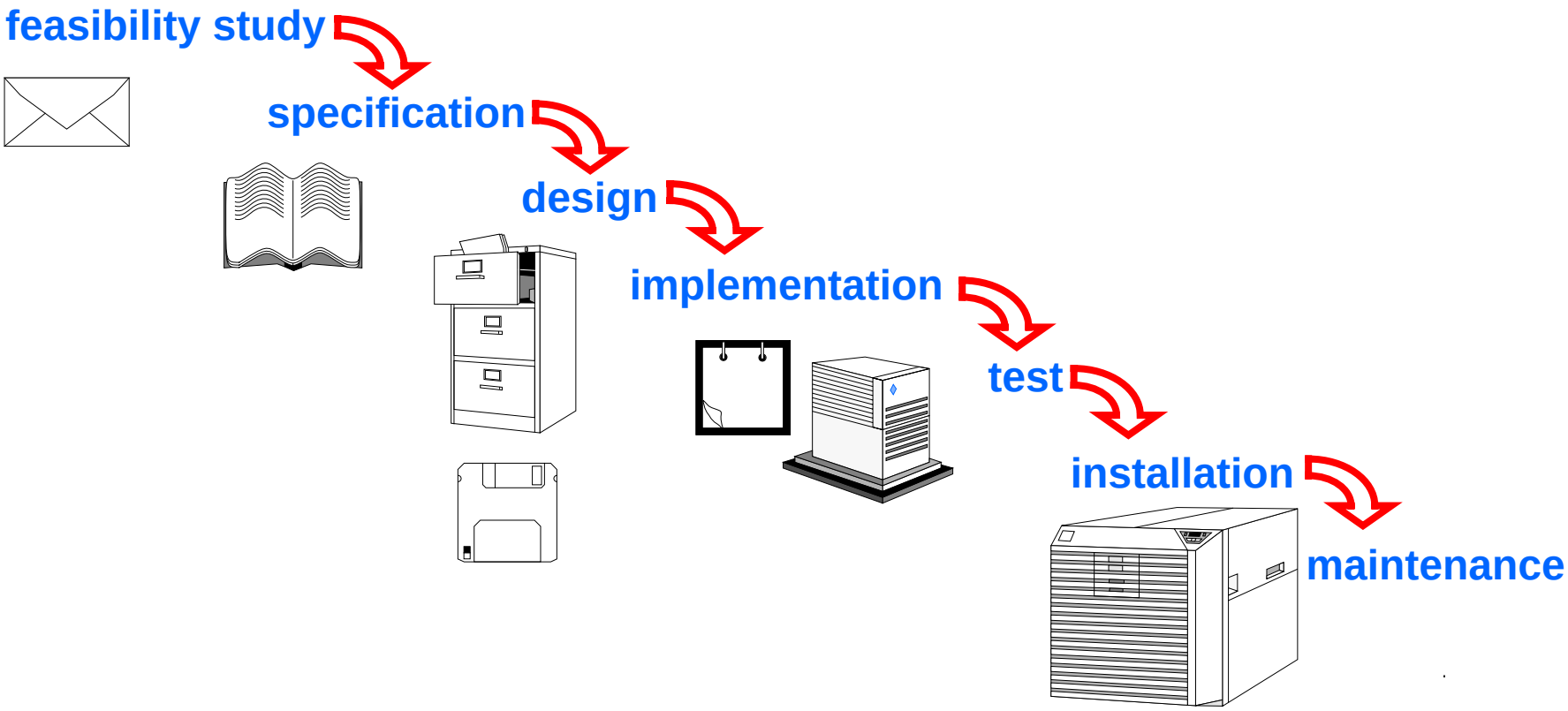


Open Software Architecture

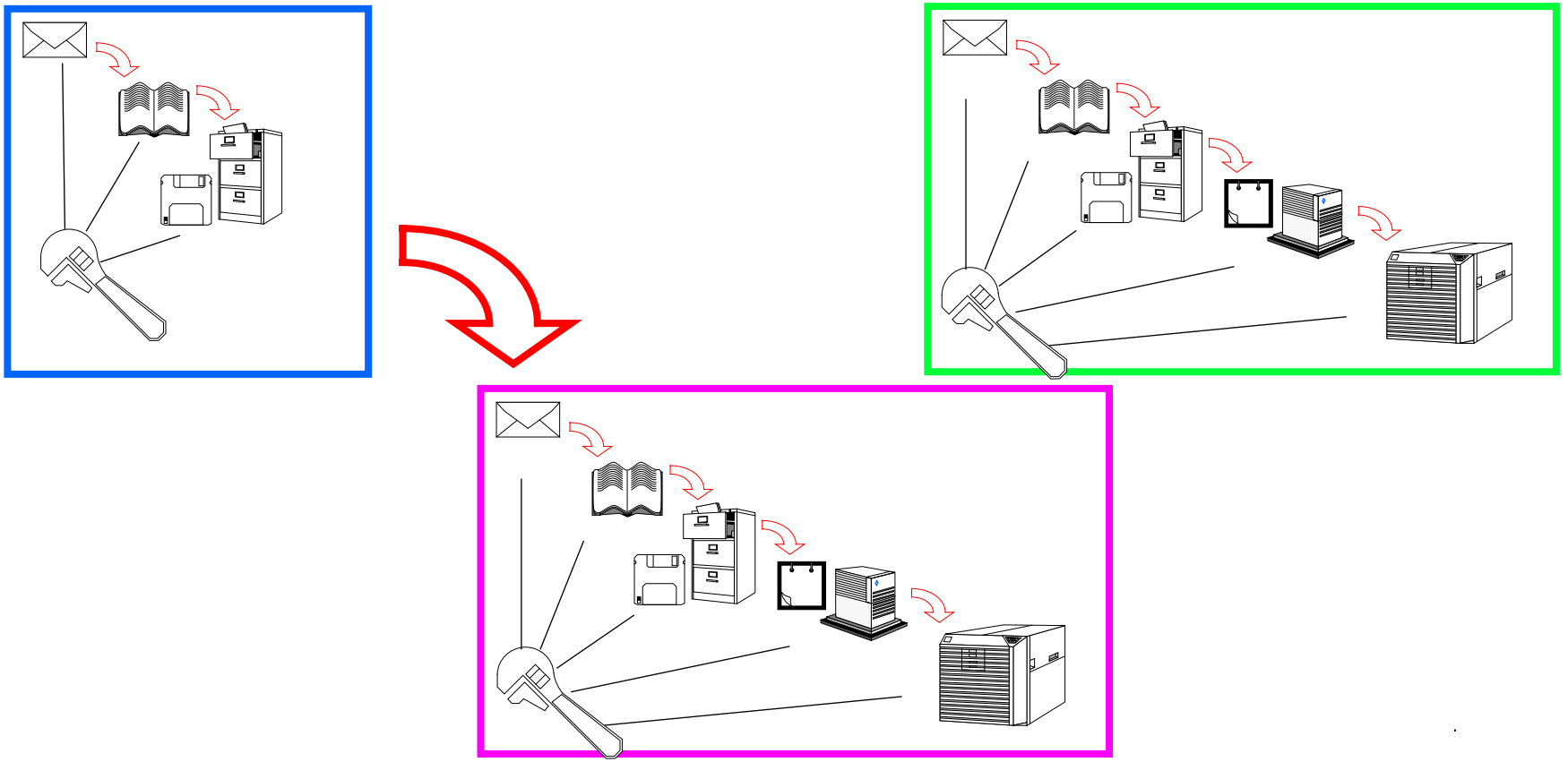
- to build effective useful software we must store information about:
  - how to perform the required functions
  - how the various parts relate to one another
- open systems must carry their own model so they can evolve, therefore
  - specifications of components must be visible from the system
  - implementations of components must be available to the system

. . . . . but how should this information be structured?

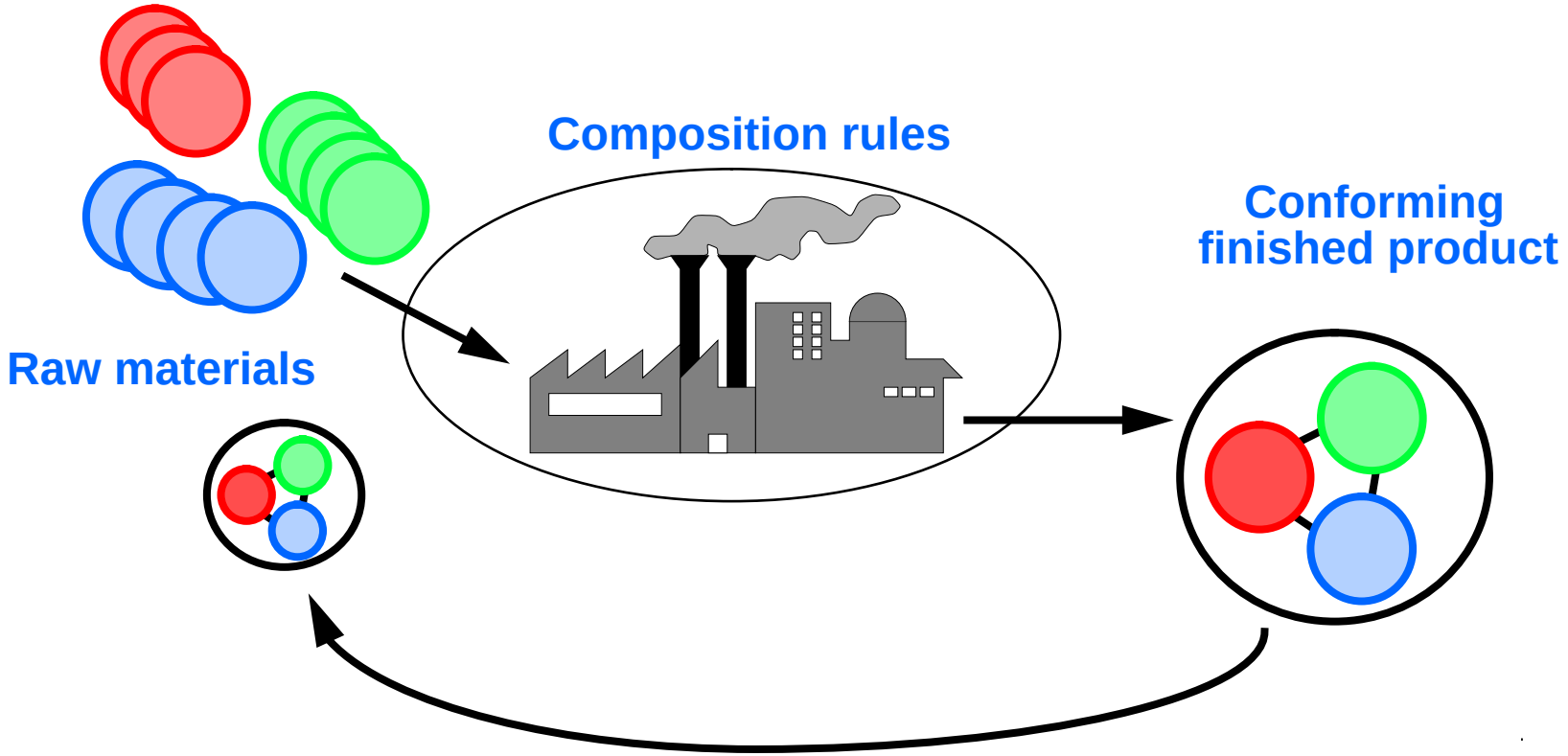
The (traditional) design process



Distributing the design process



Architecture is about structure

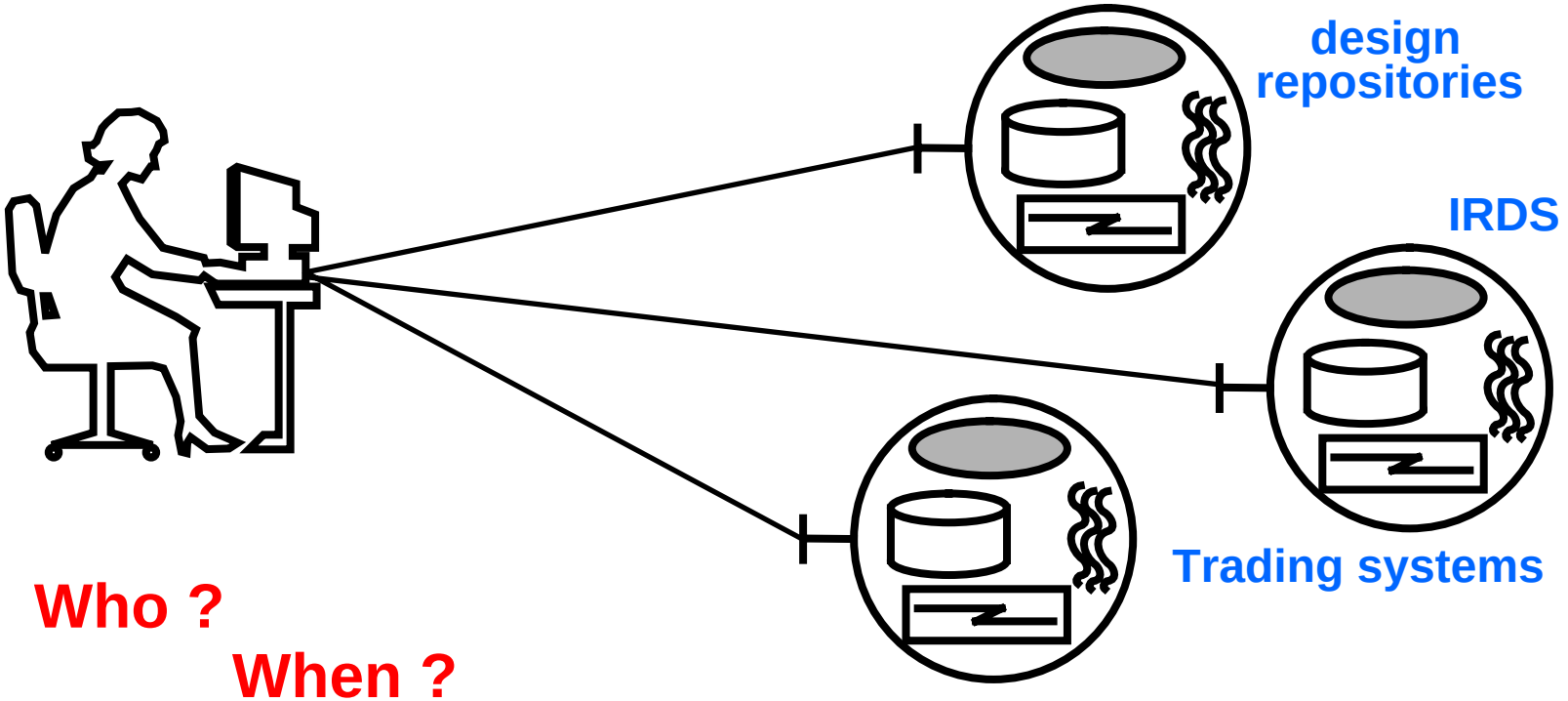


Component description

- **by the process which created them . . . . .**
- **or . . . . . as components in their own right**

purpose:	**what is it for?**
meaning:	**how may it be used?**
structure:	**what other components does it need or consist of?**
guarantees:	**what promises will be kept?**
technology:	**what technologies are necessary?**

Access to descriptions

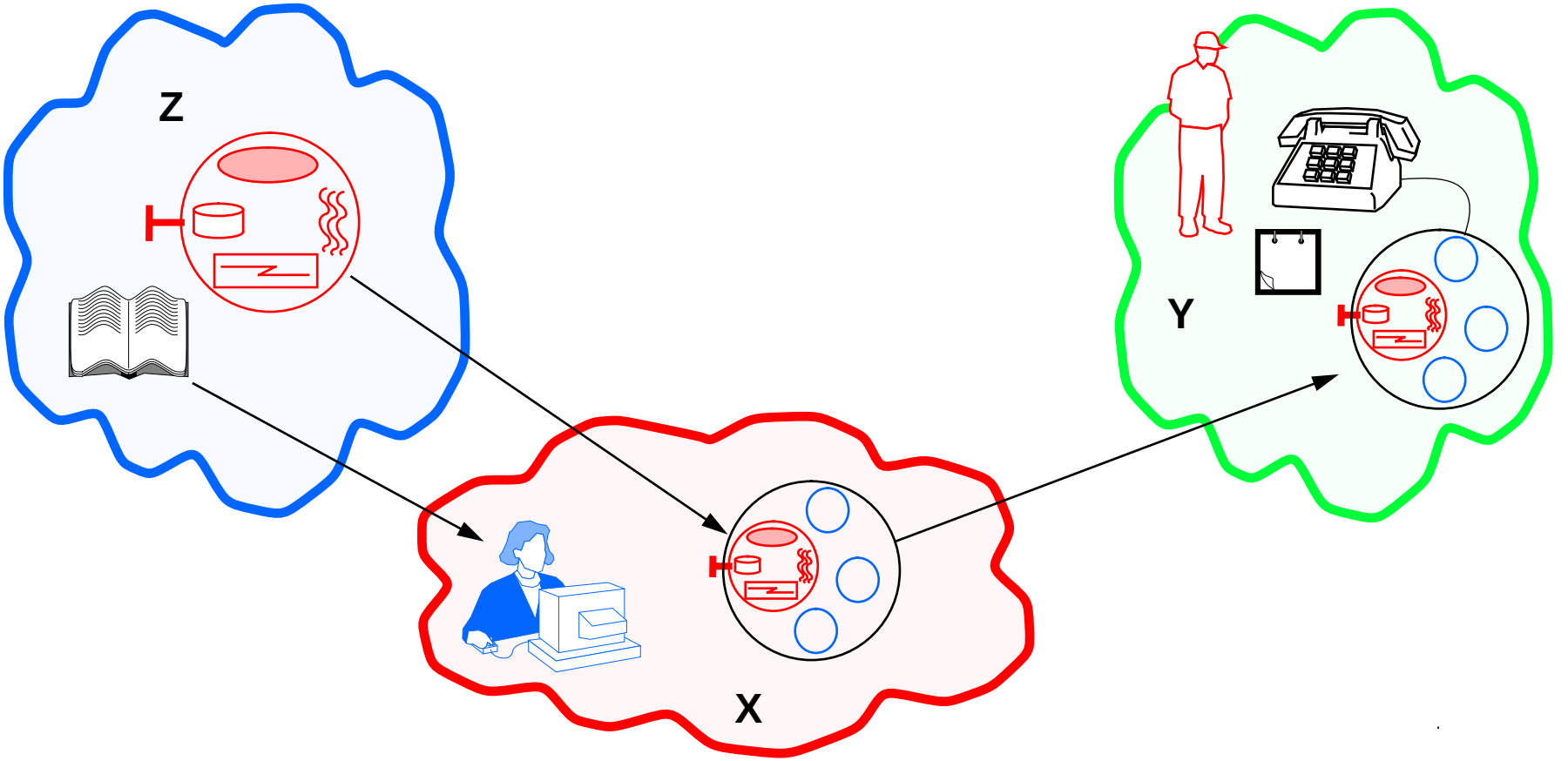


Who ?

When ?

Why ?

Challenges



Summary

- everything should go faster, be more efficient, better tailored and less costly
- architecture is about structure: components and combination rules
- keep things separate and allow substitution
- keep information about components not their construction process
- make this information available as widely as possible

Are you ready to face the challenges?