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 - Integrating Legacy Systems

Mark Madsen

Abstract

A major problem - the major problem - facing business today is the ability to manage change for profit and growth.

The legacy systems built in the past were not built for flexibility. How does one architect and design systems for flexibility? And how is this done while keeping existing systems running?

Today we build tomorrow's legacy. What can be done stop this problem recurring.

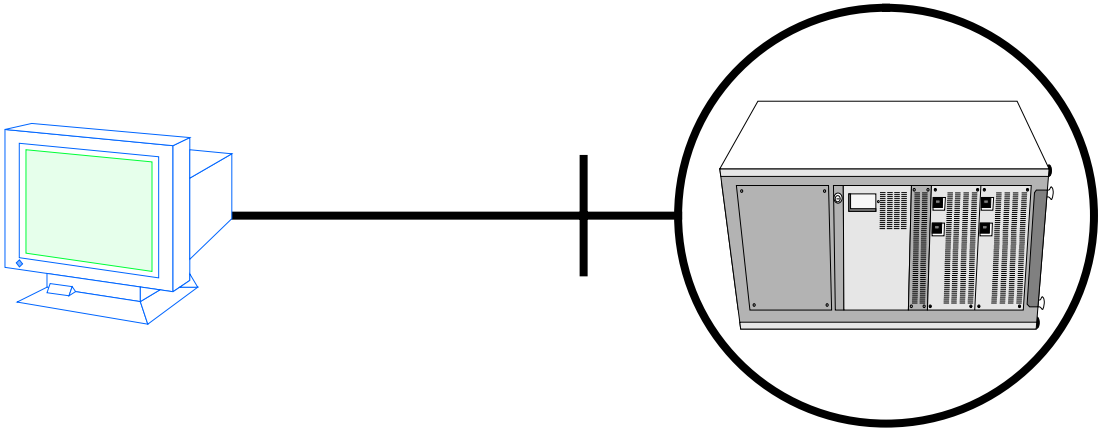
This module of the ANSAwise training programme explains the characteristics of legacy systems and shows what can be done to mitigate the impact of these characteristics for the future.

[This module was originally prepared for presentation at Eurocontrol Bretigny as part of the course on "Distributed Systems and CORBA", January 1996.]

APM.1694.03 **Approved** 3rd April 1996
Briefing Note

Distribution:
Supersedes:
Superseded by:

Integrating Legacy Systems



In this session

- Explore the origins of the Legacy Problem
- Show how it has arisen in the past
- Show how it arises now
- Show how it can arise in future
- Describe mechanisms for mitigating it

What Is The Legacy Problem?

- It is the difference between

  - the way we used to do things, and

  - the way we want to do things now

History of the Legacy Problem

- **This falls into distinct phases**
  - **past legacy problem**
  - **present legacy problem**
  - **future legacy problem**

The Past Legacy Problem

- **Incompatible data and file formats**
- **Hardware incompatibility**
- **Software tied to hardware**
- **Proprietary protocols and networks**

How The Past Legacy Problem Was Solved

- **Standardisation of**
  - **character codes**
  - **on-the-wire formats**
- **Interoperability facilities in operating systems**

The Present-Day Legacy Problem

- **Primary cause is applications without a separable server component**
- **Contributing factors**
  - **absence of clear interfaces (often due to poor language support)**
  - **insufficient encapsulation allowing state to be externally visible**

Solutions to the Present-Day Legacy Problem

- **Standardisation of object architectures**
  - **CORBA**
- **Standardisation of communications protocols**
  - **CORBA GIOP/IIOP**
- **Allowing interoperable systems to be built**
  - **encapsulated**
  - **behind service interface**

The Future Legacy Problem

- We are working to develop better legacy systems!
- Future legacy problems will arise from present-day systems that are
  - monolithic
  - insecure
  - not real-time
  - not scalable
- The problem cannot be avoided, but can be mitigated

Solutions to Future Legacy Problems

- **The best time to solve these is now!**
- **Use best techniques available**
  - **modular engineering: languages**
  - **trading and federation: domains**
  - **configurable infrastructure: tradeoffs**
  - **abstract and automate: tools**

Different Approaches to Legacy Systems

- **Transformation**
  - slow
  - expensive
- **Re-engineering**
  - slow
  - less expensive
- **Integration**
  - faster
  - inexpensive

Transforming Legacy Systems

- **Re-analyse requirements**
- **Scope available functionality**
- **Match requirements to design**
- **Re-implement**

Analyzing Requirements

- **Model the enterprise as a set of communities**
  - **each with specified objectives**
  - **each offering services to other communities**
- **Model the interactions between communities**
  - **examine the common information and information flows**
  - **define the constraints in these interactions**
  - **specify the interface in the form of a ‘contract’**

Re-Engineering Legacy Systems

- **Break system down by functional components**
- **Identify interfaces between components**
- **Build object-based replacements for each**
- **Roll in each replacement component in turn**

Integrating Legacy Systems

- Define role of each subsystem
- Define interfaces for each subsystem
- Build an object wrapper for each subsystem
- Make new clients use the interface
- Re-engineer components at the performance bottlenecks

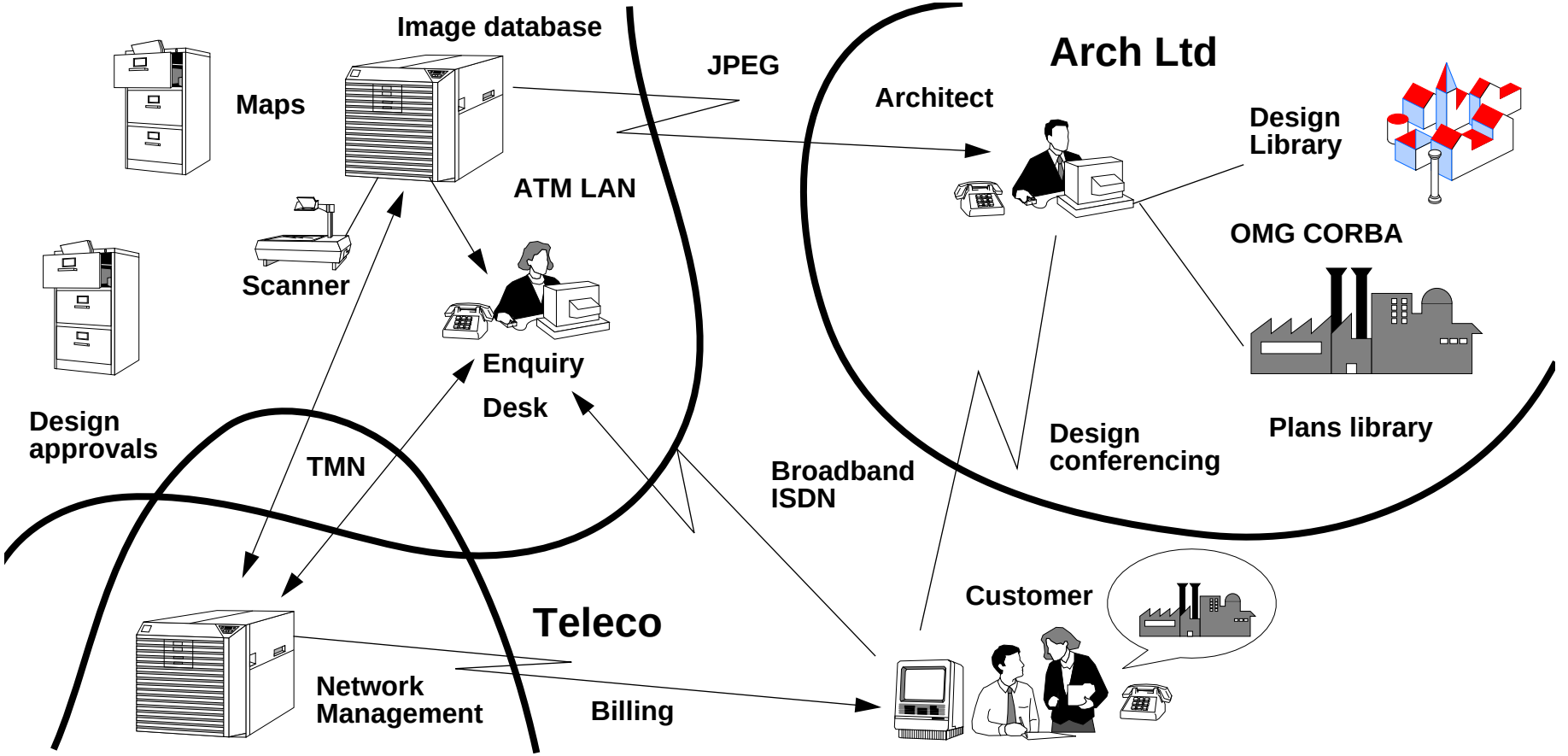
Obstacles to Legacy Integration

- Language complexities/restrictions
- Need to connect back-end to a CORBA system
- Limited to decentralisation rather than distribution

A scenario - urban planning

- The local authority for Newtown wants to encourage new housing and high-technology industry to move in
- The authority decides to provide electronic access to its Planning and Land Registry Functions
- The leading architects in the town (Arch Ltd) develop interactive design conferencing services with their clients
- The local authority out-sources management of its telecommunications services to Teleco plc

Newtown - Basis for federation



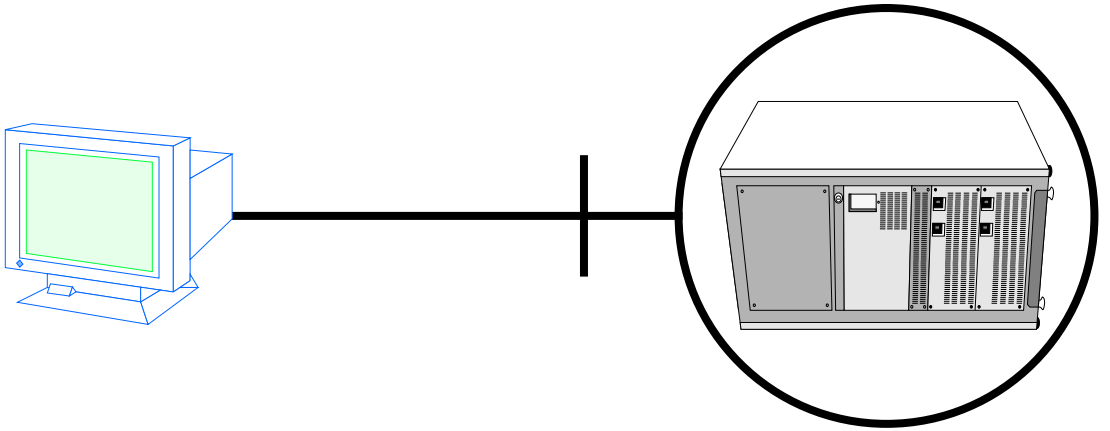
Aids to Integration

- **Modular engineering:**
  - **object-based languages for writing object wrappers**
- **Trading and federation:**
  - **gateways to cross from legacy to new domain**
- **Configurable infrastructure:**
  - **informed trade-offs between, for example, transparency and efficiency**
- **Abstract and automate:**
  - **toolkits for adding new functionality**

The Federation Wish List

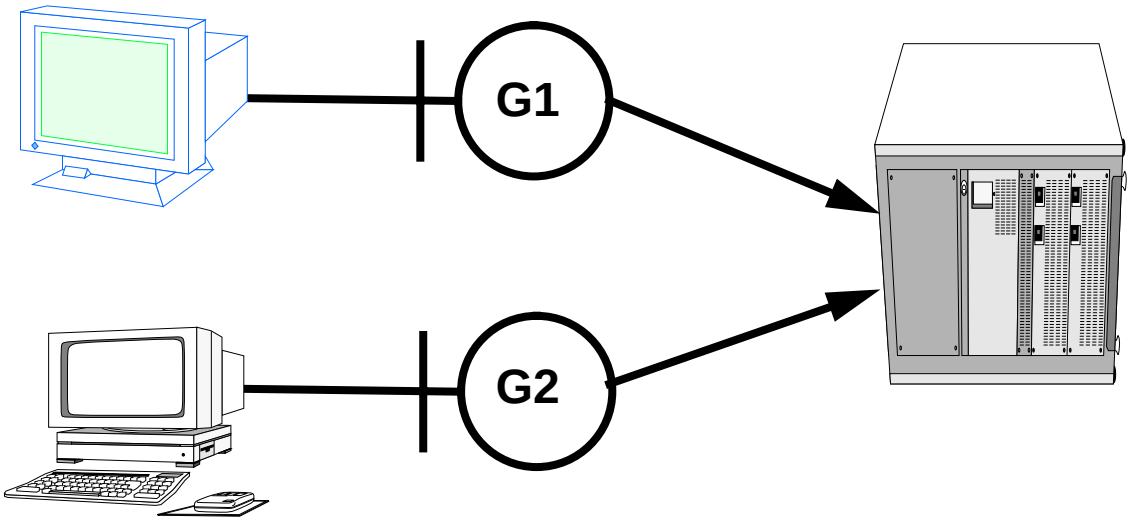
- Many legacy integration problems arise from new business situations
- These new business situations often involve federation of previously separate systems
  - integration of products from different vendors
  - need to span application boundaries
  - need to protect enterprise boundaries
  - desire to enable inter-organisational computing

Object Wrappers for Integration



- **Wrappers hide the legacy behind an interface**
  - **the legacy system is no longer visible except through the interface**
  - **the new clients must be able to use the available interface(s)**

Gateways for Integration



- **Gateways provide access to the old system from a new domain**
  - **different gateways may be provided for different domains**
  - **or for different kinds of functionality**

Legacy Case Study: IBM's CICS

- CICS is IBM's large-scale transaction processing system
- Used by many large companies (IBM claim all the Fortune 500)
- System has been built over a period of 30 years
- Servers for proprietary IBM systems (hard and soft)
- Client is required to be an IBM 3270 display terminal

The CICS Legacy Problem

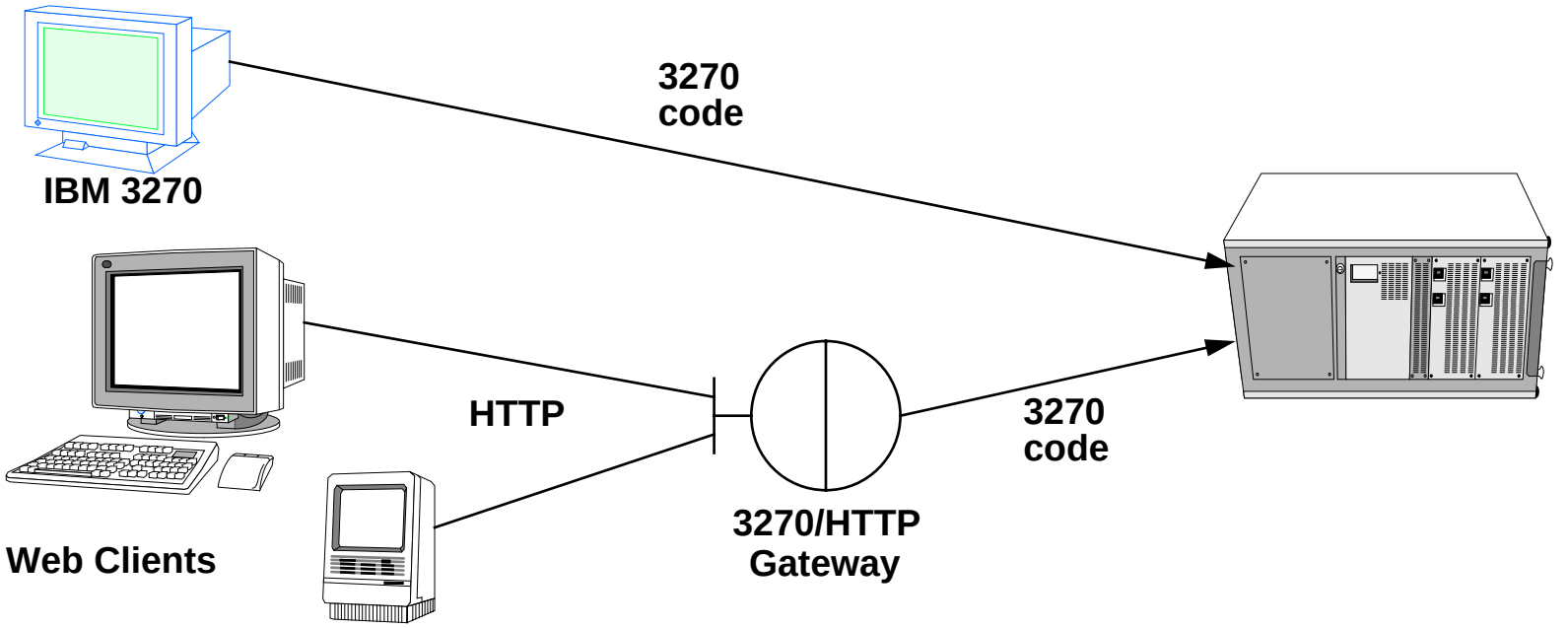
- More companies want to use the WWW
  - for internal information dissemination/exchange
  - for external dealings with other companies
- But CICS is heavily tied to IBM hardware and software

Integrating CICS with the WWW

- **IBM used the gateway approach**
  - **preserves existing functionality so present business processes remain unchanged**
  - **but allows wider use of business information without hardware and system upgrades**
  - **moves CICS functionality into the multi-platform world**

CICS to WWW Gateway Design

- **CICS service sees its client as a smart IBM 3270 display terminal**



- **Web clients see CICS information delivered by a webserver**

Summary

- The legacy problem is a permanent feature of systems
- Legacy systems can be transformed, re-engineered, or integrated
  - integration is the fastest and least expensive
- CORBA is an ideal framework for legacy integration

Finding Out More

- **We must learn to build a better legacy for the future:**
  - **see *APM.1513 Building a Better Legacy*, for a close study of the problems and their solutions**
- **For more on object wrapping, see**
  - ***The Essential CORBA: Systems Integration Using Distributed Objects* by T.J. Mowbray & R. Zahavi**
- **For details of IBM's CICS legacy enhancement programme, see *http://www.hursley.ibm.com/cics/saints/index.html***