Legacy Systems

- Older software systems that remain vital to an organisation

Objectives

- To explain what is meant by a legacy system and why these systems are important
- To introduce common legacy system structures
- To briefly describe function-oriented design
- To explain how the value of legacy systems can be assessed

Topics covered

- Legacy system structures
- Legacy system design
- Legacy system assessment

Legacy systems

- Software systems that are developed specially for an organisation have a long lifetime
- Many software systems that are still in use were developed many years ago using technologies that are now obsolete
- These systems are still business critical that is, they are essential for the normal functioning of the business
- They have been given the name legacy systems

Legacy system replacement

- There is a significant business risk in simply scrapping a legacy system and replacing it with a system that has been developed using modern technology
  - Legacy systems rarely have a complete specification. During their lifetime they have undergone major changes which may not have been documented
  - Business processes are reliant on the legacy system
  - The system may embed business rules that are not formally documented elsewhere
  - New software development is risky and may not be successful

Legacy system change

- Systems must change in order to remain useful
- However, changing legacy systems is often expensive
  - Different parts implemented by different teams so no consistent programming style
  - The system may use an obsolete programming language
  - The system documentation is often out-of-date
  - The system structure may be corrupted by many years of maintenance
  - Techniques to save space or increase speed at the expense of understandability may have been used
 - File structures used may be incompatible

The legacy dilemma

- It is expensive and risky to replace the legacy system
- It is expensive to maintain the legacy system
- Businesses must weigh up the costs and risks and may choose to extend the system lifetime using techniques such as re-engineering.
- This is covered in Chapters 27 and 28

Legacy system structures

- Legacy systems can be considered to be socio-technical systems and not simply software systems
  - System hardware - may be mainframe hardware
  - Support software - operating systems and utilities
  - Application software - several different programs
  - Application data - data used by these programs that is often critical business information
  - Business processes - the processes that support a business objective and which rely on the legacy software and hardware
  - Business policies and rules - constraints on business operations

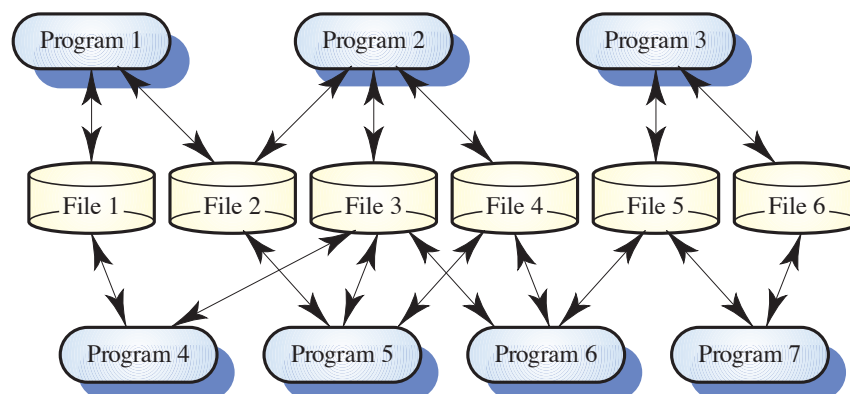# Legacy system components
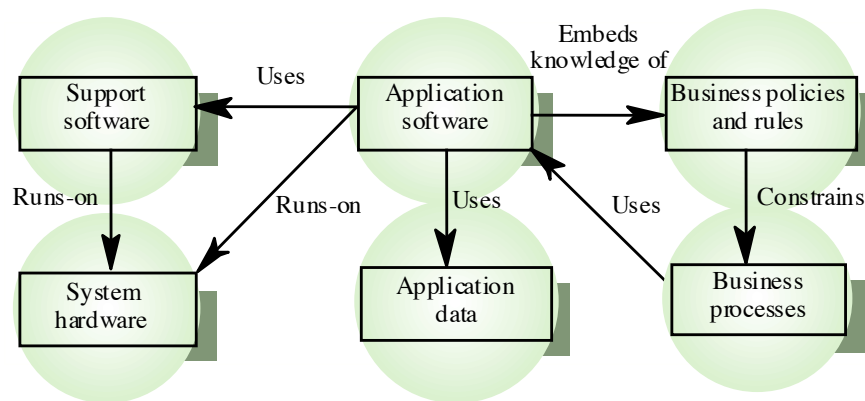
Layered model

Socio-technical system

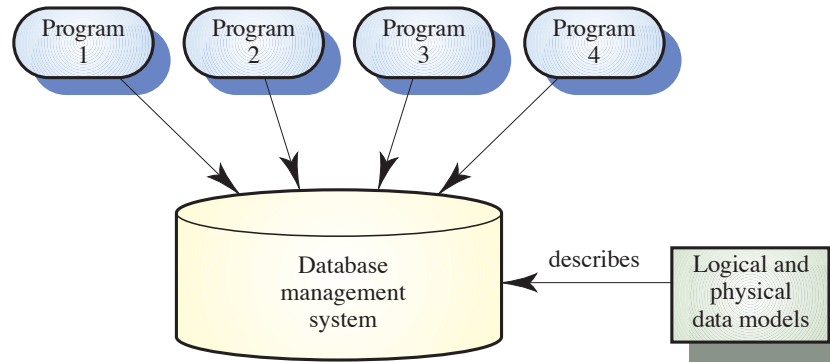| Business processes |
| --- |
| Application software |
| Support software |
| Hardware |
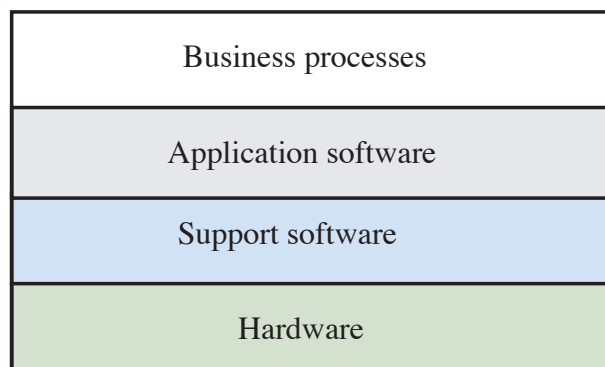
System change

- In principle, it should be possible to replace a layer in the system leaving the other layers unchanged

- In practice, this is usually impossible
  - Changing one layer introduces new facilities and higher level layers must then change to make use of these
  - Changing the software may slow it down so hardware changes are then required
  - It is often impossible to maintain hardware interfaces because of the wide gap between mainframes and client-server systems

---

Legacy application system

Database-centred system


```
Program 1    Program 2    Program 3    Program 4
```

Database management system ← describes — Logical and physical data models

Transaction processing

Account queries and updates

Serialised transactions

Teleprocessing monitor

Accounts database

ATMs and terminals
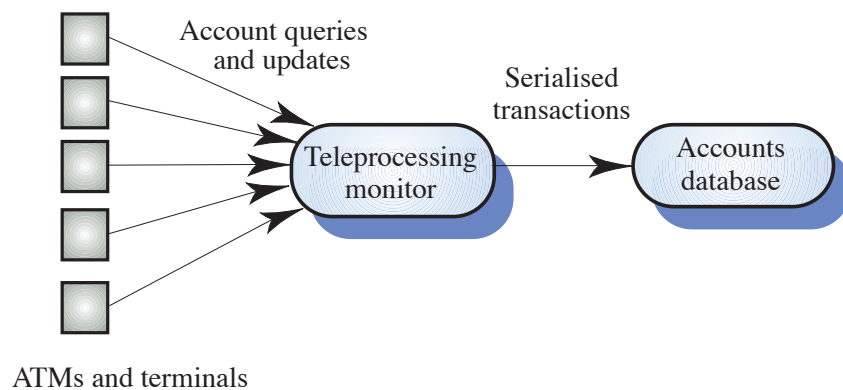
Legacy data

- The system may be file-based with incompatible files. The change required may be to move to a database-management system
- In legacy systems that use a DBMS the database management system may be obsolete and incompatible with other DBMSs used by the business
- The teleprocessing monitor may be designed for a particular DB and mainframe. Changing to a new DB may require a new TP monitor

Legacy system design

- Most legacy systems were designed before object-oriented development was used
- Rather than being organised as a set of interacting objects, these systems have been designed using a function-oriented design strategy
- Several methods and CASE tools are available to support function-oriented design and the approach is still used for many business applications

A function-oriented view of design
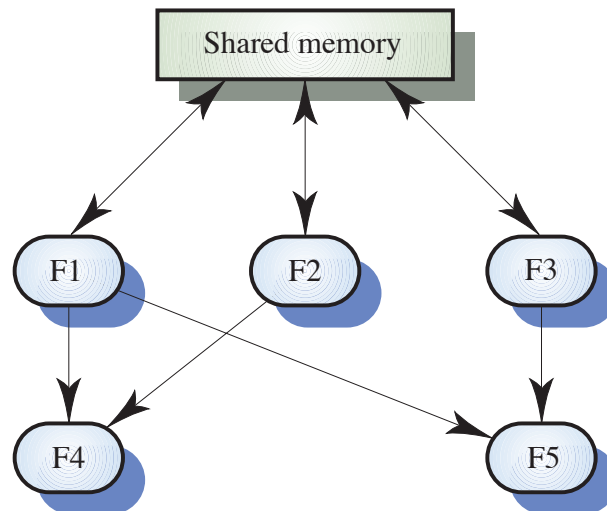
Functional design process

- **Data-flow design**
    - Model the data processing in the system using data-flow diagrams
- **Structural decomposition**
    - Model how functions are decomposed to sub-functions using graphical structure charts
- **Detailed design**
    - The entities in the design and their interfaces are described in detail. These may be recorded in a data dictionary and the design expressed using a PDL

Input-process-output model

System

Input   Process   Output
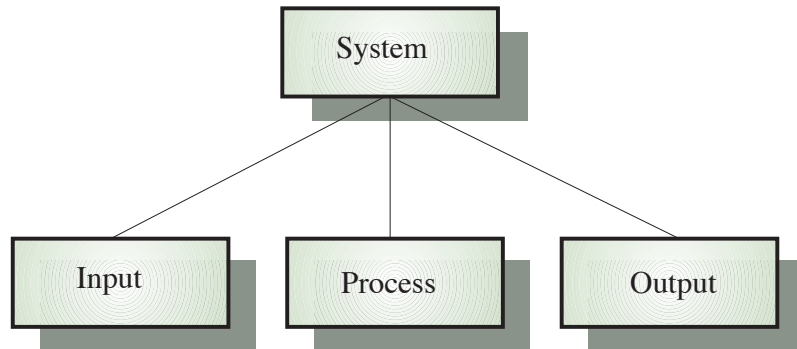
Input-process-output

- Input components read and validate data from a terminal or file
- Processing components carry out some transformations on that data
- Output components format and print the results of the computation
- Input, process and output can all be represented as functions with data 'flowing' between them

Data flow diagrams

- Show how an input data item is functionally transformed by a system into an output data item
- Are an integral part of many design methods and are supported by many CASE systems
- May be translated into either a sequential or parallel design. In a sequential design, processing elements are functions or procedures; in a parallel design, processing elements are tasks or processes

Payroll system DFD
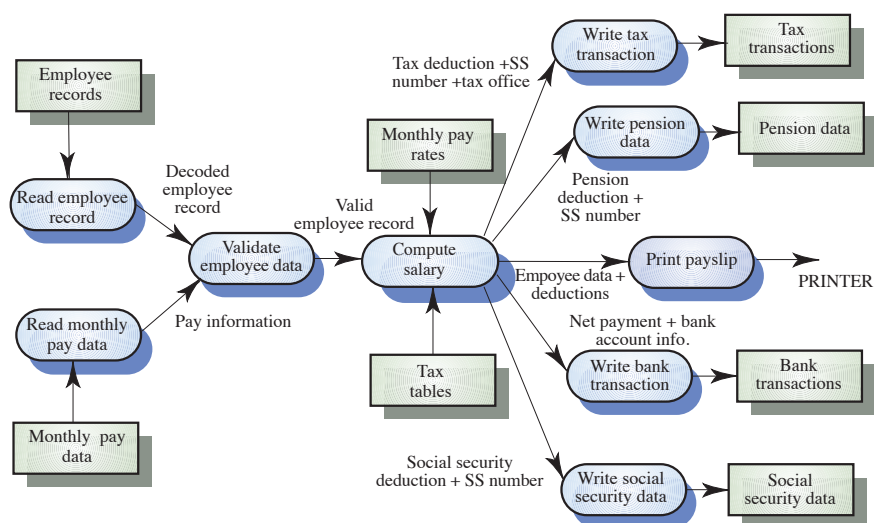
Payroll batch processing

- The functions on the left of the DFD are input functions
  - Read employee record, Read monthly pay data, Validate employee data
- The central function - Compute salary - carries out the processing
- The functions to the right are output functions
  - Write tax transaction, Write pension data, Print payslip, Write bank transaction, Write social security data

Transaction processing

- A ban ATM system is an example of a transaction processing system
- Transactions are stateless in that they do not rely on the result of previous transactions. Therefore, a functional approach is a natural way to implement transaction processing

```
INPUT

loop
    repeat
            Print_input_message (" Welcome - Please enter your card") ;
    until Card_input ;

    Account_number := Read_card ;
    Get_account_details (PIN, Account_balance, Cash_available) ;

PROCESS

    if Invalid_card (PIN) then
            Retain_card ;
            Print ("Card retained - please contact your bank") ;
    else
    repeat
            Print_operation_select_message ;
            Button := Get_button ;
            case Get_button is
                when Cash_only =>
                        Dispense_cash (Cash_available, Amount_dispensed) ;
                when Print_balance =>
                        Print_customer_balance (Account_balance) ;
                when Statement =>
                        Order_statement (Account_number) ;
                when Check_book =>
                        Order_checkbook (Account_number) ;
            end case ;
            Print ("Press CONTINUE for more services or STOP to finish");
            Button := Get_button ;
    until Button = STOP ;

OUTPUT

    Eject_card ;
    Print ("Please take your card ) ;
    Update_account_information (Account_number,  Amount_dispensed) ;

end loop ;
```

Design description of an ATM

Using function-oriented design

- For some classes of system, such as some transaction processing systems, a function-oriented approach may be a better approach to design than an object-oriented approach

- Companies may have invested in CASE tools and methods for function-oriented design and may not wish to incur the costs and risks of moving to an object-oriented approach

Legacy system assessment

- Organisations that rely on legacy systems must choose a strategy for evolving these systems
    - Scrap the system completely and modify business processes so that it is no longer required
    - Continue maintaining the system
    - Transform the system by re-engineering to improve its maintainability
    - Replace the system with a new system
- The strategy chosen should depend on the system quality and its business value

System quality and business value
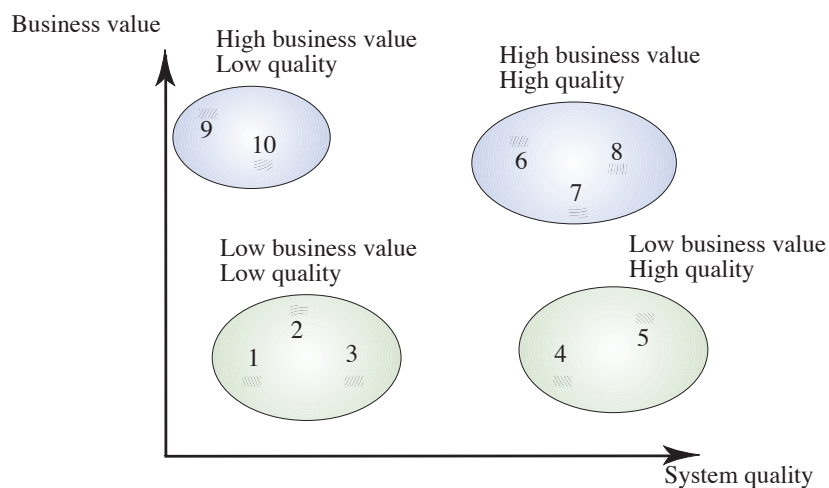
Legacy system categories

- Low quality, low business value
  - These systems should be scrapped
- Low-quality, high-business value
  - These make an important business contribution but are expensive to maintain. Should be re-engineered or replaced if a suitable system is available
- High-quality, low-business value
  - Replace with COTS, scrap completely or maintain
- High-quality, high business value
  - Continue in operation using normal system maintenance

Business value assessment

- Assessment should take different viewpoints into account
  - System end-users
  - Business customers
  - Line managers
  - IT managers
  - Senior managers
- Interview different stakeholders and collate results

System quality assessment

- Business process assessment
  - How well does the business process support the current goals of the business?
- Environment assessment
  - How effective is the system's environment and how expensive is it to maintain
- Application assessment
  - What is the quality of the application software system

Business process assessment

- Use a viewpoint-oriented approach and seek answers from system stakeholders
  - Is there a defined process model and is it followed?
  - Do different parts of the organisation use different processes for the same function?
  - How has the process been adapted?
  - What are the relationships with other business processes and are these necessary?
  - Is the process effectively supported by the legacy application software?

Environment assessment

Factor	Questions
Supplier stability	Is the supplier is still in existence? Is the supplier financially stable and likely to continue in existence? If the supplier is no longer in business, are the systems maintained by someone else?
Failure rate	Does the hardware have a high rate of reported failures? Does the support software crash and force system restarts?
Age	How old is the hardware and software? The older the hardware and support software, the more obsolete it will be. It may still function correctly but there could be significant economic and business benefits to moving to more modern systems.
Performance	Is the performance of the system adequate? Do performance problems have a significant effect on system users?
Support requirements	What local support is required by the hardware and software? If there are high costs associated with this support, it may be worth considering system replacement.
Maintenance costs	What are the costs of hardware maintenance and support software licences? Older hardware may have higher maintenance costs than modern systems. Support software may have high annual licensing costs.
Interoperability	Are there problems interfacing the system to other systems? Can compilers etc. be used with current versions of the operating system? Is hardware emulation required?

Application assessment

Factor	Questions
Understandability	How difficult is it to understand the source code of the current system? How complex are the control structures which are used? Do variables have meaningful names that reflect their function?
Documentation	What system documentation is available? Is the documentation complete, consistent and up-to-date?
Data	Is there an explicit data model for the system? To what extent is data duplicated in different files? Is the data used by the system up-to-date and consistent?
Performance	Is the performance of the application adequate? Do performance problems have a significant effect on system users?
Programming language	Are modern compilers available for the programming language used to develop the system? Is the programming language still used for new system development?
Configuration management	Are all versions of all parts of the system managed by a configuration management system? Is there an explicit description of the versions of components that are used in the current system?
Test data	Does test data for the system exist? Is there a record of regression tests carried out when new features have been added to the system?
Personnel skills	Are there people available who have the skills to maintain the application? Are there only a limited number of people who understand the system?

System measurement

- You may collect quantitative data to make an assessment of the quality of the application system
  - The number of system change requests
  - The number of different user interfaces used by the system
  - The volume of data used by the system

Key points

- A legacy system is an old system that still provides essential business services
- Legacy systems are not just application software but also include business processes, support software and hardware
- Most legacy systems are made up of several different programs and shared data
- A function-oriented approach has been used in the design of most legacy systems

Key points

- The structure of legacy business systems normally follows an input-process-output model
- The business value of a system and its quality should be used to choose an evolution strategy
- The business value reflects the system's effectiveness in supporting business goals
- System quality depends on business processes, the system's environment and the application software