In this chapter you will learn to:

- recognise and describe a transaction
- identify, describe and use a batch transaction processing system
- distinguish between the storage of collected data and the storage of processed data in a batch system
- identify, describe and use a real time transaction processing system
- compare and contrast batch and real time transaction processing
- analyse an existing transaction processing system to determine its strengths and weaknesses
- design and implement procedures for validating entered data
- assess the work routine of a clerk in a manual transaction system to determine its suitability for automation
- identify participants, data/information and information technology for the given types of transaction processing systems
- describe the relationships between participants, data/information and information technology for the given types of transaction processing systems
- for a scenario diagrammatically represent transaction processing using data flow diagrams
- distinguish between the different types of TPS
- store digital data in databases and other files in such a way that it can be retrieved, modified and further processed
- implement systems to store paper transactions
- select and apply backup and recovery procedures to protect data
- document, including diagrammatical representations, the steps in batch processing
- document, including diagrammatical representations, steps in real time transaction processing
- identify systems for which batch is appropriate and is not appropriate
- distinguish between on-line real time and batch systems
- create and use a transaction processing system
- describe the operation of relevant hardware and how each is used to collect data for transaction processing
- design and justify paper forms to collect data for batch processing
- design user friendly screens for on-line data collection
- identify existing procedures that may provide data for transaction processing
- create user interfaces for on-line real time and batch updating, and distinguish between them
- identify situations where data warehousing and data mining would be an advantage
- assess the impact on participants involved in transaction processing
- identify jobs that have changed and/or jobs that have been created as a result of transaction processing, and report on the implications of these changes for participants in the system
- discuss alternatives for when the transaction processing system is not available and explain why they need to be periodically tested
- identify security, bias and accuracy problems that could arise from the actions of participants
- recognise the significance of data quality

Which will make you more able to:

- apply and explain an understanding of the nature and function of information technologies to a specific practical situation
- explain and justify the way in which information systems relate to information processes in a specific context
- analyse and describe a system in terms of the information processes involved
- develop solutions for an identified need which address all of the information processes
- evaluate and discuss the effect of information systems on the individual, society and the environment
- demonstrate and explain ethical practice in the use of information systems, technologies and processes
- propose and justify ways in which information systems will meet emerging needs
- justify the selection and use of appropriate resources and tools to effectively develop and manage projects
- assess the ethical implications of selecting and using specific resources and tools, recommends and justifies the choices
- analyse situations, identify needs, propose and then develop solutions
- select, justify and apply methodical approaches to planning, designing or implementing solutions
- implement effective management techniques
- use methods to thoroughly document the development of individual or team projects.

In this chapter you will learn about:

Characteristics of transaction processing systems

- a transaction - a series of events important to an organisation that involve a request, an acknowledgement, an action and an outcome
- the components of a transaction processing system, including:
  - purpose
  - data
  - information technology
  - processes
  - participants
- batch transaction processing - the collection and storage of data for processing at a scheduled time or when there is sufficient data
- real time transaction processing - the immediate processing of data
- the significance of data validation in transaction processing
- the historical significance of transaction processing as the first type of information systems

Types of transaction processing systems

- web-based
- non web-based
- on-line real time
- batch
- systems that appear real time, responding as the transactions occur, but where the actual updating is batch processed, such as credit card transactions

Storing and retrieving in transaction processing systems

- storage of digital data in databases and files
- retrieval of stored data to conduct further transaction processing such as printing invoices
- systems to store paper records of transactions
- data backup and recovery, including:
  - grandfather, father, son
  - off-site storage
  - secure on-site storage
  - full and partial backups
  - recovery testing
  - suitable media
  - specialised backup software
  - transaction logs
  - documenting backup and recovery procedures
  - mirroring
  - rollback
- updating in batch systems:
  - historical significance
  - limitations of batch processing
  - technology required
  - steps in a batch update
  - suitable applications
- updating in on-line real time systems:
  - relevance and impact
  - technology required
  - hardware requirements - large secondary storage
  - software requirements – on-line database and user friendly interface
  - steps in on-line real time processing
  - suitable applications

Other information processes in transaction processing systems

- collecting in transaction processing:
  - hardware including
    - Automatic Teller Machines (ATM)
    - barcode readers
    - Radio Frequency Identification (RFID) Tags
  - collection from forms
  - screen design for on-line data collection
  - web forms for transaction processing (real time and batch)
- analysing data, in which output from transaction processing is input to different types of information systems, such as:
  - decision support
  - management information systems
  - data warehousing systems (for data mining)
  - enterprise systems

Issues related to transaction processing systems

- changing nature of work and the effect on participants, including:
  - the automation of jobs once performed by clerks
  - shifting of workload from clerks to members of the public
- the need for alternate procedures to deal with transactions when the TPS is not available
- bias in data collection:
  - when establishing the system and deciding what data to collect
  - when collecting data
- the importance of data in transaction processing, including:
  - data security
  - data integrity
  - data quality
- control in transaction processing and the implications it has for participants in the system
- current and emerging trends in transaction processing
  - data warehousing and data mining
  - Online Analytical Processing (OLAP) and Online Transaction Processing (OLTP)

4

OPTION 1
TRANSACTION PROCESSING SYSTEMS

Transaction processing systems are crucial to the operation of most finance, banking and electronic commerce organisations. Transaction processing is primarily concerned with maintaining data integrity. Such systems can operate at the single database level, but they also operate at higher levels where data in many databases and even many different systems is involved. For example transferring funds from one financial institution to another.

So what is a transaction? A transaction is a series of events that when performed together complete some unit of work that is important to an organisation. Each transaction has two possible outcomes, either it is a complete success or it is a complete failure.

> **Transaction**
> A unit of work composed of multiple events that must all succeed or must all fail. Events perform actions that create and/or modify data.

If a transaction is successful then all the events contained within the transaction must have performed their actions successfully. However, if one or more events are unable to complete their actions then the whole transaction must fail, which requires the data to be left in the same state it was in prior to the transaction commencing. This means any events that could successfully perform their actions must be stopped. For example when transferring funds between accounts two events must occur; one account is debited and another credited. If the debit event fails then the credit event must be stopped, similarly if the credit event fails then the debit event must be stopped.

Managing the success or failure of transactions is an essential process performed during transaction processing. Transaction processing systems include mechanisms for ensuring events can be completed successfully, but not yet permanently. Essentially the transaction processing system requests that each event occur and receives a response indicating that the actions performed are guaranteed to succeed or have failed. If a successful response is received for all events then the transaction as a whole can be "committed", meaning each event is requested to store its data changes permanently within the appropriate databases or systems. If one or more events have failed then the transaction is "rolled back", meaning each event is requested to abort all actions. In response each event sends an acknowledgement to confirm they have performed the request.

A transaction can include events that perform actions on a single database, many databases or on a variety of different information systems. These databases and systems can be widely distributed and in some instances they are operated by different organisations. The detail of how such transactions are processed will become clearer throughout the chapter.

> **GROUP TASK Discussion**
> Brainstorm a list of typical transactions and their component events and actions. Discuss problems that may cause these transactions to fail.

In this Option we commence by examining characteristics of transaction processing systems. This includes a brief examination of the history of transaction processing, how transaction processing automates manual tasks, the components of transaction processing systems and how such systems maximise the accuracy of data. We then examine real time and batch transaction processing systems. In real time systems each transaction is immediately processed online, whilst batch processing collects input data over time and then at some later time batches of many similar transactions are processed. Backup and recovery strategies and technologies are examined. We then examine collection hardware and forms used for collection. The data within transaction processing systems is used as input to other systems; we briefly consider examples of such systems. Finally we discuss issues related to transaction processing systems.

CHARACTERISTICS OF TRANSACTION PROCESSING SYSTEMS

Transaction processing is one of the earliest commercial uses of computer systems. In this section we examine some early examples of transaction processing to illustrate how such systems automate and improve upon manual transaction processing. We then examine features of manual transaction processing systems that make them well suited to automation. Finally we examine modern transaction systems – their components and how they maintain the accuracy of data.

HISTORICAL SIGNIFICANCE OF TRANSACTION PROCESSING

The operations performed by transaction processing systems were up until the 1950s performed solely by clerks using manual processes. Early computers were originally developed to solve scientific and mathematical problems for government and military. It was during the 1950s that the application of computers to business and financial records emerged.

Prior to the 1980s it was common for complete transaction processing applications to be developed (often in Cobol) for each individual organisation. During the 1980s database management systems emerged to manage and control access to databases. Today most transaction processing systems are based on one or more relational database management systems (RDBMS) with client applications being written to meet an organisation's specific needs. Some of the significant developments that have led to today's transaction processing systems are outlined below.

Fig 4.1
UNIVAC I the first commercially available computer used for transaction processing.

- UNIVAC I (Universal Automatic Computer), released in 1951, was the first commercially produced computer to gain wide acceptance by the public. The UNIVAC I was based on vacuum tubes and was the first computer to be routinely used for batch processing of business transactions. UNIVAC I was designed and built by John Presper Eckert and John William Mauchly. Their company Eckert-Mauchly Computer Corporation was bought by Remington Rand – both Eckert and Mauchly continued to work for Remington Rand after selling the company. The UNIVAC II and UNIVAC III were subsequently released.

- The programming language Cobol (Common Business Oriented Language) was developed in 1959. At the time computers were largely used for scientific and mathematical calculations. Cobol, as its name suggests, was targeted directly at business applications and is still widely used on large mini and mainframe computers. Cobol was the first language for large scale transaction processing.
- In 1964 IBM released its highly successful System/360 range of computers and peripherals (see *Fig 4.2*). These general-purpose systems supported approximately 40 different peripherals and included the ability to include redundant components to improve fault tolerance. Information systems based on System/360 supported real time input and processing from hundreds of attached terminals.
- In 1969 IBM released the first version of CICS (Customer Information Control System). In terms of transaction processing, CICS is a transaction processing monitor or TPM – it manages the processing of transactions from multiple clients to multiple servers. This software product has been continuously upgraded and is widely used today.
- SQL was first developed in the early 1970s by IBM under the name "System R". The design ideas for System R were a direct result of Ted Codd's work – Codd is considered the founder of relational database theory. At the time System R was viewed as a product to allow users to directly interrogate databases. The original designers never intended it to become a language that would be used from within applications.

Fig 4.2
IBM System/360 Model 65 operator consol attached to the CPU.

- In the early 1980s commercial general-purpose relational database management systems (RDBMS) emerged. Oracle just beating IBM's release of DB2. These systems used SQL both to create relational databases using DDL (Data Definition Language) statements and to view and update relational data using DML (Data Manipulation Language) statements.
- SQL first became an ANSI (American National Standards Institute) standard in 1986, however most current database systems, although compliant with most of the standard SQL syntax also include their own non-standard extensions.
- Microsoft entered the RDMS market in the 1990s with its SQL Server product. Microsoft SQL Server evolved from the Sybase DBMS; Microsoft dissolved their partnership with Sybase and renamed their product SQL Server.
- Today Oracle, IBM's DB2 and Microsoft's SQL Server dominate the market, however some open source products such as MySQL have significant market share within small to medium sized organisations.
- Today large enterprises such as banks, large corporations and government departments use transaction processing monitor (TPM) software to manage transactions across a variety of databases and applications of different types, operated by different organisations and in different locations. Common TPMs in use include IBM's CICS and Encina products, BEA System's Tuxedo software and more recently Microsoft Transaction Server (MTS). TPMs are an example of enterprise systems as they manage critical data and processes across an entire organisation.

GROUP TASK Discussion
Initially transaction processing software was written for each specific application. Today it is common to use standard DBMS and TPM software and only the client applications are custom solutions. Propose likely reasons why this has occurred.

GROUP TASK Research
Research and identify examples of enterprise systems. Determine whether these systems process transactions.

AUTOMATION OF MANUAL TRANSACTION PROCESSING

Processing of manual transactions almost always follows a strict sequence of events. Each event must be acknowledged as complete before the next commences and if any event fails then the entire transaction is aborted. In manual systems, events are performed by clerks and other personnel according to strict predefined rules. Indeed in large organisation's it is common for each clerk to repetitively perform just one of the events within each transaction. The transaction is handed onto the next clerk responsible for the next event in the sequence. The strict sequence and rules of such transactions make them particularly well suited to automation using computers.

As a simple example let us consider a manual system used within a small store and then assess the benefits of automating this system. The store is operated by a husband and wife team who have time during the day to perform all sales, purchasing, stocktaking and other transactions manually. The store uses a simple cash register, which is essentially a calculator with an attached cash drawer. The cash register does keep a total of all sales processed during the day. The store has an EFTPOS terminal, which operates as a separate system.

Sales – this transaction occurs to process each customer's purchases.
1. Locate price on item and enter into cash register.
2. Repeat 1 for each product.
3. Calculate total.
4. Receive payment from customer.
5. If EFTPOS payment then wait for approval and hand EFTPOS receipt to customer.
6. Enter payment amount into cash register.
7. If cash payment then calculate and hand change to customer.
8. Hand register receipt to customer.

Stocktake – this transaction collects data to enable the storeowner to calculate the quantity of each product to purchase.
1. Make a photocopy of stocktake sheets. These sheets specify the required number of each product when fully stocked, the product's supplier and also columns for recording current stock in the store.
2. Count and record number of each product on shelves.
3. Count and record number of each product in storeroom.

Purchasing – this transaction produces purchase orders for each supplier.
1. Complete Stocktake.
2. Calculate number of each item to purchase and record on stocktake sheets.
3. Create purchase order for supplier.
4. Work through stocktake sheets recording each product from current supplier.
5. Calculate order total.
6. Repeat steps 3, 4 and 5 for each supplier.
7. Submit all purchase orders to suppliers via fax.

GROUP TASK Discussion
Three transactions are described above, however there are other transactions that need to occur. Propose other likely transactions and outline a set of possible events occurring to perform these transactions.

Notice that much of the data used by all three of the above transactions is the same. It is the information generated by the transaction that is different. Furthermore the output from one transaction is used as data for another transaction. For example each sales transaction reduces the amount of stock, and each stocktake transaction produces the data required for purchasing. Such observations make this system well suited to automation. The flow of data and information entering and leaving each of these transactions is modelled on the data flow diagram in *Fig 4.3*. Note that each of the transactions is represented as a process as they are composed of events that process data in some way. Each of these transactions could be expanded into a lower level DFD or a step-by-step description that details their component events.

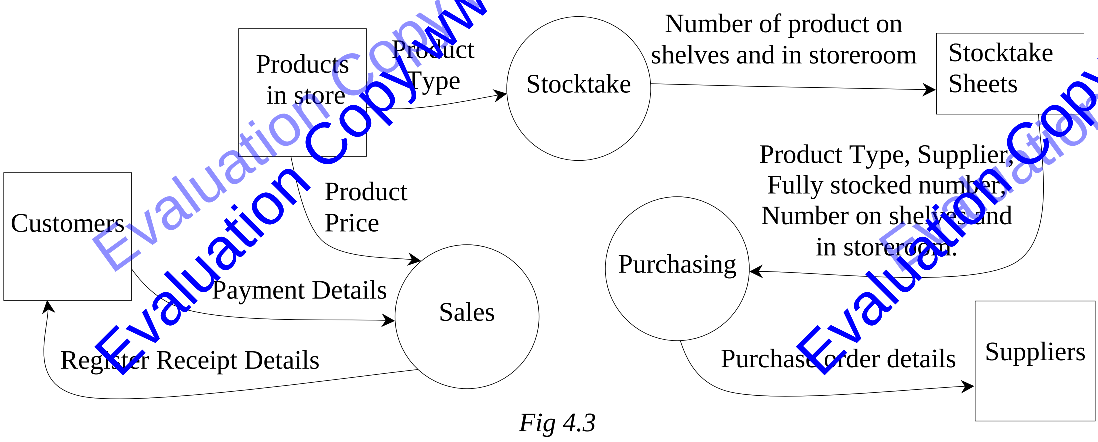


Fig 4.3
Data flow diagram modeling the flow of data between a store's manual transactions.

The stocktake sheets perform many of the tasks performed by a database, hence on the DFD a data store is used. They store all the data required by the purchasing transaction process. In addition the stock take sheets allow processing to halt between the stocktake process and the purchasing process. The Products in Store entity could also have been represented as a data store as each product stores its price in the form of a price tag and its product type. In reality these are the actual products, hence representing them as an external entity makes more sense.

No doubt it is clear that this system could be automated using a relational database to integrate sales, product, supplier, orders and stocktake data. Later in this chapter we shall examine a point of sale (POS) system, which is essentially an automated version of the above manual system. At this stage we are interested in the strengths and weaknesses of manual systems and of automation. Let us consider some general areas relevant to most manual systems together with common strengths and weaknesses of automation. We shall then discuss our local store example in an attempt to assess its suitability for automation.

Manual system strengths:

- Minimal start-up costs – little or no initial capital expenditure.
- Minimal training time and costs.
- Quick response to changing requirements.
- Well suited to small organisations where participants have time and fulfil multiple roles.
- Responds well to human insight and intuition.

Manual system weaknesses:

- Analysis of historical data is difficult and time consuming.
- Transactions take considerably longer to process.
- Difficult to rigidly enforce transaction rules and sequences.
- Redundant or duplicate data is a feature of most manual systems.
- Some human errors are to be expected.
- System becomes more and more difficult to manage as it grows.
- Making backups of data is difficult and is rarely, if ever, performed for all data.

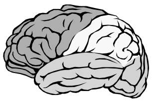
Automated transaction processing strengths:

- Much faster transaction processing.
- Less repetitive work for participants.
- Enforces the sequence and rules for each transaction.
- Calculation errors are virtually eliminated.
- Ability to integrate transaction processing with outside organisations.
- Historical data available for statistical and financial analysis.
- Backups easily made and restored if system fails.
- System easily grows as transaction processing needs grow.

Weaknesses of automated transaction processing:

- Significant start-up costs to purchase information technology.
- Extensive training required to operate the system.
- Changes to requirements often require specialised expertise to implement.
- Rigidly enforces existing transaction rules and sequences for all data.
- Less total work for humans resulting in lower employment.
- Reliance on information technology – failure of one or components can cripple the entire system.

In our local store example the storeowners are a husband and wife team who are currently able to complete the manual transactions. In this case the time saved through automation is unlikely to result in increased profits. Furthermore the cost required to set-up and learn to operate a new automated system is unlikely to be justified. If the storeowners had sufficient expertise to design and develop their own automated system then this would be worthwhile. Without an automated system it is difficult for the owners to accurately monitor sales trends over time. If they were able to perform such historical analysis then perhaps significant savings could be made by maintaining more efficient stock levels of products according to predicted demand at different times of the year. It is likely that this is currently occurring in a somewhat ad-hoc manner for obviously seasonal items – such as ice creams, Christmas decorations and gloves.



Consider the following:

Each of the following businesses currently use a manual system for recording their various transactions.

- A hardware store that stocks thousands of different items and has a staff of 8 employees working at all times.
- A small bookstore that is able to supply any title but maintains minimal stock. The store purchases titles as they are ordered by customers.
- A carpenter who substantially does subcontract work for 3 builders but does do some small jobs for residential customers.
- An eBay store that started out selling approximately 5 items per day, but is now selling 50 items per day.

GROUP TASK Discussion
Assess the suitability of an automated transaction processing system for each of the above businesses. Discuss likely advantage and disadvantages of automation compared to retaining their existing manual systems.

COMPONENTS OF TRANSACTION PROCESSING SYSTEMS

In this section we examine the various components of transaction processing systems. Like all information systems (see *Fig 4.4*), transaction processing systems operate within an environment that includes information processes, participants, data/information and information technology. All these components work together to meet the system's purpose. Each transaction is an information process and is therefore composed of events that are also information processes. For example adding a new customer to a database involves collecting and storing information processes. All information processes are performed using the resources within the system. The system's resources include participants, data and information, and information technology.

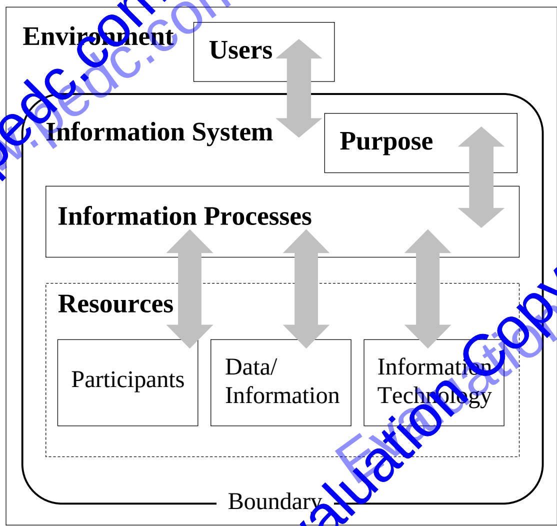


Fig 4.4
Components of an information system.

We have already introduced the general nature of transactions and much of the remainder of this option continues to examine different types of transactions in more detail. Therefore in this section we shall concentrate on participants, data/information and information technology within transaction processing systems. Recall that participants are people who carry out or initiate information processes. Information technology includes the hardware and software that carries out information processes.

Participants

Anybody who interacts directly with a transaction processing system becomes a participant in that system – they are integral to the system's operation. Therefore participants include people who work for the organisation that operates the transaction processing system and also people (often customers of the organisation) who enter data that initiates transactions. For example a bank employee is a direct user (and participant) as they initiate the printing of monthly bank statements. Customers become direct users (and participants) when they use Internet banking to initiate say the transfer of money between accounts. On the other hand indirect users are not participants, they send and/or receive data from the system but do not directly cause its entry or display. For example when a monthly or quarterly bank statement is received by a customer in the mail the customer is not a participant, rather they are an indirect user as they did not initiate the generation of the statement directly.

People in the environment only become participants in online real time transactions. Real time transactions are performed immediately in response to user or participant input. On the other hand for batch processes people from the environment may well provide data, but no transaction processing occurs or is initiated by that person. Rather the transaction is performed along with other similar transactions at some later time or when a sufficient quantity of data is present. Consider the difference between a transaction performed using an ATM compared with writing a cheque. When using an ATM the user initiates and therefore causes the transaction to be performed immediately – essentially they are performing duties similar to a bank teller. However when writing a cheque the customer has little control over when the transaction is actually processed and furthermore they are not interacting directly with the bank's transaction processing system.

Data/Information

In the majority of transaction processing systems data is stored in databases – usually relational databases. This data is transformed into information by the system's information processes. We studied the organisation and design of relational databases in depth in chapter 2. All the information in regard to tables, records, relationships, referential integrity, data validation, data integrity and data verification applies to transaction processing systems. However in transaction processing systems a further issue exists – how to ensure the integrity (correctness and accuracy) of data during transactions. What if another user or process views or alters data during a transaction? What if the data received from another system has problems? What if the system fails in some way during a transaction? In regard to data and information such issues are resolved by recording the detail of all transactions in a transaction file or log. How these transaction records help to resolve these issues will become clearer in the next section on data validation and data integrity.

Within transaction processing systems additional data is always created to record details of each transaction that occurs. In older systems the actual live data was commonly known as the master file and the details of each transaction was recorded in a transaction file. The application controlled and managed both the transaction file and the master file. All changes being recorded in the transaction file during transaction processing, with changes to the master file only being made when transactions are finally committed. Newer systems still create such transaction data (often called a transaction log), however management of this transaction data is left up to the DBMS and, if used, the transaction processing monitor rather than the application software. Most commercial operating systems also provide transaction capabilities as part of the file system. Such operating systems create transaction records that allow actions on complete files to form part of transactions. These operating system capabilities are also available to other applications, including transaction processing monitors.

To simplify our discussion let us refer to the transaction data or transaction file as a transaction log and the actual data as the master file. Recall that transactions can be committed or rolled back. The transaction log contains the essential data that facilitates this ability. When an event occurs as part of a transaction two possibilities arise:

1. *Fig 4.5* describes the first possibility for an event that modifies a single record. The event occurs however the changed or added records are recorded in the transaction log and no change is

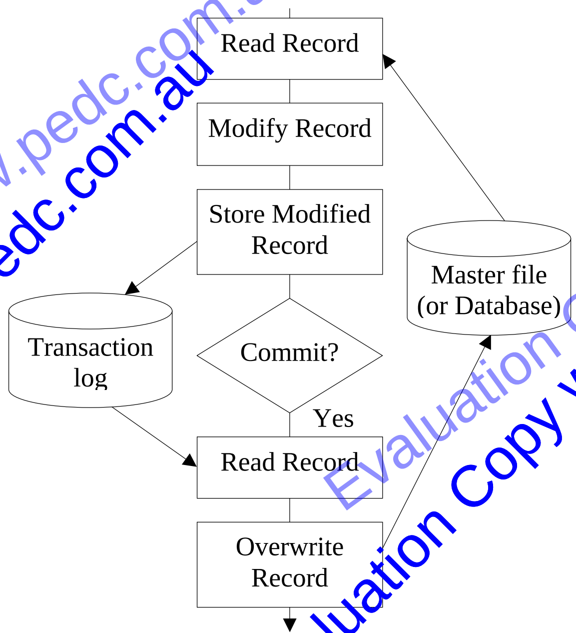


Fig 4.5
Flowchart describing modifying a record as part of a transaction where the master is not altered until the transaction commits.

yet made to the actual data in the master file. If the transaction is committed then the records in the transaction log replace or are added to the master file. If the transaction is rolled back then the records in the transaction log are not written to the master file.

2. Copies of the original unchanged data are recorded in the transaction log and then the changes are made immediately to the actual data within the master file. If the transaction is committed then nothing more needs to occur. If the transaction is rolled back then the record in the transaction file is copied back over the actual data in the master file. When new records are created as part of a transaction the transaction log must contain an entry specifying the record to delete should the transaction be rolled back.

GROUP TASK Activity
Create a flowchart to model the processes occurring to modify an existing record in the master file using the second strategy described above.

In either case the transaction log is used to enable the committing or rolling back (or even rolling forward) of events within transactions. Most current DBMSs actually record both before and after versions of the data within their transaction logs – in essence they allow implementation of both the above possibilities. This means the transaction log is really a log of all the activities performed on the data.

The most compelling reason for maintaining before and after versions of all data changes is to provide a backup of all recent changes since the last backup. The database (or master file) can be restored from the most recent backup and then the transaction file can be used to commit (or roll forward) all transactions performed since the restored backup was made. If at the time of failure some transactions were incomplete then those events that formed part of such transactions can be rolled back. Such restore operations are essentially automated within most modern DBMS and transaction processing monitor software products.

A complete transaction log is also useful during audits as it shows when, what and who performed each transaction. Utilities are available for most DBMS products that allow the transaction log to be analysed in detail. Such utilities also allow transactions in the log to be rolled back and rolled forward individually.

GROUP TASK Research
Transaction log files continually grow in size – sometimes their size can exceed the size of the actual database. Research techniques and strategies for ensuring transaction logs do not grow excessively.

Information Technology

The hardware and software forms the information technology of the system. Transaction processing systems vary enormously in both size and scope. A small database may serve just a few local users, however a similarly small database may serve many more users via the web. Larger critical transaction processing systems perform thousands or even millions of transactions daily. The hardware and software requirements vary enormously; hence in this section we shall introduce some general areas for consideration. Later in this chapter we examine more specific examples where the detail of the hardware and software can be specified more precisely.

- **Hardware**

Possible hardware for transaction processing systems includes:

- Server machines that include redundant components to improve fault tolerance. In medium to large systems multiple servers provide access to the same database. If one server fails or is taken offline for maintenance or upgrading then the other servers automatically take up the extra load. Larger government and multinational organisations commonly use mainframe computers that are able to support thousands of users and access to enormous databases.

- Storage devices with sufficient capacity and data access speeds to support the size of the database and the number of users. Commonly RAID storage is used. For transaction logs even on relatively small systems a mirrored RAID solution is common to ensure that a single failure of one drive does not result in loss of incomplete and recent transactions.
- Communication devices and transmission media able to support the number of required users and data access speeds to ensure acceptable response times. Many servers include multiple NICs to achieve higher data access speeds.
- Backup devices such as tape drives, tape libraries, CD burners and DVD burners. In some systems complete copies of the data are maintained on mirrored hard disks located in a different location to the operational data.
- Client workstations for running the client applications and interacting with the system's participants and users. These machines may include specialised collection devices such as barcode scanners, RFID readers, touch screens and magnetic ink character recognition (MICR) readers. The client machines may be dedicated devices such as ATMs or EFTPOS terminals or they may be personal computers connected via the Internet or the organisation's network.

- **Software**

Possible software for transaction processing systems includes:

- DBMS software to manage and control the transactions performed on linked databases. We discussed the operation of DBMSs in chapter 2. The DBMS runs on one or more servers and provides services to client applications to enable them to access the databases within the transaction processing system. Each DBMS includes the ability to manage transactions performed on databases under its control and includes a transaction log.
- Client applications that are installed on client workstations and provide the interface for participants to initiate and perform the system's information processes. The client applications make requests, often in the form of SQL statements, to the database servers using a client-server architecture. For larger systems that perform transactions across many servers or many systems the client applications send their requests via the transaction processing monitor.
- Proprietary software applications that are designed and developed to meet the needs of a specific organisation. Such software is common in large transaction processing systems running on mainframe machines. These software applications are written from the ground up – including providing, specifying and controlling access to data directly without the use of a DBMS. The term proprietary means the software is produced for a specific system or organisation and is generally owned by that organisation.
- Transaction processing monitors (TPMs) are software applications that coordinate the transaction processing of large transaction processing systems. *Fig 4.6* describes the general software architecture of transaction processing systems that include transaction processing monitors. These large systems commonly include many database servers that may access the same logical database or may access different databases. In addition transaction processing monitoring applications can connect to systems operated by other organisations.

Each server or system has its own resource manager (refer *Fig 4.6*) that makes available resources to the TPM. A resource manager is essentially a software product that provides an interface between the resource and the transaction processing monitor.

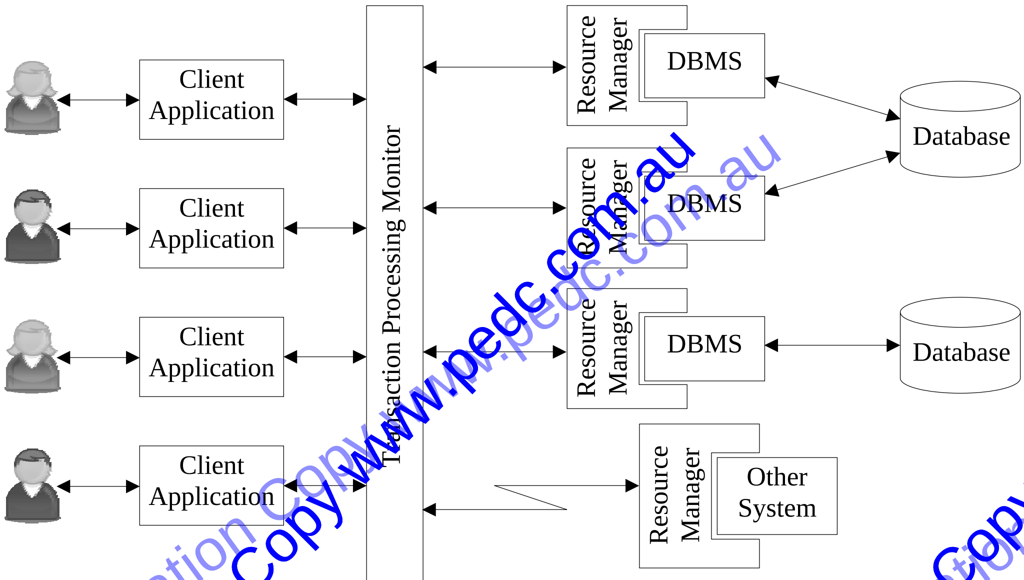


Fig 4.6
General architecture of a system that includes a transaction processing monitor.

The main task of transaction processing monitors is to ensure the integrity of transactions that include events which execute on different servers and/or systems. The TPM controls the commit and rollback of the total transaction in response to requests sent to and acknowledgements received from resource managers. Each server or system performs its own lower level transaction and reports the outcome (success or failure) back to its resource manager, who in turn communicates with the TPM. In addition TPMs are able to balance the load of transactions sent to each server. Transaction processing monitors are also known as transaction managers or transaction processing services. Examples include IBM's CICS and Encina products, BEA System's Tuxedo and Microsoft Transaction Server.

GROUP TASK Discussion
A user initiates the transfer of funds from a Commonwealth Bank account to a Westpac bank account. Discuss the role of transaction processing monitors during this transfer.

DATA INTEGRITY

Data Integrity
A measure of how correct and accurately data reflects its source. The quality of the data.

The integrity of data is critical in all transaction processing systems. Recall from our earlier work on database systems (Chapter 2) that data integrity is a measure of how correct and accurate data is compared to its source. In Chapter 2 we considered three techniques for improving data integrity, namely data validation, data verification and also referential integrity. In this section we briefly discuss examples of each technique within transaction processing systems. We then introduce the ACID properties of transactions and the type of problems they solve.

Data Validation

Data validation checks ensure reasonable data enters the system. In transaction processing systems data that is incorrect at the time of collection is likely to cause a variety of different problems when it is later used as part of transactions. There are two different types of data validation commonly performed. The first ensures the data entered is of the correct data type and format. This is generally performed by the client application. The second is more difficult as it aims to ensure the data entered is correct in terms of the business rules of the enterprise. That is, it determines if the data is correct in terms of its ability to be processed. For example when ordering a book the ISBN is often entered as a unique identifier. Data validation within the client application ensures the correct number of digits are entered. The book store's business rules require that the ISBN must exist within their database. Therefore a query must be executed to validate that this is indeed true.

> **Data Validation**
> A check, at the time of data collection, to ensure the data is reasonable and meets certain criteria.

A single data entry error that is undetected can affect numerous transactions across many organisations. For example consider a BPay reference number on a supplier's invoice that is being paid by a customer using Internet banking. Let us assume this invoice must be paid before the goods are shipped. If the BPay reference number is entered incorrectly by the customer then the total transaction will eventually fail. The consequences of this simple data entry error is costly for both the customer and also for the organisations involved in the transaction. The bank must inform the customer of the problem, however the customer is not aware of any potential problem and hence they are unlikely to check their bank messages. The supplier does not receive the funds and therefore will reissue the invoice or simply not supply the goods. The customer is not happy as they are unaware of the error and hence wonder why their goods do not arrive. Resolving the problem involves further time and cost for all parties. These issues could be resolved by validating the BPay reference number prior to the transaction commencing.

Data Verification

Data verification is used to maintain the integrity of data over time. This is a difficult task in most information systems and is rarely 100% successful. For example people and also businesses move location, change their phone numbers, credit card numbers and even change their names. Ensuring that such changes are reflected in the data is the aim of data verification processes.

In large government and commercial transaction processing systems data verification becomes an enormous undertaking. Currently in Australia there is no single unique identifier that can legally be used to identify individuals across all these systems. If such an identifier was available then it would be possible for individuals and organisations to change their details in one place and have these changes replicated to other systems. Privacy concerns prevent such practices. For example in the mid 1980s the federal government attempted to release the "Australia Card", which was to contain a unique number for each Australian citizen and resident. This number was to be used to link records between most government departments and even between commercial organisations. As a result of public outcry over privacy

> **Data Verification**
> A check to ensure the data collected and stored matches and continues to match the source of the data.

concerns the legislation was never passed. Currently tax file numbers (TFNs) and Australian Business Numbers (ABNs) are shared between many government agencies albeit with strict controls in regard to how data can be linked and used. In Australia it is illegal for private organisations to use TFNs and ABNs to link data from multiple sources.

> **GROUP TASK Discussion**
> Discuss advantages and disadvantages of widespread use of a unique identifier for each Australian citizen and resident.

Referential Integrity

In relational databases referential integrity ensures all foreign keys in linked tables match a primary key in the related table. This means a record in the primary table must exist before records can be added to the table containing the linked data. If referential integrity is not enforced then orphaned records will exist. In general such records cause significant problems when queries are executed on the database.

Within a single database referential integrity is enforceable and hence problems simply cannot occur within the database. When many databases are involved or identifiers are being entered by users then problems are inevitable. Data validation and verification issues can affect referential integrity. For instance, entering an incorrect BPay reference number means that the primary records held in the various organisations' databases cannot be linked to the customer's payment. The Australia Card aimed to provide a primary key for each Australian that could be used as the foreign key in many linked databases. Both systems are attempting to use a unique identifier in an attempt to enforce referential integrity across multiple databases.

> **GROUP TASK Discussion**
> Brainstorm real world examples of data validation and data verification that aim to improve the referential integrity (and therefore the data integrity) of databases.

ACID Properties

ACID is an acronym for atomicity, consistency, isolation and durability. The aim is to ensure all transactions comply with these four properties. They ensure that transactions are never incomplete (atomicity), the data is never inconsistent (consistency), transactions do not intrude or affect each other (isolation) and that the results of a completed transaction are permanent (durability). All these properties combine to ensure the integrity of all data is maintained before, during and after each transaction.

To illustrate each of the ACID properties let us use an example transaction – making an airline reservation using a credit card. This transaction includes the following general sequence of events:

1. Reserve a seat on a specific flight.
2. Process and approve credit card payment.
3. Issue and record ticket details.

- **Atomicity**

To be atomic all events within a transaction must complete successfully or none at all. If any single operation fails then the entire transaction is aborted. This involves rolling back all events completely so that the data is returned to its original state. If all events are successful then the transaction is committed, which means the data changes are made permanent or durable.

In our airline transaction imagine what would occur if just one operation failed but the others were committed. If no seat were reserved then the passenger would arrive with a paid ticket but with no available seat. If the payment is not processed and approved then the passenger receives a seat and ticket for free – great for the passenger, but not so good for the airline. If no ticket is issued or recorded then the passenger and airline have no record of the transaction resulting in the passenger being refused a seat.

- **Consistency**

The consistency property ensures transactions take data from one consistent state and then when the transaction completes the data is left in a consistent state. For a single event on a single database this is enforced using referential integrity and validation rules. When the transaction includes many events and spans many databases or systems then consistency must apply across all these databases and systems.

In our airline transaction a business rule is likely to require the total number of reserved seats to be equal to the number of tickets issued. If a seat is reserved but does not result in a ticket being issued then the data is inconsistent in regard to this business rule. Many other rules are also likely, such as, a customer must be assigned to each reserved seat, all tickets must be paid for and each ticket must be assigned to a specific flight and passenger.

- **Isolation**

Transactions must process data without interfering with or being influenced by other transactions that are currently executing. In effect each transaction logically executes in isolation to all other transactions. During the processing of a transaction the data is often placed in an inconsistent state. For example when transferring funds between accounts, money is debited from one account and then credited to another account. After the debit but before the credit the data is in an inconsistent state. This inconsistent state should not be exposed to other transactions. Furthermore the records involved should not be available for other transactions to change until the transaction is completed. If the isolation property is not observed then queries will return inconsistent results and other transactions will process with potentially erroneous data.

In small systems where only one transaction executes at a time the isolation property is simple to achieve as one transaction completes before the next commences. If many transactions can execute at the same time then the solution is more involved. However even the largest transaction processing systems must ensure their method of implementing the isolation property results in the same effect as executing each transaction sequentially.

When multiple transactions can execute concurrently all data involved in a transaction must be locked such that other transaction processes cannot alter it. We discussed record locking strategies used by DBMSs in chapter 2 – these strategies are also used within transactions that span multiple databases and systems. Note that locking does not alter the actual data, rather it prevents other operations from changing the data. As a transaction is committed the actual data is altered. Significantly other processes are aware that a record has been locked by another transaction. Therefore other transactions must wait for the lock to be released before they proceed.

Record locking, transaction logs and the “two-phase commit” nature of transactions all influence each other and combine to implement the isolation property. The term “two-phase commit” refers to events being performed temporarily (phase one) during a transaction and then being committed (phase two) if the transaction completes successfully. The first phase is recorded in the transaction log and also involves the record being locked. The second phase alters the actual data permanently and releases the record lock.

Consider our airline transaction example. Imagine the isolation property is not present and a single seat remains available on a flight. Many passengers are now able to simultaneously reserve this single seat successfully, as long as each transaction commences prior to committing one of the other transactions. Furthermore they will go on to pay and be issued with a ticket. When passengers board the flight the airline will discover there are more passengers than available seats.

- **Durability**

Durability ensures that committed transactions are absolutely permanent. Theoretically this means that even if the whole world crashes the changes made by the transaction will be OK. In real systems durability ensures that during a commit the results are actually written to some physical storage device. Notification of a successful commit can therefore be reasonably relied upon.

At first it may seem that executing an update query when committing will ensure durability of the changes, however in many systems data is held in RAM for a period of time and is only written to secondary storage as required. Such systems improve performance, however if power is lost then the contents of RAM is permanently lost. Therefore durability specifically requires all changes to be written to permanent or secondary storage before the transaction is truly committed.

In our airline example, imagine an example transaction is apparently committed successfully. Now say the durability property is not present within the “issuing and recording ticket” event. Suppose the system fails and this operation is not recorded. When the passenger goes to board the flight their ticket will not exist on the system. However inconsistencies will be present as a reservation will exist for the passenger and a record of payment also exists. Resolving this issue will be costly in terms of time and also in terms of inconvenience for the passenger.

HSC style question:

Define the term *transaction* and explain how data integrity is maintained during processing of transactions.

Suggested Solution

A transaction is a unit of work composed of a sequence of events. All actions performed by all events must succeed for the transaction to be committed permanently. If any single event within a transaction fails then all events within the transaction are aborted or rolled back. Commonly each event within a transaction alters data within a database.

Whenever data is altered the potential exists for inaccuracies to be introduced and the integrity of data to suffer. Transactions avoid such possibilities through their ACID properties. Atomicity ensures a transaction succeeds completely or fails completely. Consistency ensures each transaction takes the data from one consistent or correct state to another consistent or correct state. This means inaccuracies or data integrity issues are only possible during processing of a transaction. This possibility is dealt with by the isolation property. This ensures data changes are not available to other transactions until they have been committed. The durability property ensures all changes made by all events occurring within all committed transactions are permanently written to storage. This increases data integrity as it guarantees the consistency of the data after each transaction completes is maintained permanently.

SET 4A

1. Which of the following best describes a transaction?
   (A) An event that alters or creates a record within a database.
   (B) Multiple events that must all succeed or all must fail.
   (C) A system that controls the execution of many transactions across many databases or systems.
   (D) A process that alters data in different records, databases or systems.
2. Transaction processing using computers first emerged during the:
   (A) 1980s
   (B) 1970s
   (C) 1960s
   (D) 1950s
3. A transaction log contains:
   (A) details of the data added or updated during processing of transactions.
   (B) details of the original data prior to it being updated by transactions.
   (C) sequential copies of the data within the master file.
   (D) Answer A and/or B
4. Manual transactions performed by clerks are often well suited to automation because they:
   (A) are boring and repetitious for participants to perform.
   (B) follow a strict predefined sequence of rules.
   (C) can be performed as batch processes.
   (D) commonly include just one operation that alters data.
5. Bank customers become participants when they:
   (A) write a cheque.
   (B) receive a statement in the mail.
   (C) withdraw cash from an ATM.
   (D) All of the above.
6. Examples of TPMs include:
   (A) SQL Server, Oracle, DB2
   (B) SQL Server, CICS, MTS
   (C) Oracle, Encina, Tuxedo
   (D) CICS, Tuxedo, MTS
7. The data needed during commit and rollback processes is stored within the:
   (A) transaction log.
   (B) master file.
   (C) operational database.
   (D) data source.
8. Which of the following is the most significant task performed by TPMs?
   (A) Manage access to many remote DBMS servers within an enterprise system.
   (B) Provide an interface between client applications and resource managers.
   (C) Manage and control transactions whose events span multiple databases and/or systems.
   (D) Force all events within a transaction to be permanently committed.
9. Over time existing data becomes less and less accurate. Which of the following is undertaken to improve this situation?
   (A) Data verification.
   (B) Data validation.
   (C) Referential integrity checks.
   (D) Ensure transactions adhere to the ACID properties.
10. Transaction A reads data whilst transaction B is executing. Transaction B is rolled back, however transaction A commits. It is later determined that transaction A has introduced inconsistencies into the data. Which ACID property is NOT present?
    (A) Atomicity
    (B) Consistency
    (C) Isolation
    (D) Durability

11. Define each of the following terms?
    (a) Transaction (c) TPM (e) Data validation
    (b) Data verification (d) Referential integrity (f) Data integrity
12. Recommend suitable data validation techniques when collecting each of the following?
    (a) Exam marks that are out of 100.
    (b) A pair of dates, where the first date must be prior to the current data and the second must be at least 1 week after the first date.
    (c) Adding a product and required quantity to a customer's order.
13. Outline the history of computer based transaction processing systems.
14. Explain how transaction processing systems implement the ability to commit or rollback transactions.
15. (a) Why are most manual transactions well suited to automation? Discuss.
    (b) What data integrity issues are resolved when all ACID properties are enforced? Discuss.

Pages 381 to 400

Not included in this Evaluation Copy

Historically batch processing was the first type of transaction processing. In the early days of computers all input was via punch cards – this included the actual program code as well as the data to be processed. Each card was manually punched by an operator in preparation for input (see *Fig 4.22*). Completed stacks of punch cards were physically loaded into the computer and processed sequentially. In these early days online real time processing of multiple transactions was simply not possible. The hardware performed a single task at a time and the output was stored sequentially on magnetic tape. As a consequence problems associated with multiple transactions accessing the same data simultaneously did not exist – that is, the isolation ACID transaction property was simply not an issue. Furthermore the processing resources were limited and also costly, therefore batch jobs were scheduled to maximise the use and efficiency of precious processing resources.

Fig 4.22
Operators using key punch machines to create punch cards for batch processing in the 1960s.

Today batch processes are generally performed in parallel with other processes. As a consequence ACID properties must be observed during most batch jobs, including the isolation property. Consider the scenario where a number of different organisations in various locations are processing transactions that access the same data. For example the same credit card number may form part of batch transactions in locations in many different countries. If these transactions happen to overlap then without the ACID isolation property data integrity problems will result.

GROUP TASK Discussion
In old systems each transaction in a batch is performed sequentially. Are any of the ACID properties required within such systems? Discuss.

The processing resources of all computer systems are limited, therefore batch processing even today is scheduled to ensure that each set of batch transactions can complete in a timely fashion. This means many batch processes are scheduled to occur during evenings or weekends when real time processing requirements are lowest. Such scheduling not only ensures CPU processing resources are available, it also reduces the wait time for transactions as it is less likely that other transactions will be simultaneously requiring access to the same data.

Batch transactions that are restricted to a single organisation can be processed offline. This was the normal situation prior to the widespread use of high-speed communication links between organisations. Consider a company's bill generation, all data originates from one single organisation's database. In this case a static snapshot copy of the database can be used to generate bills. Any sales that occur during the bill generation process are not included until the next batch bill generation process occurs.

User interaction with batch processes is restricted to input prior to the commencement of processing and to deal with problems after batch processing completes. Furthermore employees rather than customers commonly initiate batch processing. As a result the design of user interfaces for batch processing is different – they are designed for rapid entry. Often such screens accept numerically coded input via the keyboard's numeric keypad or a barcode scanner. Screen elements designed specifically for mouse input are avoided and keyboard shortcuts are available.

In this section we examine three examples of common batch transaction processing. We examine the processes occurring to clear presented cheques, the generation of bills or invoices and credit card transactions where they appear to be real time but are often processed in batches.

CHEQUE CLEARANCE

Cheques are a paper-based system that prior to the widespread use of computers were the primary method for exchanging funds over long distances. Since the 1970s credit cards and EFTPOS have steadily and significantly cut the total value and number of cheque transactions. In 1994 the daily value of cheque transactions in Australia was approximately $25 billion, in 2004 the daily value had reduced by four fifths to just $5 billion. Despite this reduction, cheques are still expected to form a significant proportion of total financial transactions for the foreseeable future.

A cheque is essentially a promise by a payer (the person or organisation writing the cheque) to pay the payee (the receiver of the cheque) some amount of money. Cheque clearance processes commence when a payee deposits a cheque at their local bank branch. The purpose of cheque clearance processes is to expedite the secure transfer of funds from the payer's account to the payee's account via the banking and clearance network.

However cheques are promises and promises can and are broken. In the case of cheques this regularly occurs when the payer has insufficient funds in their account to cover the value of the cheque. Other problems can also occur, such as forged, unsigned, illegible, altered, lost and stolen cheques. Cheque clearance processes include safeguards to identify and deal with such problems.

Prior to the late 1990s cheque clearance processes did not involve any electronic communication between banks and cheques took 5 or more working days to clear. Today cheque details are exchanged electronically between banks and most cheques clear into payee's accounts in 3 working days. Compared to totally electronic transfers, such as EFTPOS and direct deposit, 3 working days is an eternity. This reality is a consequence of the manual processing inherent in a paper-based system. The value of the cheque must be determined, commonly using scanners and OCR software or in some cases it is manually entered. Signatures and dates require verification and the actual paper cheques are physically exchanged between banks. Until the 1990s all cheques were physically returned to the payer's branch for clearance. Today banks operate their own central facilities that perform cheque clearance processes for many branches.

Financial transaction clearance procedures in Australia are legislated by Government and controlled by the Australian Payments Clearing Association Ltd. (APCA). All major banking and financial institutions, including the Reserve Bank, are APCA members. APCA operates a number of clearance systems, including the Australian Paper Clearance System (APCS) used primarily for the clearance of cheques.

Typical steps for cheque clearance in Australia include:

1. Payee Fred receives a cheque in the mail from payer Freda (who has an account with DEF bank). Fred deposits the cheque into his account at his local branch of ABC bank.
2. In some cases the teller immediately passes the cheque through a MICR (Magnetic Ink Character Recognition) reader to determine the payer's BSB (Bank State Branch) number and account number and the teller

Fig 4.23 MagTek's Mini-MICR cheque reader.

manually enters the value of the cheque. When this occurs the funds are immediately credited to Fred's account as unavailable funds. Usually these funds immediately begin to accumulate interest in Fred's account. More commonly the cheques, together with the deposit slip, are simply filed for later batch processing.

3. During the afternoon all cheques deposited at local branches of ABC bank are physically transported to a central outwards processing facility operated by ABC bank. Some smaller banks share such facilities with larger banks.
4. At ABC bank's outwards processing facility high speed MICR (Magnetic Ink Character Recognition) readers read payer BSB (Bank State Branch) numbers and account details from each cheque. Scanners automatically determine the value of the cheque and the details on deposit slips. Each cheque is encoded with its own unique ID so it can be traced should it later be dishonoured or stopped. Most banks also print the cheque value on the cheque using MICR printers. Payee accounts are credited with funds if this has not already occurred at the branch. Based on the BSB numbers, cheques are automatically sorted into bundles destined for different banks together with the total value of each bundle. *Fig 4.24* shows IBM's 3890 sorter which is able to read MICR characters (*Fig 4.25*) and sort up to 2400 cheques per minute. Note that completion of this batch process provides electronic records of all cheques deposited into payee accounts operated by the bank.

Fig 4.24
IBM 3890 high speed cheque sorter includes MICR reader and optional scanner.

5. Each bundle of cheques is transported to a central check clearing house operated by APCS. Appointed representatives of all banks exchange bundles of cheques. In addition the net difference between exchanged bundles is calculated. For example the representative from ABC bank may hand the DEF bank representative cheques totalling $2.2 million, whilst DEF bank hands ABC bank bundles of cheques totalling $2.5 million. In this case the net difference of $300,000 is transferred from ABC bank to DEF bank. At this stage all cheques are now under the control of the payer's bank. In our example Freda's cheque is now in the hands of her bank – DEF bank.

1234567890
a BSB
b Amount
c Domestic
d Dash

Fig 4.25
Standard MICR characters.

6. Bundles of cheques are now physically transported to the central inwards processing facility of each bank – Freda's cheque goes to DEF bank's inwards facility. Currently most facilities are within major cities such as Sydney and Melbourne. The cheques commence being batch processed. Each cheque again passes through a MICR reader and scanner. The scanner determines the value of each cheque whilst the MICR reader determines the account. For each cheque, the system ensures there are sufficient funds in the payer's account, verifies the authenticity of the cheque and debits the value of each cheque from the payer's account. Problem cheques are diverted for manual examination. Cheques where there are insufficient funds or other problems are dishonoured. The ID encoded by the payee bank is used to determine and inform the payee's bank of such problems.

In the past cheques were sorted into individual branch bundles and physically transported to branches for final batch processing. Today account details and images of account holder signatures are available online, therefore verification can now take place centrally via secure communication links. It is the removal of the need to physically transport cheques back to their branch of origin that has reduced clearance times from 5 days to the current 3 days.

> **GROUP TASK Discussion**
> Each cheque passes through two distinct batch processes. Identify and describe the operations performed during these batch transactions.

> **GROUP TASK Discussion**
> Identify the information technology, data/information and participants involved in the cheque clearance process.

> **GROUP TASK Research**
> Many countries, including Australia, are analysing systems that digitise images of entire cheques – in the banking industry this is known as "cheque truncation". Some countries have already implemented such systems. Research advantages of such cheque truncation systems

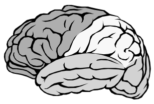
BILL GENERATION

In many systems the generation of bills or invoices is well suited to batch processing. When orders for products or records of services provided are already within the system then no extra data collection is required prior to generating invoices. No user interaction is needed and multiple invoices are usually generated at the same time. Often bills are generated during times when the resources of the system are not being used – commonly during the night. Consider telephone, electricity, gas, rates and other regular household bills. The data exists within the organisation's database and therefore batch processing can be used to generate bills.

Even small businesses that process small numbers of orders each day use batch processing. The orders are entered as they are received throughout the day and then in the afternoon all the day's invoices are printed as a batch job. The orders are packed manually using details from the printed invoices. Each order is then dispatched together with the invoice. The invoicing database schema we produced in chapter 2 when describing normalisation is typical of such a system (refer *Fig 2.70* on page 185). This database would be queried to return all invoice details for the current day. This query is then used as the record source for a report that generates and prints the day's invoices.

> **GROUP TASK Discussion**
> Analyse the Invoicing database created in chapter 2 to determine the data and processes required to generate invoices for the current day.

Apart from the relatively static product details and prices the data required to generate each invoice is largely independent of the data on all other invoices. This data independence means that invoices can be generated in any desired order and more significantly multiple invoices can be generated simultaneously. This characteristic is particularly significant for large systems that generate many thousands of bills. To generate say monthly telephone bills requires reading each customer's address details and records of all the calls made within the billing period. The batch process does not need to access, update or create data in any other system. Also during processing no data is updated or created within the telephone company's database that is accessed during the generation of any other customer's phone bill. This processing independence means parallel processing can be used to drastically reduce the total processing time required.



Consider the following:

For large batch systems, where many thousands of bills are generated in a single job, it is common to make a snapshot copy of the live data. This snapshot is an offline copy of the actual data as it was at the end of the billing period – maybe the end of a year, quarter, month, week or even the end of a day. The online version of the database continues to operate without its performance being degraded by the batch processes, and the batch processes are not interrupted by the online processes. Because of the independence of the data and processing the snapshot copy can be split into different parts that are physically stored and batch processed in parallel on different storage devices and using different CPUs (refer *Fig 4.26*). This distributed processing strategy reduces processing time significantly – if a batch job takes two hours to complete on one machine then it will take approximately half this time if two machines are used. Such systems use high-speed digital printers that link with automatic folding and envelope insertion devices.

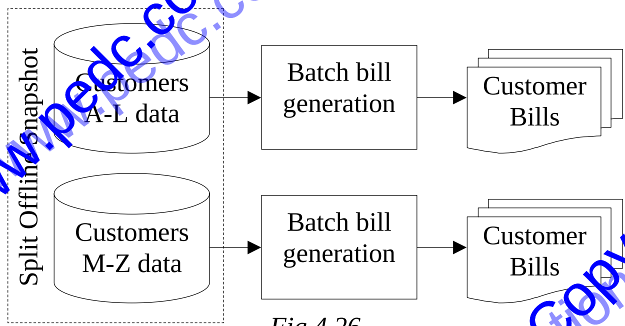


Fig 4.26
Parallel batch processing using a split offline snapshot copy of the online database.

The time taken to retrieve records from secondary storage is a significant limiting factor in terms of improving the performance of large systems. Splitting and physically storing parts of the database on different storage devices is one technique that improves performance. Another significant technique used by large batch systems is sequentially accessing data. This involves accessing data in the order in which it is physically stored on the disk. In the past, data was physically stored sequentially on tape and hence it was either impossible or extremely inefficient to read the data randomly – rewinding and fast-forwarding the tape takes time. Sequential access was a necessity rather than a choice if jobs were to complete in a reasonable amount of time – even for small batch jobs. Today the read/write heads within hard disks are able to quickly jump directly to required records, however this still takes time. Furthermore hard disks read data in complete sectors and commonly multiple adjacent sectors are also read. This data is stored in the drive controller's cache. If random access is used then much of the data that is physically retrieved is not actually processed. If sequential access is used then all data retrieved is subsequently processed and the movement of read/write heads is minimised. Note that significant performance gains are only possible when the transactions processed are independent of each other and the data they access is physically stored sequentially. Transaction processes that use retrieved data as the criteria for searches and that write data require careful analysis and design if the advantages of sequential access are to be maximised. For instance, the order in which processes are performed can be significant or it may be more efficient to remove an operation from a transaction and perform it separately on all the data. For independent processes, such as those required for bill generation, the ACID properties can be relaxed somewhat in order to improve performance.

GROUP TASK Discussion

Many people now receive bills online via the web, in general these bills are still the result of batch processing. Why not generate these bills online and in real time? Discuss.

CREDIT CARD TRANSACTIONS (REAL TIME OR BATCH?)

From the customer's perspective credit card transactions appear to be processed in real time. The customer presents their card to the sales assistant who then swipes the card through an EFTPOS terminal and has the customer sign a receipt. If the signature on the card matches the signature on the receipt then, from the customer's perspective, the transaction is complete. On the web, customers enter their credit card details into a secure web page and then within seconds the transaction is approved – it appears the entire transaction is complete. In reality many credit card transactions are batch processed during the evening following the sale, whilst some are actually completed in real time. Note that a credit card transaction is not complete until the funds have moved into the merchant's account.

All credit card transactions involve at least four significant parties. Customers who are credit card holders, merchants who are generally retailers, card issuers who manage the customer side of credit card transactions and acquirers who manage the merchant side of transactions. Most acquirers and issuers are banks who share the expense of operating the network and technology between issuers and acquirers predominantly via the MasterCard and Visa systems. Let us consider the general sequence of events that occurs to process credit card transactions (refer to *Fig 4.27*):

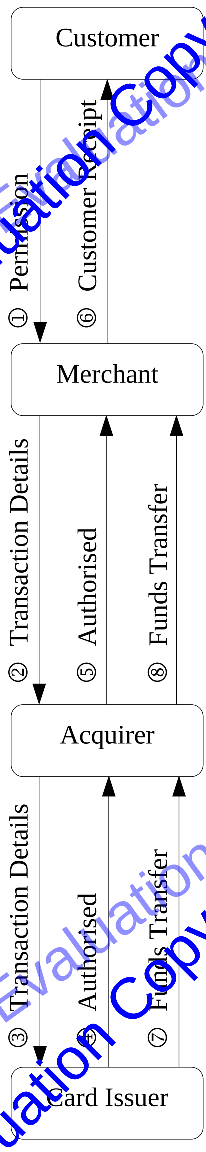


Fig 4.27 Communication during a credit card transaction.

① Customer gives merchant permission to access credit in their account to pay for goods or services. For "card present" transactions handing over the card and signing verify that permission has been given. For "card not present" transactions, such as telephone and mail order, the verbal or written order and credit card details are sufficient verification of permission.

② Merchant creates and transmits transaction details manually or electronically to their acquirer. This can occur via an EFTPOS terminal, manually by written voucher or over the Internet via a payment gateway. For larger value manual transactions approval (steps ③, ④ and ⑤) are performed over the phone.

③ The acquirer receives the transaction and determines the card issuer. The transaction details are then forwarded electronically to the card issuer.

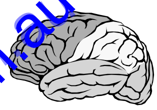
④ The card issuer checks the customer has sufficient credit remaining to cover the transaction and reserves these funds. An authorisation code is sent back to the acquirer.

⑤ The acquirer receives the authorisation code and electronically forwards it back to the merchant. On EFTPOS terminals the word "APPROVED" is commonly displayed. For manual transactions approval is given over the phone.

⑥ Merchant receives approval message generates a receipt and hands it to customer. If customer is present then they first sign the merchant's copy of the receipt and the merchant verifies the signature against the signature on the card. The receipt includes a unique number that identifies the transaction within the merchant, acquirer and issuer's systems.

⑦ The card issuer transfers the funds out of their account and forwards the funds to the acquirer. Often many transactions are batch processed together hence a single large transfer takes place together with details of individual transactions.

⑧ The acquirer deposits the value of each of their merchant's transactions into each merchant's account. In most cases this occurs each evening to finalise the days transactions.



Consider the following:

The above sequence of steps occurs for all credit card transactions however there are many different systems that perform these steps at different times and perform some or all of the steps as batch processes. In some cases other organisations are involved that relay data between merchants and acquirers or to perform processing on behalf of merchants.

Let us consider some typical examples and highlight when real time processing and when batch processing is used.

- Retail EFTPOS terminals supplied and connected directly to a particular bank use a combination of real time and batch processing for credit card transactions. When a customer's credit card is swiped the terminal communicates with the bank (acquirer) via a telephone line to authorise the transaction in real time. The bank transmits a reference retrieval number (RRN) back to the EFTPOS terminal and the terminal displays "APPROVED PLEASE SIGN". The customer signs the receipt and the retailer verifies the signatures on the card and receipt match. If the signatures do not match then the transaction is reversed – this reversal is another transaction sent to the acquirer.

  At the close of business each day the EFTPOS terminal "settles" with the acquirer bank. The settlement process transmits details of all transactions to the bank. The bank then batch processes all transactions during the evening resulting in the funds (less any bank charges) being deposited into the retailer's merchant account.

- Most retailers now use EFTPOS terminals for their credit card transactions as described above, however manual systems are still available as a fall back should the EFTPOS terminal or link to the bank fail. Using a manual system the retailer manually takes an impression of the customer's card on a voucher. The voucher is manually completed by the retailer and then signed by the customer Each retailer has a floor limit. If the total value of the transaction is above the floor limit then the retailer telephones the bank for manual authorisation. If authorised the bank reads out an authorisation number, which is manually written on the voucher. Each voucher includes the original, which is later submitted to the bank, a copy for the customer and a copy for the merchant.

  At the close of business the retailer completes a merchant voucher that includes the total number and value of all vouchers. The merchant summary together with the original of all vouchers is then deposited at the retailer's local bank branch (acquirer). The vouchers are batch processed by the bank during the evening.

GROUP TASK Discussion
Why do you think the batch settlement process is performed, why not simply complete each transaction in real time as it occurs? Discuss.

- Some retailers are authorised by their bank to accept mail, phone or fax credit card orders. These are known as MOTO (mail order telephone order) merchant accounts. Banks scrutinise retailers more thoroughly to verify that they are trustworthy and honest before MOTO merchant accounts are approved. Once approved the retailer is able to initiate credit card transactions without the card actually being present – just the credit card number and expiry date is required. The details of the transactions are manually entered into the EFTPOS terminal or can be manually written onto a voucher. As less information is available to verify each transaction the retailer must agree to accept a higher level of risk should transactions be disputed. The transactions are processed similarly to above, however banks often charge higher rates compared to "card present" transactions.
- Internet credit card transactions for large volume businesses are usually processed in real time. Commonly the merchant's website collects details of the purchase, such as products and prices. The website then directs customers to a payment gateway which completes the actual financial transaction such that the funds are moved immediately from the customer's account into the merchant's account. This transfer involves both the authorisation and funds transfer steps occurring simultaneously and immediately.

> **GROUP TASK Discussion**
> Banks view Internet credit card transactions as high risk. Propose reasons why this is the case? Does real time processing reduce the risk? Discuss.

- Other Internet credit card transactions, particularly for smaller businesses, are actually processed manually using the retailer's existing EFTPOS terminal and MOTO merchant account. The credit card details and the details of the purchase are transmitted securely to the merchant without any interaction with banks. The merchant then initiates the transaction manually via their EFTPOS terminal. Such transactions are settled, along with any in-store purchases, during the evening using batch processes.
- Businesses that charge customers on a regular basis use batch processing. In this case the business creates a file containing the details of multiple transactions. This file is uploaded to the merchant's acquirer bank where it is batch processed during the evening. The business must hold an authority from each customer to perform each transaction. Such batch systems are used for purchases that require regular payments, for example topping up toll card accounts, making loan repayments and for payment of telephone, electricity, rates and other regular bills.

> **GROUP TASK Discussion**
> The above system does not use real time processing at all. The transactions are entirely batch processed. Discuss advantages for the customers, merchants, acquirers and issuers.

- Private companies now provide EFTPOS services and dedicated terminals to retailers. Often these EFTPOS terminals connect via the Internet and are operated by or connected to Internet payment gateways. Transactions performed on such terminals are processed in real time in much the same way as web transactions performed via payment gateways.
- Companies such as PayPal offer credit card processing services that do not require merchants to have their own merchant account with an acquirer. Rather the company uses their own merchant account and acquirer to process transactions on

behalf of other merchants. These systems generally cost more per transaction and hence are used by individuals and businesses that process credit card transactions infrequently.

GROUP TASK Discussion
Analyse each of the above systems and identify where real time processing is being used and where batch processing is being used. Discuss the appropriateness of each type of processing for the given system.

HSC style question:

BigBizzCorp is a medium sized business which uses a traditional batch payroll system to produce weekly payslips for each of its 200 employees who work in one of 10 departments.

Each day when the employees come into work, they ‘clock on’ by locating their employee time card and punching into a special clocking system, which prints the current time on their time card in today’s position. At the end of the day, the employee punches their time card again to allow it to print the time they have just finished for the day.

At the end of each week, the paymaster collects these 200 time cards, and enters the start and end times for each day for each of the employees into the Payroll system.

When the weekly payroll is run, a single payslip is produced for each employee showing their hours worked for this week together with their pay, taxation and superannuation details. An overall summary of the weekly payroll is also produced for use by management in their budgetary processes.

(a) The data entry screen for entering each employee’s start and end times into the batch payroll system is reproduced below:

Weekly Payroll for the week ending ../../..

Department: --

Employee number: ------ Employee Name: --------------

	Start Time	End Time	Total Hours
Monday:	--:--	--:--	--:--
Tuesday:	--:--	--:--	--:--
Wednesday:	--:--	--:--	--:--
Thursday:	--:--	--:--	--:--
Friday:	--:--	--:--	--:--

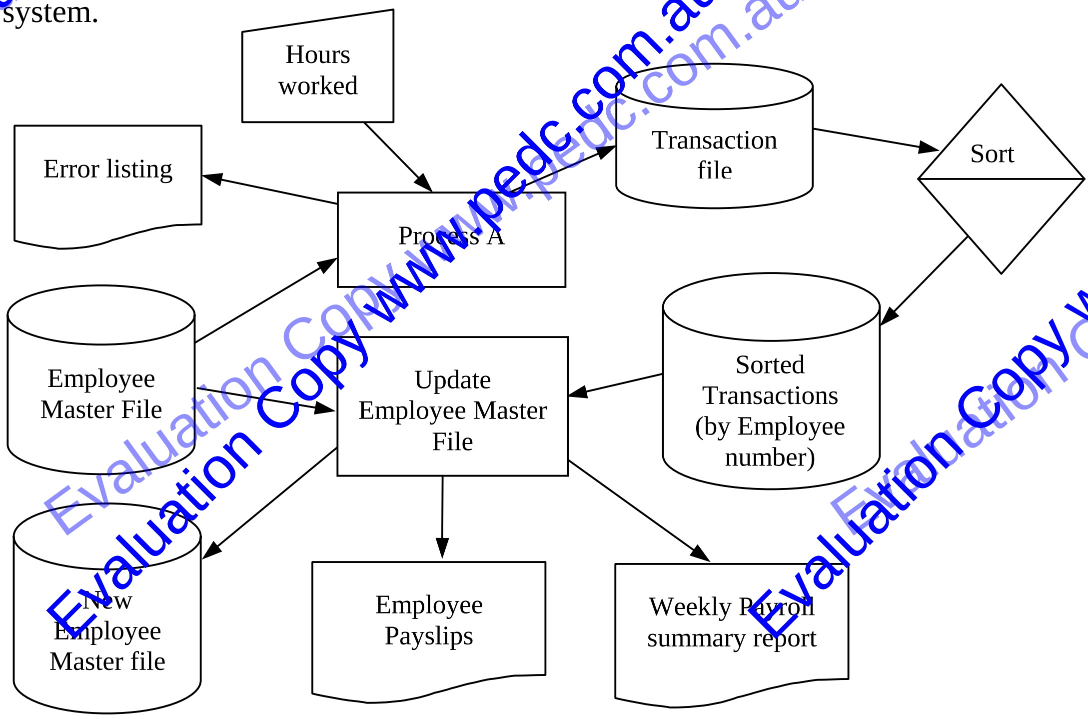
Total Hours for the week: --:--

[Done] [Next]

(i) Identify fields on the above screen where data would be entered directly by the paymaster. Explain how the remaining fields would be populated during data entry.

(ii) Propose suitable validation processes that could be performed on the data entered through this screen. Justify your responses.

(b) The systems flowchart originally created during the development of the above batch payroll system is reproduced below. The flowchart diagrammatically represents the steps performed by BigBizzCorp's batch payroll processing system.



(i) Explain the processing likely to be occurring within Process A. Refer to the output produced, including the error listing, as part of your response.

(ii) Describe the method of data access being used each time a file is read from or written to within the above system.

(c) Analyse the strengths and weaknesses of the current batch system and assess the effects of altering this system to a real-time system.

Suggested Solution

(a) (i) The paymaster enters the week ending date and then enters just the start and end time for each day for each employee. The department, employee number and employee name being populated sequentially from the Employee Master File – the data entry process progressing to the next employee each time the Next button is selected.

The Total Hours are generated in real time once each pair of start/end times has been entered. Similarly the Total Hours for the week would be calculated by summing the Total Hours fields – this field being updated as each day's times are entered.

(ii) Fields to be validated include:

- Week Ending – the date must be a valid date, less than today's date. If data is entered incorrectly, the management summary report will have the incorrect date on it.

- Department number – must be a valid department within the company (eg between 1 – 10). If data is entered incorrectly, the management summary report will bill the payroll to an inappropriate or non-existent department.
- Employee number - must be a valid employee number within the company (eg between 1 – 200). If data is entered incorrectly then during the update process, the pay and hours will be accredited to the incorrect employee, or there will be no such employee to match to the transaction, and the update will fail.
- Each day's start and end times – must be a valid time, during valid working hours of business and with end time greater than start time. If data is entered incorrectly, the calculated hours worked on that day will be incorrect and the pay will not be correctly calculated. A common error is to transpose the start and end times, and the validation should check for this.
- Total hours for the week – must calculate to be within a reasonable expected range (say 0 – 60 hours). Users could be prompted to check or re-enter times if the total exceeds the expected maximum.

(b) (i) Process A occurs each time a set of start and end times is entered for each employee record retrieved from the Employee Master File. First validation is performed on each field. If the transaction is correct then it is written to the transaction file – the data would likely include the employee number, the day and a pair of start and end times. If an error is encountered then a line is printed – depending on the operation of the UI, errors could include unknown employee numbers, departments or unreasonable daily total hours. Process A occurs for each transaction, hence all correct transactions are written to the transaction file with an error listing being printed of all problem transactions.

(ii) All data access within the system is sequential. Process A reads each employee record from the Employee Master File one after the other. Process A also writes transactions one after the other so that the transaction file contains a sequence of records where each record is comprised of an Employee Number, day, start time and end time. The sort process arranges these into order based on Employee Number. These sorted records being written sequentially to the Sorted Transactions file.

The Update Employee Master File process would read an employee record (from the Employee Master File) and a sequence of transaction records (from the Sorted Transactions file) that match the employee number. These records are joined, a payslip is printed and a new record for that employee is written to the New Employee Master File. This process repeats sequentially for each employee in the Employee Master File.

(c) Possible strengths include:

- Only a single computer is required.
- Only a single direct user is required.
- Data required only at the end of each week so system is well suited to batch processing.
- Simple system that reflects manual processes.
- No real time queries required which suits batch processing.

Possible weaknesses include:

- Dedicated data entry person required.
- Time delay to process payroll.
- No real time queries are possible.

- Difficult to alter to meet new requirements – i.e. Designed for the specific payroll task.

Effects of altering to a real-time system:

- The Employee master file would need to be rewritten as a direct access file. This would allow queries at any time by management as to who has been absent this week, or how many hours have been worked so far.
- It would be possible to allow employees to enter their own times directly through an interactive system that creates a transaction each time an employee logs on to start or end their day. This eliminates the need for a data entry person with associated costs, possible bias or errors in the data entry.
- Validation can be done instantly at the time of data entry by the employee, without the need for a clerk to look back through the transactions and correct them if they are identified as being in error.

Comments

- In part (a) there are various different ways to interpret the operation of the screen. As this is a batch system, perhaps each employee is displayed one after the other and the user has no control over this order – meaning only start and end times are entered directly as in the above answer. Or possibly the employee number is entered which causes that person's name to be displayed ready to enter their start and end times. Or possibly department could be entered so a sequence of employee's within that department is presented.
- In part (a) (i) it is likely that the total hours data, which is calculated from the start and end times, would not be written to the transaction file. This data is calculated on the screen for use by the data entry user – it performs a data verification role.
- In part (a) (ii) the validation processes described could include checking the employee or department exists within the Employee Master File.
- On the systems flowchart a printed error listing is created during data entry of start and end times. Although this is possible, today it is more likely that during data entry such errors would be displayed on the screen.
- As the master file is updated problems can occur. During batch processing these problems are generally directed to an error log (usually a file). The systems flowchart included in the question does not detail how users are informed of errors that may occur during the Update Employee Master File process.
- In (b) (ii) it is possible to interpret the Process A read from the Employee Master File as random access as the data entry screen can be interpreted to be looking up employees one by one based on the user's employee number inputs. If this is the case then the system is not enforcing the employee order as would occur if access were sequential. Furthermore the transactions are sorted prior to further processing. This implies that Process A does not collect, create and then write the transaction records in the order required by the Update Employee Master File process.
- In part (c) there are many other possible strengths and weaknesses of batch systems, and effects of altering to real time processing that could be discussed. Notice the three parts to the question – strengths of batch processing, weaknesses of batch processing, and effects of altering to real time. In a Trial or HSC examination equal marks would likely be allocated to each of these three parts.
- In a Trial or HSC examination Part (a) (i) and (ii) would likely attract 3 marks each. Parts (b) (i) and (ii) would attract 4 marks each and part (c) would attract a total of 6 marks. Therefore this question would form a complete Transaction Processing Systems questions worth a total of 20 marks.

SET 4C

1. In most batch processing systems the transaction file contains:
   (A) the results or changes made to the master file after transaction processing.
   (B) the data required to process transactions.
   (C) a copy of all data that has been altered or added to the master file.
   (D) details of all transactions that have been successfully committed.

2. User interaction with batch processes includes:
   (A) preparing and/or collecting data prior to batch processing commencing.
   (B) correcting errors after batch processing has completed.
   (C) scheduling when batch jobs should be performed.
   (D) All of the above

3. The isolation ACID property can be relaxed when transactions are:
   (A) processed in parallel.
   (B) processed sequentially.
   (C) performed in real time from multiple sources.
   (D) batch processed.

4. Which of the following is the most significant reason why cheque clearance takes considerably longer than EFTPOS or credit card clearance?
   (A) MICR readers are slow compared to magnetic swipe readers.
   (B) Signatures must be manually verified at the point of sale.
   (C) Ensuring sufficient funds are in the payer's account is performed manually.
   (D) Cheque details are collected from paper documents at different locations.

5. During batch processing, errors detected are commonly written to a file rather than displayed on screen. Which of the following is the best reason why this occurs?
   (A) To permanently record details of all errors encountered.
   (B) It allows batch processes to occur when nobody is present.
   (C) So users are freed to complete real time processes.
   (D) To allow processing to continue without interruption.

6. Recurring household bills are particularly well suited to batch generation because:
   (A) such systems include sequential secondary storage devices.
   (B) the data required to generate the bills already exists within the system.
   (C) large companies have staff dedicated to the bill generation process.
   (D) most households pay such bills using direct deposit or credit cards.

7. Which of the following occurs at check clearance houses operated by the APCS?
   (A) Bundles of cheques are exchanged between banks.
   (B) Cheques are scanned to determine their value.
   (C) The value of each cheque is withdrawn from the payer's account.
   (D) Funds are deposited into each payee account.

8. The four significant parties in all credit card transactions are:
   (A) Customers, retailers, banks and Visa or MasterCard.
   (B) Customers, merchants, clearance houses and banks.
   (C) Customers, merchants, acquirers and issuers.
   (D) Payment gateways, merchants, banks and card companies such as Visa and MasterCard.

9. According to banks, which of the following lists credit card transactions in descending order of risk?
   (A) Internet, MOTO, Card Present
   (B) Card Present, MOTO, Internet
   (C) MOTO, Internet, Card Present
   (D) Card Present, Internet, MOTO

10. Which of the following best describes batch processing?
   (A) Collecting occurs over some time and then many transactions are processed together at a later time.
   (B) Transactions are processed soon after the required data has been collected.
   (C) Many similar transactions are processed in parallel.
   (D) Transactions are added to a queue and are processed in the order in which they were received.

11. Recount the steps that occur once a cheque is deposited until the funds can be withdrawn.
12. Construct a diagram to describe the order of processing occurring to complete a typical "card present" credit card transaction.
13. Sometimes ACID properties can be relaxed during batch processing. Discuss using examples.
14. Compare and contrast the general nature of real time and batch transaction processing.
15. Explain why systems that collect transaction data on paper forms are suited to batch processing.

BACKUP AND RECOVERY

Backup is the process of making a copy of data in case the original is lost or damaged. Recovery is the opposite of the backup process where the backup copy of the data is restored and placed back into the system. We introduced backup and recovery in chapter 2 when discussing techniques for securing data. In this section we shall concentrate on various procedures used to perform backups together with advantages and disadvantages.

> **Backup**
> To copy files to a separate secondary storage device as a precaution in case the first device fails or data is lost.

Backups provide a snapshot copy of data at particular points in time. Each backup copy allows the system's data to be recovered back to the state it was in at the precise time the backup copy was made. In the event of total system failure, such as a hard disk crash or a fire that destroys the data completely, it is important to be able to recover to a point as close as possible to the time the failure occurred.

The most common reason for total system failure is hard disk failure – in particular components that move the read/write heads. It is a fact that all hard disks will eventually fail. Research (refer *Fig 4.28*) indicates that rates of failure are high with new hard disks, largely due to manufacturing faults. Failure rates are significantly lower for disks that are approximately 1 to 6 years of age. Failure rates then rise again as components begin to wear out.

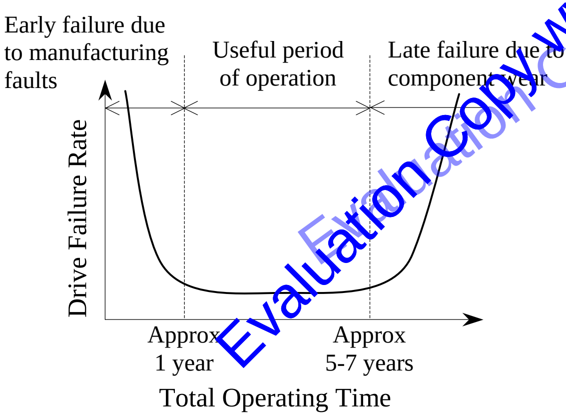


Fig 4.28
Hard disk failure rates over time.

There are many other problems that occur where backup copies made at different times in the past are invaluable. For instance when a user makes changes to a file and later wishes to revert to a previous version. Also viruses are often detected after a period of time has elapsed. In each case having many historical copies of the data allows the system (or a single file) to be restored to a previous state. To recover from the broadest range of possible problems requires backup copies to be made regularly and each backup to be kept for a reasonable amount of time.

> **GROUP TASK Discussion**
> Brainstorm lists of possible failures where the most recent backup is most useful and another list where older backups are more useful.

Just how often backups are made and for how long they are kept is dependent on the value and nature of the data. The value of data includes the costs associated with recreating the data, together with the cost of the system being inoperable. Currently recreating 10MB of data is estimated to cost on average about $50,000. Furthermore it is estimated that some 43% of businesses that experience a severe or total loss of data never reopen. Clearly the importance of performing regular backups and ensuring they can be reliably restored is critical.

> **GROUP TASK Discussion**
> We have mentioned RAID storage at various times throughout our work. Do mirrored RAID solutions remove the need to make backups? Discuss.

FULL AND PARTIAL BACKUPS

There are three different types of backup that are commonly used within most backup procedures – full backups, incremental backups and differential backups. Both incremental and differential backups involve making partial backups.

Full Backup

As the name suggests, a full backup is a complete copy of all data within the system. This can be a complete image of the entire hard disk(s), including the operating system, program files, configuration settings and of course data. For most transaction processing systems it is the data that is of particular value – the software and configuration settings rarely change and are far easier to restore. Therefore most businesses perform full backups of all their data files on a regular basis.

Full backups are the easiest to restore should failure occur. The full backup is simply copied back into the operational system. Unfortunately copying all files takes a long time and requires large amounts of storage, therefore it is often impractical to perform full backups on a daily basis. Common backup procedures specify that full backups be made on a weekly basis, usually commencing on Friday afternoons and for large systems continuing over the weekend.

Most operating systems store an archive bit along with each file. The archive bit is set to true when changes are made to a file and when a file is first created. When full backups are made all archive bits are set to false indicating that a backup copy of each file has just been made.

Incremental Backup

Incremental backups include making partial backups that copy all files that have changed or been created since the last backup – the last backup may have been a full, or partial backup. An incremental backup therefore includes only those files where the archive bit is true. As a consequence performing an incremental backup is significantly faster and requires significantly less storage compared to a full backup. After each file has been copied its archive bit is set to false. Therefore if incremental backups are performed each afternoon then each incremental backup copies only those files that have been altered or created since the previous afternoon's backup was made.

Before incremental backups can be made it is necessary to first perform a full backup. Commonly faster incremental backups are then made on at least a daily basis. The significant saving in backup time is counteracted by the extra time required to recover the data. During a recover the latest full backup is first restored, then each partial backup is restored in the order in which they were made. Hence files that have changed since the full backup progressively overwrite the older versions as each partial backup is restored.

Differential Backup

A differential backup uses partial backups to makes copies of all files that have been altered or created since the last full backup. If such partial backups are made each day then each will contain copies of all files within all previous partial backups since the last full backup was made. To restore to the most recent backup requires first restoring the full backup and then restoring just the most recent partial backup.

In terms of archive bits, differential backups copy all files where the archive bit is true, however differential backups do not alter any archive bits. Therefore over time one would expect more and more archive bits to be true and hence more and more files are included within subsequent differential backups. The size of the differential backup continues to grow until the next full backup is completed.

TRANSACTION LOGS, MIRRORING AND ROLLBACK

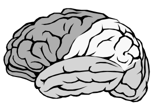
Recall that transaction logs contain historical details of each transaction performed including details of transactions that are currently being processed. These details can be used to restore a transaction processing system back to a consistent state at some precise point in time – completed transactions can be recommitted or rolled back and incomplete transactions can be continued or rolled back. Let us consider some disaster situations where transaction logs, mirroring and rollback assist system recovery.

- Imagine the drive controller on a database server has failed during a busy period – which is when such faults usually occur. At the time of failure numerous transactions were incomplete. To recover from this disaster requires the server to be shut down, a new drive controller installed and the server restarted. The transaction log is then used by the system to automatically rollback all incomplete transactions, which returns the data to its most recent consistent state.

  What about transactions that span multiple servers and systems? In general, most transaction systems automatically abort (rollback) actions that have not been committed after a specified period of time – this deals with most issues. However, most systems include further safeguards to ensure this occurs. If the transaction was initiated by the server that crashed then it sends each system involved in each transaction a message detailing the transaction and the specific actions they should abort. What about actions performed on the crashed server that formed part of a transaction initiated by another system? In this case the server informs the initiating system who then rolls back the complete transaction.

- A database server is attached to two RAID storage devices. The first RAID device stores the main databases and uses RAID striping to improve data access speeds. The second is a mirrored RAID device and is used to store the transaction logs for all databases. Now suppose a disaster occurs which totally destroys the first RAID device and all but one of the hard disks on the second RAID device. Because the second RAID device was mirrored, the remaining hard drive will contain a complete copy of the current transaction log – I'd copy this to a fresh hard drive!

  In this case recovery first requires installing new RAID devices, installing software and then restoring the data from the most recent set of backup media. We now have consistent data but it is missing all changes made since the last backup. The solution to this problem is to use the transaction log to roll forward and recommit all transactions performed since the last backup.



Consider the following

Backup and recovery protects against each of the following:

- Hardware errors and failure.
- Software errors.
- Physical theft or destruction of hardware.
- Unauthorised or unwanted changes to data, due to viruses or hacking, for example.
- Intentional changes to data that for various reasons need to be reversed.

GROUP TASK Discussion

For each of the above dot points, discuss suitable backup techniques that will protect the data. Consider the use of secure onsite and offsite storage, full and partial backups, mirroring and the use of transaction logs.

BACKUP MEDIA

Magnetic tape remains the dominant media for backing up data on large systems, including most transaction processing systems. Other forms of backup media include hard disks, CDs and DVDs. Compared to magnetic tape, the limited capacity, lower data transfer speed and higher cost of these alternatives makes them unviable alternatives for backup of most large systems. Currently online businesses are emerging where backups can be made over the Internet. Some large organisations maintain their own dedicated high-speed communication links to remote backup sites.

Magnetic Tape

Magnetic tape is a sequential access media contained within cassettes or cartridges and is currently the most convenient and cost effective media for backup of large quantities of data. Magnetic storage, including tape was described in some detail back in chapter 2, therefore we restrict our discussion to their widespread use for backup purposes.

Fig 4.29
Various types of magnetic tape cartridges.

A single inexpensive magnetic tape can store the complete contents of virtually any hard disk; currently magnetic tapes (and tape drives) are available that can store in excess of 500GB of data at just a few cents per gigabyte. Most backup systems compress data prior to it being written to tape, this compression usually doubles the capacity of most tapes – a 500GB tape can actually be used to backup 1TB of system data.

Tape cartridges encase a much larger surface area of storage material than other forms of removable storage. The ability to backup such large amounts of data using just one tape far outweighs the disadvantages of sequential access. In any case both backup and restore procedures are essentially sequential processes. Furthermore tape cartridges are light, portable and do not contain complex electronics. This makes the cartridges suitable for long term and offsite storage.

There are two different technologies currently used to store data on magnetic tape, helical and linear. In the related Preliminary textbook we discussed the detailed operation of helical and linear tape drives.

Tape libraries, such as the one shown in *Fig 4.30*, include multiple tapes and multiple tape drives. A robotic system moves tapes between the storage racks and the tape drives. Such systems allow tapes to be automatically rotated according to the details of the organisation's backup procedures. The tape drives are just normal single drives whose operation has been automated. The use of many standard tape drives improves the fault tolerance of the tape library as complete drives can be replaced without affecting or even halting backup processes.

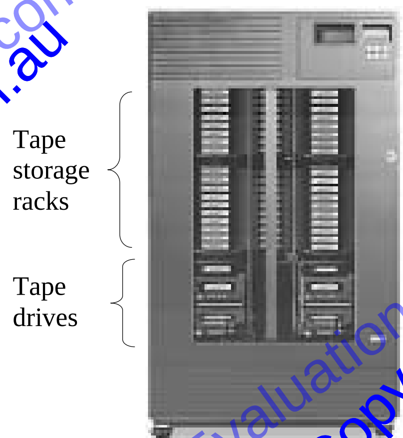


Fig 4.30
Qualstar's TLS-58132 tape library stores up to 340 terabytes of data.

Various different size tape library devices are available to suit the backup demands of different information systems. Small tape libraries are available that hold just four tapes and use a single drive; these devices provide capacities suited to most small businesses. Larger devices hold hundreds or even thousands of tapes and contain many drives. Large government departments and organisations link multiple tape library devices together; such systems hold hundreds of thousands of tapes and many thousands of tape drives.

Hard disks

The use of hard disks for backup has recently become popular for smaller systems. External hard disk devices are available that connect to a computer via high-speed USB or firewire ports, whilst others connect directly to Ethernet networks. In terms of cost these alternatives are still significantly more expensive than tape if the equivalent level of protection is to be achieved – currently tapes costs tens of dollars each whilst similar capacity hard disks cost hundreds of dollars each. For backup processes many hard disks are required. Nevertheless for small business and home backup purposes external hard disks are now a viable alternative. For larger systems the physical size, weight and mechanical complexity of hard disks is significant when the media must be transported to secure offsite storage.

Note that mirrored RAID systems use multiple hard disks to store copies of data. These systems protect data and provide fault tolerance should one of the mirrored drives fail. Such systems do not protect data against total system failure and are of no use when historical data is required to rebuild the system to a prior state. Hard disks used for backup are configured to perform full backups and partial backups such that the system (or individual files) can be restored to previous states.

GROUP TASK Research
Research the current cost of external hard disks with a similar capacity to the hard disks within current personal computers.

Optical Media – CD and DVD

For single machines and small businesses optical media is a popular and low cost backup solution – most computers include CD/DVD read/write drives and rewritable DVD media is relatively inexpensive. A single layer DVD stores 4.7 gigabytes and a double layer DVD approximately 8.5 gigabytes of data, this is sufficient for backing up most hard disks. Even when data compression is used multiple rewritable DVDs will likely be needed for a full backup, however a single DVD is generally sufficient for partial backups.

For backups of larger systems DVD media does not have sufficient capacity to make it a viable alternative to tape. Currently a single magnetic tape cartridge can store more than 500GB of data, this amount of data requires in excess of 50 DVDs. Large government and commercial organisations use tape libraries that backup to hundreds or even thousands of tape cartridges. Clearly using DVDs would require 50 times more DVD disks than tapes. This is unviable in terms of quantity of disks but also in terms of physically moving the disks in and out of drives.

Recordable "write once-read many" (WORM) CDs and in particular DVDs are used for archiving critical data that must be archived in an unalterable form for long periods of time – often exceeding 10 years. In most instances the data must be archived permanently because of governmental or organisational requirements. In most cases this data will rarely be read. Such applications include medical, legal and taxation records that were traditionally stored on paper. For larger systems optical "jukebox" devices are available. These devices include multiple optical drives together with automatic disk changers. The Pioneer DRM-3000 shown in *Fig 4.31* includes two DVD drives and 6 magazines that each holds 50 DVD-R disks.

Fig 4.31
Pioneer DRM-3000 optical disk jukebox.

GROUP TASK Research
Research and then compare and contrast the current capacity and cost of tape cartridges compared to optical disks.

Online Systems

Businesses are beginning to emerge on the Internet that specialise in providing online backup and recovery for individuals and small businesses. These online systems totally automate the backup process for users. All data is transferred via the Internet to a secure remote site. The remote site then manages the secure storage of the data on behalf of the individual or business. Clearly the remote site must use some form of secure and permanent storage. When first using an online backup system a full backup must be made, which is a time consuming process. After the initial backup, incremental backups are made at regular intervals – in some cases every time a file is saved. Such systems enable recovery of different historical versions of individual files as well as recovery of complete systems.

Large organisations that manage large volumes of critical data maintain complete operational copies of their entire system at remote locations. Such copies include the hardware, software, communication lines and data. Data from the original site is continually backed up via online communication lines to the remote site or sites. This is the ultimate in fault tolerance as a complete system failure, such as a fire or terrorist attack, can be recovered instantly by simply activating the backup site.

GROUP TASK Research
Research using the Internet current online backup services. Determine the capabilities and cost of such services.

BACKUP PROCEDURES

The same backup media should not be used continuously to perform backups. Rather multiple sets of backup media should be purchased and used. The aim is to maintain many complete backup copies produced at different times such that the system's data can be recovered back to a variety of different past states. If only a single set of backup media is used then failure of the media can spell disaster. Furthermore many problems, such as viruses, may go undetected for some time. In these cases a backup copy produced prior to the problem occurring is invaluable.

A definite backup procedure is required that is documented and applied consistently. Most backup procedures fail as a result of human error. Therefore it is vital that backup procedures are thoroughly understood and are simple to apply. It is particularly important for the people who perform the backups to be aware of their importance – backups can easily become a chore that are easily overlooked. The procedure should specify which set of media is to be used for each backup and when and where backup copies should be stored offsite.

Backup procedures should also specify how backup copies are to be verified to ensure they will actually work in the event of failure. Commonly the backup software verifies all data on the media as the backup is being made – essentially after writing, the data is read back into RAM and compared to the original. Specialised backup software is available that can be configured to enforce the backup procedure – including verification. However human assistance is still needed to physically change the backup media and to ensure media is stored offsite as required. It is advisable to manually perform a test recovery at regular intervals to ensure recovery operates as expected. Such recovery tests should be performed using a different media drive – it is possible that tapes or other media will not operate correctly in different drives. All backup copies will be useless if the backup drive itself fails or is destroyed.

GROUP TASK Research
Research backup software included with your operating system and other examples of specialised backup software. Outline the available features.

A simple, albeit costly, backup strategy would be to make a full backup to new media at regular intervals such as every afternoon. Such a system is certainly simple to implement and for some critical or high value data such a strategy may well be an appropriate solution. However for most systems a less costly solution that reuses the backup media is generally preferred. There are three commonly used media rotation schemes; Grandfather, Father Son (GFS), Round Robin and Towers of Hanoi. We shall discuss examples of each of these schemes. To simplify our discussion we assume a single tape is sufficient for completing each backup. In reality each backup may require multiple tapes, DVDs or some other type and quantity of backup media.

Grandfather, Father, Son (GFS)

This is the most commonly used rotation scheme. GFS rotation requires daily or son tapes, weekly or father tapes and monthly or grandfather tapes. Full or partial backups are performed each working day to a son tape, except for the last workday. On the last workday a full backup must be performed to one of the weekly or father tapes. At the end of the fourth week a full backup is made to one of the monthly or grandfather tapes. The set of son tapes is reused each week, the set of father tapes is reused each month and the set of grandfather tapes is reused each year. Usually the monthly or grandfather tapes are stored offsite and the weekly tapes are stored onsite within a safe, however this is varied to suit the needs of the individual organisation.

To implement a GFS rotation within an organisation that operates 5 days per week requires four son tapes, three father tapes and thirteen grandfather tapes. Note there are 13 four-week periods in a year, not 12. The son tapes are labelled Mon, Tues, Wed and Thurs. The father tapes are labelled Week 1, Week 2, Week 3 and the grandfather tapes Month 1, Month 2, … Month 13. After making an initial full backup the schedule in *Fig 4.32* is used to determine the tape media used for each afternoon's backup.

Mon	*Tues*	*Wed*	*Thurs*	*Fri*
Mon	Tues	Wed	Thurs	Week1
Mon	Tues	Wed	Thurs	Week2
Mon	Tues	Wed	Thurs	Week3
Mon	Tues	Wed	Thurs	Month1
Mon	Tues	Wed	Thurs	Week1
Mon	Tues	Wed	Thurs	Week2
Mon	Tues	Wed	Thurs	Week3
Mon	Tues	Wed	Thurs	Month2
Mon	Tues	Wed	Thurs	Week1
Mon	Tues	Wed	Thurs	Week2
Mon	Tues	Wed	Thurs	Week3
Mon	Tues	Wed	Thurs	Month3

Fig 4.32
Grandfather, Father, Son media rotation.

Weekly and monthly backups should always be full backups, however the daily or son backups can be full or partial backups. If a relatively small amount of data is present then full backups can be used throughout. When full backups are used just one tape is required to restore data from the most recent backup, or indeed from any backup. If differential daily backups are made then two tapes are required to restore to the most recent backup – the last weekly full backup is first restored followed by the most recent daily differential backup. If incremental daily backups are used then the most recent weekly full backup is restored followed by restoration of each of the subsequent incremental daily backups. Using full daily backups simplifies the restore process at the expense of longer backups. Using differential daily backups results in slightly more complex restore processes, but reduces the time taken for daily backups significantly – only files changed since the last full backup are copied. Using incremental daily backups

Pages 421 to 440

Not included in this Evaluation Copy

ISSUES RELATED TO TRANSACTION PROCESSING SYSTEMS

There are numerous significant issues that should be considered when designing and operating transaction processing systems. In this section we restrict our discussion to issues in regard to:

- The changing nature of work.
- The need for alternative non-computer procedures.
- Bias in data collection.
- Data security, data integrity and data quality issues.
- Control and its implications for participants.

THE CHANGING NATURE OF WORK

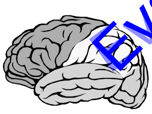
The nature of work has seen significant change since the 1960s. These changes have been both in terms of the types of jobs available and also in the way work is undertaken. The widespread implementation of computer-based systems, and in particular transaction processing systems, has been the driving force behind most of these changes. In the early 1970s many thought that the consequence of new technologies would be a reduction in the total amount of work needing to be done; this has not occurred, rather new industries and new types of employment have been created. Many people are now working longer hours, in more highly skilled, and stressful jobs than ever before.

Industries that once employed significant numbers of clerks have seen the greatest changes. The majority of tasks traditionally performed by clerks are now automated. Consider banks, transaction processing systems have largely replaced the numerous clerks that once worked within each branch. Furthermore the widespread use of ATMs, EFTPOS and credit cards mean customers rarely need to visit the bank. The data entry tasks performed by bank staff are now performed by the customer, in the case of ATMs and by retailers in the case of EFTPOS and credit card transactions. In recent years the Internet has changed how transaction data is collected and processed. It is now common to complete totally automated online purchases. No human employed by the retailer needs to have any direct interaction with customers during the transaction's processing.

THE NEED FOR ALTERNATIVE NON-COMPUTER PROCEDURES

What happens when a transaction processing system fails? Perhaps there is a power failure, lightning strike, fire, theft or communication lines are broken. Maybe the data within the system has been lost or some hardware components are inoperable. Recovery then involves purchasing replacement hardware, rebuilding systems and restoring data. This takes time and during this time an alternative mode of operation is required. For large centralised systems such problems are resolved by maintaining backup power generators and redundant communication lines at complete mirrored sites. For smaller systems, alternative non-computer procedures are needed if the organisation is to continue to operate. Commonly the only alternative is a return to paper based non-computer procedures whilst the problems are corrected.

Alternative non-computer procedures should be trailed and tested at regular intervals to ensure they operate as planned. In particular such tests should ensure all participants understand and are able to correctly implement the procedures. For example when banks supply retailers with EFTPOS terminals they commonly supply stock of manual paper forms. These paper forms allow the retailer to continue trading despite failure of the EFTPOS system. However sales assistants must know how to process sales using these paper forms – this requires training and regular testing.



Consider the following examples of system failure:

- A local post office is broken into and all computers are stolen. Upon phoning Australia post it is determined that it will be one week before replacements arrive.
- A thunderstorm disrupts the communication lines into a large warehouse. The warehouse is informed that the lines are unlikely to be restored for 3 days. The transaction processing systems at the warehouse receives and processes hundreds of orders per day that are subsequently shipped out by a fleet of 20 trucks.
- The ATMs outside a busy bank branch are ram raided and the cash boxes are stolen. It will take at least two weeks for replacement ATMs to be installed.

GROUP TASK Discussion
Propose possible non-computer procedures that could be used to minimise the effects of each of the above system failures.

GROUP TASK Discussion
Explain possible techniques that could be used to train participants and test the procedures proposed above.

BIAS IN DATA COLLECTION

Bias is an inclination or preference that influences most aspects of the collection process; the result of bias during collection is inaccurate data leading to inaccurate outputs from the system. Those involved in collecting data must aim to minimize the amount of bias present.

Bias
An inclination or preference towards an outcome. Bias unfairly influences the outcome.

When deciding on the data to collect bias can be introduced. Often incomplete data is collected with the aim of simplifying the system. For example it is common for loan applications to collect data on a person’s income based entirely on their last few tax returns. This data is used to assess each individual’s ability to repay the loan; the assumption being that an individual’s income is likely to remain relatively constant over time. In fact many people, particularly those who own or operate businesses, are able to adjust their income to suit their expenses. By simply collecting past income data the success of each loan application is biased in favour of salary and wage earners at the expense of business owners.

Locating or identifying a suitable source of data for collection is another potential area where bias can occur. Often efficiency of data collection means that the cheapest or most available source of data is used rather than the best source of data. Consider surveys; the data source for all surveys should aim to be a representative sample of the entire population. However for ease of collection many organisations collect survey data from users over the Internet. Internet users, in most cases, are not a representative sample of the population; in general Internet users are younger, have higher incomes and possess higher technology skills than the general population. Consequently results derived from such surveys will not accurately reflect the entire population.

The collecting process itself should take into account the likely perceptions held by those on whom the data is collected. People answer questions and fill out forms differently based on their perception of how the data will be used. For example a survey conducted by the Australian Taxation Office is likely to yield different results to a similar survey conducted by the Australian Bureau of statistics. Individuals would likely perceive the tax office as being interested in their individual responses whereas a survey conducted by the Australian Bureau of Statistics is more likely to be viewed as truly anonymous.

DATA SECURITY ISSUES

A summary of strategies we have examined to combat data security issues include:

- Passwords- Passwords are used to confirm that a user is who they say they are. Once verified the user name is then used by the system to assign access rights.
- Backup copies- A copy of files is made on a regular basis.
- Physical barriers- Machines storing important data and information, or performing critical tasks are physically locked away.
- Anti-virus software- All files are scanned to look for possible viruses. The anti-virus software then either removes the virus or quarantines the file.
- Firewalls- A firewall provides protection from outside penetration by hackers. It monitors the transfer of information to and from the network. Most firewalls are used to provide a barrier between a local area network and the Internet.
- Data encryption- Data is encrypted in such a way that it is unreadable by those who do not possess the decryption code.
- Audit trails- The transaction log includes details of who and when transactions were performed. It is possible to work backwards and trace the origin of any problem that may occur.

DATA INTEGRITY ISSUES

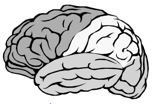
A summary of strategies we have examined to maximise data integrity include:

- Data validation- checks, at the time of data collection, to ensure the data is reasonable and meets certain criteria.
- Data verification- regular checks to ensure the data collected and stored matches and continues to match the source of the data.
- Referential integrity- ensuring all foreign keys in linked tables match a primary key in the related table.
- ACID properties- ensuring transactions are never incomplete (atomicity), the data is never inconsistent (consistency), transactions do not affect each other (isolation) and that the results of a completed transaction are permanent (durability).
- Minimising data redundancy- Normalising reduces or eliminates duplicate data within individual relational databases, however when transactions span multiple databases issues will arise. The use of unique identifiers shared between organisations allows individual entities to be accurately identified.

DATA QUALITY ISSUES

Data integrity is about the accuracy of the data – how well it matches and continues to match its source. Data quality takes this one step further, it concerns how reliable and effective the data is to the organisation. For example, responses on survey forms may well be entered accurately into a system, however the quality of the data will be poor if the respondents didn't answer honestly or as intended. The resulting information will be unreliable and ineffective. Other data quality issues occur when combining data from different systems.

Consider creating a data warehouse from many databases. Some records will describe the same entity differently; both may be correct, so which record is best? The organisation of databases is likely to be different; different keys, data types or schemas, for example. The meaning attached to an attribute can change over time; perhaps a client application was modified and now stores different data in some old field. Combining such data is difficult, unreliable and inefficient. Furthermore the effectiveness and reliability of the information from subsequent data mining and OLAP systems is reduced. Data Quality Assurance (DQA) standardises the definition of data and includes processes that "scrub" or "cleanse" existing data so it meets these data quality standards.



Consider the following data security, integrity and/or quality issues:

- A hacker gains access to an organisation's system. They download customer credit card details and use them to make various purchases over the Internet.
- An RTA employee alters driving test results so that licences are issued to people who failed their driving test.
- A analyst using a data mining application uncovers links between sets of attributes that cannot possibly be true.
- A bank customer determines that a funds transfer has not been completed. The funds have left their account but have not been deposited into the other account.

GROUP TASK Discussion
For each of the above issues, determine the source of the issue and suggest suitable strategies that would assist in preventing the issue re-occurring in the future.

CONTROL AND ITS IMPLICATIONS FOR PARTICIPANTS

Control is the act of influencing or directing activities. In terms of managing the activities of employees some level of control is reasonable. Management assigns tasks and then quite legitimately expects employees to complete these tasks in a timely and accurate fashion. However whenever one has control over another the relationship is open to abuse. Determining precisely when control over participants is excessive is often a grey area. Most would agree it is reasonable for managers to monitor the activities of those they manage, however what level of monitoring is reasonable? Should management control Internet access or be able to read all email messages? Is it reasonable to monitor phone calls or remotely view a users desktop? Audit trails allow management to track which records individuals have accessed; when is such tracking reasonable? Answers to such questions differ considerably according to the management style used and the nature of the tasks participants perform.

Current management theory suggests higher levels of productivity are achieved when participants are motivated. Motivation improves when participants are given responsibility for tasks and how they are completed. Motivated employees are less likely to engage in undesirable activities and are much more likely to focus on work. When employees are assigned boring or repetitive tasks they lose motivation and then quite naturally seek to engage in other non-work related activities. When this occurs management too often imposes authoritative controls such as excessive monitoring in combination with negative consequences in an attempt to enforce control. Such measures further reduce motivation resulting in even stricter controls being imposed – a downward trend emerges. A more sustainable management style encourages trust and motivates employees to take responsibility for work they complete.

GROUP TASK Discussion
As an employee, what level of monitoring by management do you feel comfortable with? Brainstorm scenarios where monitoring and control of participants is necessary (or at least justified).

CHAPTER 4 REVIEW

1. One operation within a transaction fails, what should occur?
   (A) Other operations within the transaction should be committed.
   (B) The system should halt so that the reason for the failure can be corrected.
   (C) All operations within the transaction should be rolled back.
   (D) No further transactions should be performed until the problem is resolved.
2. Participants are those people who:
   (A) are the source of data used by the system.
   (B) receive information output from the system.
   (C) interact directly with the system.
   (D) analyse data within the system.
3. Transaction logs used by most DBMSs include details of records:
   (A) prior to being altered.
   (B) after they have been altered.
   (C) added and deleted.
   (D) All of the above.
4. The file used to store data collected prior to batch processing is commonly called:
   (A) an error file.
   (B) a master file.
   (C) a database.
   (D) a transaction file.
5. Checks to ensure data entered is reasonable are known as:
   (A) data validation checks.
   (B) data verification checks.
   (C) data integrity checks.
   (D) data redundancy checks.
6. Which ACID property ensures either all or no operations within a transaction are committed?
   (A) Atomicity
   (B) Consistency
   (C) Isolation
   (D) Durability
7. Strict sequential processing of transactions ensures which ACID property is observed?
   (A) Atomicity
   (B) Consistency
   (C) Isolation
   (D) Durability
8. What is the main task performed by TPMs?
   (A) Providing an interface between many transaction processing systems.
   (B) Ensuring transactions performed on a database observe the ACID properties.
   (C) Monitoring and ensuring the security of transactions.
   (D) Managing transactions that span multiple database, systems and client applications.
9. At most two sets of backups will be required to completely restore data when which of the following backup types are used?
   (A) Full and incremental.
   (B) Full and differential.
   (C) Incremental and differential.
   (D) Full backups only.
10. High speed MICR readers use which technique to read the MICR line on cheques?
   (A) waveform
   (B) matrix
   (C) CCD
   (D) LED

11. Provide at least TWO examples of systems where each of the following devices is used:
   (a) MICR
   (b) Barcodes
   (c) Magnetic stripes
   (d) RFID readers and tags
   (e) TPMs
   (f) Tape libraries
   (g) Touch screens

12. Compare and contrast each of the following:
    (a) User interfaces for real time processing with user interfaces for batch processing.
    (b) Random (or direct) access with sequential access.
    (c) Data validation with data verification.
    (d) OLAP (Online Analytical Processing) and OLTP (Online Transaction Processing).
    (e) Data warehouses and data marts.
    (f) Data integrity with data quality.

13. (a) Recount the sequence of processes occurring to complete a typical credit card transaction. Assume the transaction is initiated using an EFTPOS terminal supplied by the retailer's bank.
    (b) Describe different uses of transaction logs within transaction processing systems.
    (c) Distinguish between the storage of collected data and the storage of processed data in a batch transaction processing system using an example.

14. A company's mail server records each email sent or received in a separate file. Incremental backups using a round robin rotation occur automatically every hour to an online tape library. All employees have full access to files within the tape library. Full backups are not made, however all archive bits were set to true when the system was first installed. Tapes are changed every year as there is sufficient capacity to store messages for 12 months.
    (a) Critically evaluate the above backup procedure.
    (b) Predict issues that may occur as a consequence of the above backup procedure.
    (c) Propose and justify an improved procedure for backup and recovery.

15. Analyse an online web-based purchasing system of your choice.
    (a) Determine the data items collected,
    (b) Identify the operations performed during a purchasing transaction, and
    (c) Evaluate the design of the data collection web forms.
    (d) Explain how the company could analyse the collected data to identify areas for improving its operations.