

Online Transaction Processing

Indra Tobing

An Overview

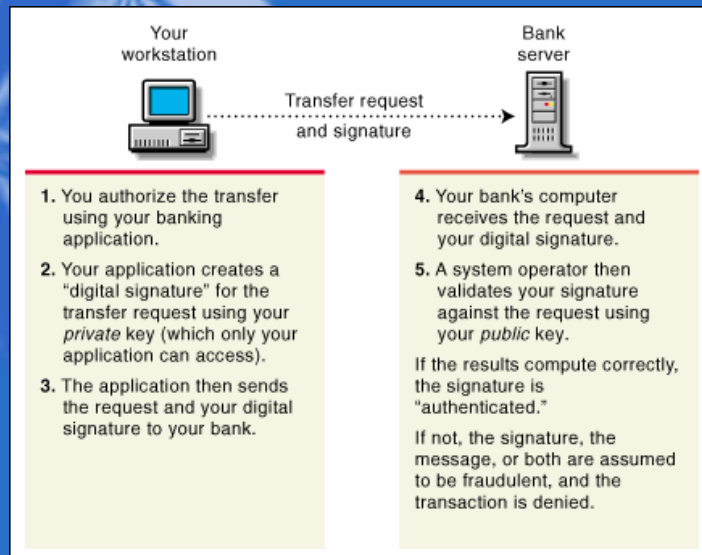
An application is a particular use to which a data processing system is put, for example:

- a payroll application
- an order entry application.

Commercial applications typically process many similar items, for example;

- an order in an order processing system,
- a seat reservation in an airline booking system,
- or a credit query in a credit control system.

An Overview



An Overview

Online Transaction Processing (OLTP) applications are client/server applications that give online users direct access to information.

The OLTP applications process units of work, called transactions. A single transaction might *request a bank balance*; another might *update that balance to reflect a deposit*.

5

An Overview

In a transaction processing system, one execution of an application program processes a single transaction.

End users have online access to the system and to enterprise data, and directly initiate transactions.

In a transaction processing environment, many users repeatedly process similar transactions, and require a fast response to each transaction.

Examples of such users are:

order entry clerks,
airline reservation clerks,
or bank tellers.

They share an environment of programs and data.

6

An Overview

In a typical transaction processing system:

Many end users run the same or similar transactions, sharing the same databases and files

The system can schedule transactions on the basis of priority attributes

The transactions are invoked by online input and generate online output

The transactions are designed for a good user interface and fast response times

7

An Overview

Transaction processing is an effective solution when end users want to:

Process unscheduled single items in unpredictable volumes and sequence

Have immediate access to enterprise data that has been updated to reflect all previous transactions

Change enterprise data immediately to reflect each transaction as it is processed.

8

An Overview

Predefinition of transactions and data allows transaction processing to be controlled efficiently.

The invocation of a transaction causes the invocation of one or more application programs that are written especially to minimize computer time and duration of locks.

A lock prevents transactions from accessing data while it is being updated by another transaction.

9

Classification

Example of transaction processing

The example in figure 1 is an order entry system. Multiple users can invoke the order entry transaction concurrently. Each transaction invocation uses the same program (in a reentrant way) to process a single order. A reentrant program can be entered before completion of a prior execution, and yet execute correctly. The design of the transaction and the transaction processing system aim to separate user "think-time" from the use of storage and other resources.

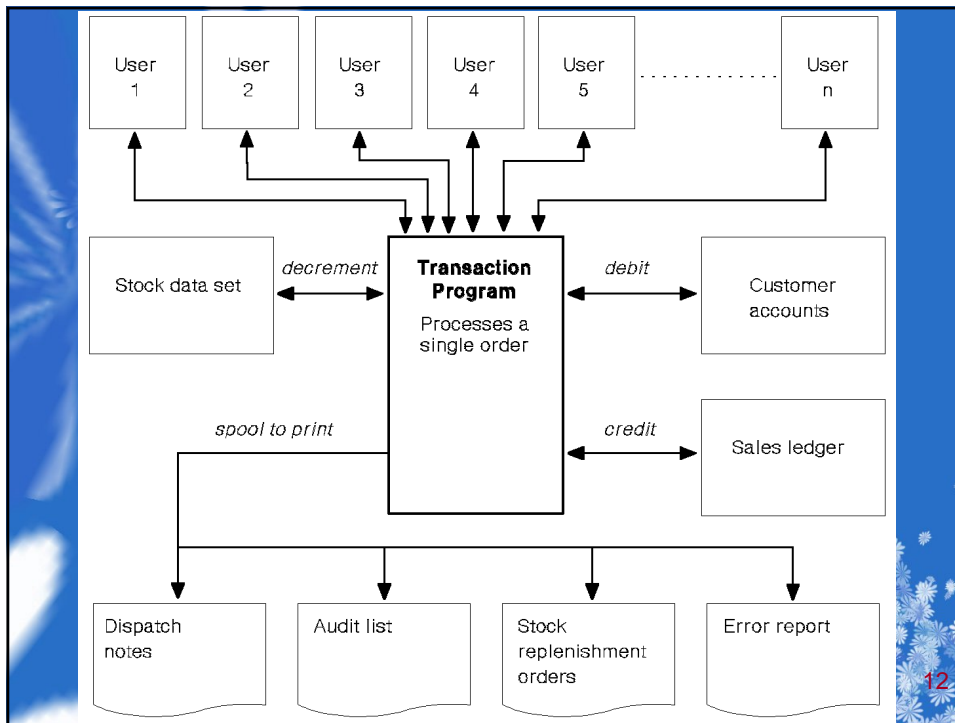
10

Classification

Example of transaction processing (cont'ed)

On receipt of an order, the end user immediately runs a transaction to process it. The transaction runs as close as possible to the receipt of the order, and sees the up-to-date state of the data as affected by all orders previously processed. Therefore, the transaction can confirm to the user whether the order is accepted, rejected, or held over. Orders can be accepted and delivery confirmed immediately.

11



Classification

Batch processing

In a batch processing system, an application program processes a batch of items or records (the processing of each record corresponding to a transaction). The batch of records has been collected over a period, and is usually presorted for sequential processing at a predetermined time. Compare this with transaction processing, in which an application program processes a single item on demand, for example, the receipt of an order.

A major difference between transaction processing and batch processing is that the transaction processing end user has online access when the application program is running, while batch processing is usually completely independent of end users.

13

Classification

Batch processing (cont'ed.)

Batch processing lacks the immediacy of transaction processing, but is an economic way to computerize clerical work, and is ideally suited to applications like payroll, where immediacy is not important.

Batch processing makes the most efficient use of computer resources, especially processing power. A batch system provides efficient processing of applications that have the following characteristics:

No online access

Minimal sharing requirements

A throughput objective of maximum data processed in minimum elapsed time.

14

Classification

Batch processing (cont'ed.)

Batch system unit scheduling is a job, which consists of one or more programs that are executed serially. A job usually corresponds to an application, but a complex application can consist of several jobs. Depending on data volumes and processor power, a job can take up to several hours processing time, during which most of the required resources are dedicated to the job.

A batch system uses processing power economically because it receives input data in a presorted batch that enables sequential processing of the data store. The batching of data to produce a continuous stream means that a batch system tends to monopolize use of the data store.

15

Classification

Batch processing (cont'ed.)

A batch system fails to meet requirements when end users want to process single items in unpredictable volumes and sequence—for example, sales staff querying a customer's credit limit before making a sale.

Transaction processing is then the solution.

Examples of batch applications include the printing of monthly credit card statements or the closing of the month end general ledgers. The following example is an order entry system, which is chosen for comparison with the previous example of a transaction processing solution for the same application.

16

Classification

Example of batch processing

The batch program (figure 2) processes the order data set sequentially, updating the customer accounts data set, sales ledger, and stock data set, and raising dispatch notes for each order. Orders can be rejected due to the state of a customer's account, or held over due to lack of stock.

During the day, sales staff prepare the data for each order and place it in a batch of data for later (in this example, probably overnight) processing. Compare this with the transaction processing solution, where, instead of preparing batch data records, the end users process each order as it is received and can immediately confirm acceptance or rejection.

17

Classification

Interactive processing

An interactive system provides online access to applications and data, which may be the user's private data or enterprise data. In this respect, interactive processing is similar to transaction processing, and the two are sometimes confused.

The difference lies in the kind of interaction. In transaction processing, user interaction usually consists of short, but repeated, requests for processing and resources, and the system and the applications combine to provide rapid response (ideally < 1 sec.) for many users simultaneously. There is no time sharing overhead. In interactive processing, the user has lengthy interactions with the system, with applications running for long periods, and the system enforces time sharing to ensure a reasonable response to each user.

18

Classification

Interactive processing (cont'ed)

Unlike a transaction processing system which aims to separate user "think-time" from the use of storage and other resources, an interactive system retains storage while the user decides what to do. Interactive updating, to be secure, requires a lengthy period during which data being updated by the interactive user is not accessible by other users. This temporary restricted access imposed by the data store manager is called a lock. The lengthy locks required by interactive updating are not acceptable when many users continually access the same database. For these reasons, interactive processing is less suitable for the order entry system used in the transaction and batch processing examples, or for any system involving repetitive processing of large volumes of data.

19

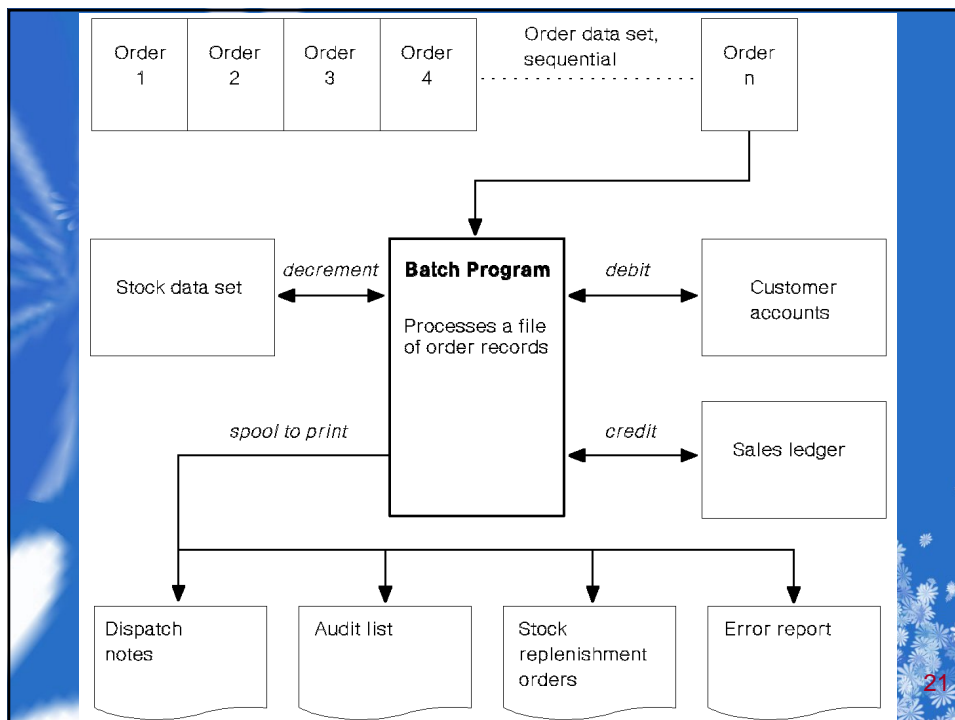
Classification

Interactive processing (cont'ed)

Interactive systems are designed to reserve sufficient computing power for each user to run computeintensive tasks. Long-term connections between the user and the system are common. The user's runtime environment is isolated from those of the other users on the machine.

Personal computing on a stand-alone programmable workstation is normally interactive processing. Unix, CMS (running under the VM operating system), and TSO (running under OS/390) provide interactive processing on a mainframe processor. Interactive applications can be user-written and indeed end-user-written, but the majority of applications running today use bought-in packages.

20



21

Classification

Interactive processing (cont'ed)

Interactive processing is characterized by:

- **Relatively long-running programs**
- **Relatively long ownership of resources and data**
- **Relatively long sessions using a single application.**

22

Classification

Interactive processing (cont'ed)

Typical interactive applications include:

- **Program development**
- **Project management**
- **Decision support (using query languages, statistical packages, and so on)**
- **Editing**
- **Graphics, including engineering design.**

23

Interactive processing (cont'ed)

Usually, this kind of processing is done on copies or extracts of the data used by a batch or transaction processing system. Read-only interactive processing is feasible for data while it is being updated by a transaction processing system. Interactive updating can disrupt a transaction processing system, because, while a record is being updated by a transaction, it is locked and access is denied to other transactions. Unlike transaction processing, interactive systems and applications are not designed to minimize the duration of locks.

24

Basic transaction processing characteristics

The basic functions that must be supplied by a transaction processing system are:

- Online processing
- High availability
- Rapid response
- Low cost per transaction
- Access and update of shared resources with integrity.

25

Basic transaction processing characteristics

Online processing

To be online, the end user needs a terminal or workstation connected to the system. The physical connection can be local, over a local area network (LAN), or a wide area network (WAN). The logical connection is provided by the transaction processing system, either alone or in conjunction with a communications access method and network control programs.

26

Basic transaction processing characteristics

High availability

A transaction processing system must be available for processing when enterprise staff need it. Today more and more enterprises require better than 99.99% availability, approaching a 24-hour day, 7-day week schedule. This must be achieved by a combination of hardware, software, application, and automated facilities to recover from any component failure. In addition to handling failures, the system should minimize the impact on the users of scheduled outages, such as hardware upgrades, software changes, batch work, and database backups and reorganizations.

27

Basic transaction processing characteristics

Rapid response

Transaction processing system is used by many people repeatedly as part of their working routine, so the response time must be short. Subsecond response time down to .3 sec. significantly improves the performance and effectiveness of people working at a terminal.

A transaction processing system can provide a facility to achieve prioritization of transactions, so that selected transactions are automatically scheduled ahead of others. This should be seen as a faster response to the selected transactions, and a slower, but acceptable, response to others.

Load balancing across a transaction processing system is required to cope with peak demands while maintaining response times. To achieve this, the system needs efficient scheduling mechanisms.

28

Basic transaction processing characteristics

Low cost per transaction

In many installations, the transaction processing system is used for the enterprise's core business. The cost of processing a transaction is part of the cost of doing business, and must be an acceptably small fraction of the value of the business transacted. The cost of a transaction is an apportionment of the total capital cost of the system required to perform the transaction (as well as running costs).

Each transaction must be scheduled, initiated, processed, and terminated efficiently. Each of these phases must be optimized to provide efficient transaction throughput. Ideally, a transaction processing system should exploit the strengths of the hardware and operating system, and of other software that can run in the same environment, such as data store managers and external security managers.

29

Configuration

The basic configuration of a transaction processing system are:

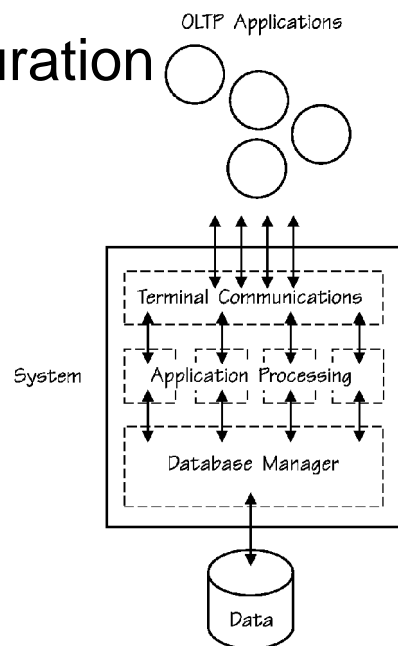
A component that interconnects to terminals or workstations

A data base manager

A number of application programs.

30

Configuration



1

OLTP Design Consideration

Transaction processing system databases should be designed to promote:

- Good data placement.
- I/O bottlenecks are a big concern for OLTP systems due to the number of users modifying data all over the database. Determine the likely access patterns of the data and place frequently accessed data together. Use filegroups and RAID (redundant array of independent disks) systems to assist in this.
- Short transactions to minimize long-term locks and improve concurrency.

32

OLTP Design Consideration

Transaction processing system databases should be designed to promote: (cont'ed)

- Avoid user interaction during transactions. Whenever possible, execute a single stored procedure to process the entire transaction. The order in which you reference tables within your transactions can affect concurrency. Place references to frequently accessed tables at the end of the transaction to minimize the duration that locks are held.
- Discrete Transaction
- Online backup.

33

OLTP Design Consideration

Transaction processing system databases should be designed to promote: (cont'ed)

- OLTP systems are often characterized by continuous operations (24 hours a day, 7 days a week) for which downtime is kept to an absolute minimum. Although Microsoft® SQL Server™ 2000 can back up a database while it is being used, schedule the backup process to occur during times of low activity to minimize effects on users.
- High normalization of the database.

34

OLTP Design Consideration

Transaction processing system databases should be designed to promote: (cont'ed)

- Reduce redundant information as much as possible to increase the speed of updates and hence improve concurrency. Reducing data also improves the speed of backups because less data needs to be backed up.
- Data that is rarely referenced can be archived into separate databases, or moved out of the heavily updated tables into tables containing only historical data. This keeps tables as small as possible, improving backup times and query performance.

35

OLTP Design Consideration

Transaction processing system databases should be designed to promote: (cont'ed)

- Little or no historical or aggregated data.
- Careful use of indexes.
- Indexes must be updated each time a row is added or modified. To avoid over-indexing heavily updated tables, keep indexes narrow. Use the Index Tuning Wizard to design your indexes.
- Optimum hardware configuration to handle the large numbers of concurrent users and quick response times required by an OLTP system.

36

OLTP Design Consideration

Transaction processing system databases should be designed to promote: (cont'ed)

- Transaction Process Monitor
- Integrity Constrain

37