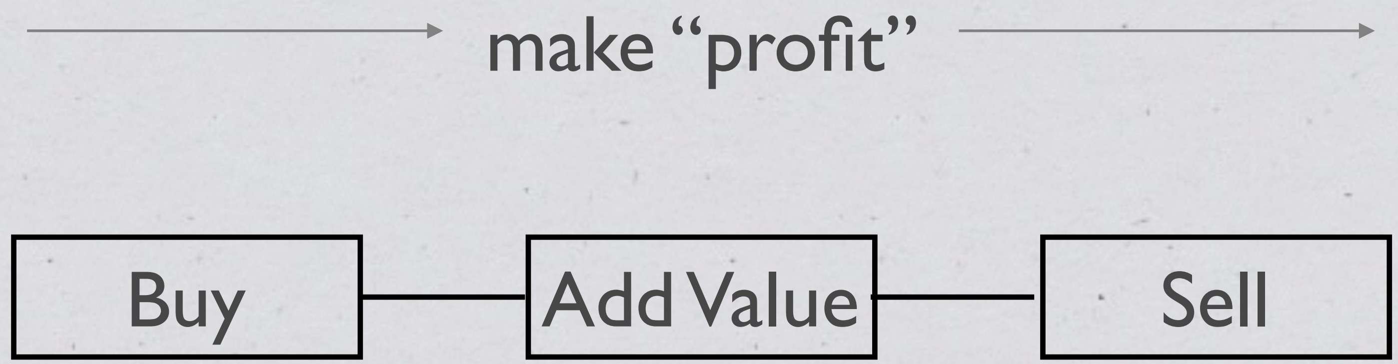
THE EVOLUTION OF ENTERPRISE SYSTEMS 1965 - 2005

Dr GEOFF SHARMAN FBCS

What do Enterprises Do?

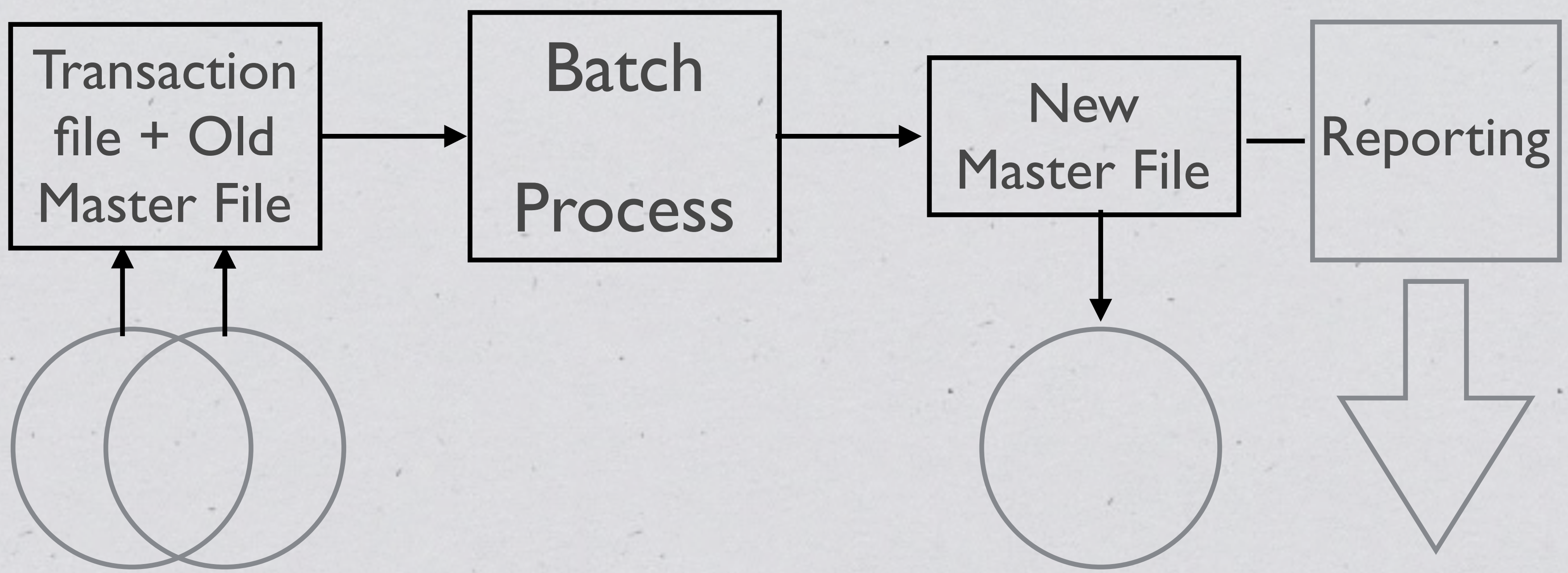


* We'll use a very simple model of what a business enterprise does, which covers manufacturing, service, utility, distribution, retail and many other types of industry
* Remember, businesses have only three ways of collecting money: transaction fees, subscription fees, and advertising revenue.

Definition

- For the purposes of this discussion, an **Enterprise System** is a production hardware/software system which is **essential to the core operations of a business**, e.g.:
    - managing **sales**, distribution, billing and customer information
    - managing “**manufacturing**”, inventory, forecasting and cost accounting
    - managing **buying**, subcontracting and the supply chain
- But **excluding** office systems, scientific systems, etc.

Enterprise Systems before 1965



- Business applications used **batch processing** exclusively (sequential files, updates within business accounting period), e.g.
    - Joe Lyons & Co. **Leo System** for batch accounting & payroll operations (1952)
- IBM announced **System/360** for business and scientific applications (1964)

1965: A year of change

- IBM had also just completed the **SABRE airline reservation system** for American Airlines, based on experience from **SAGE** (US Government Air Defense System)
- Exploiting the new technologies of data networking, “random access” disk files, and display screens
- With a new emphasis on “real time” operation
- This quickly became the model for **OnLine Transaction Processing** systems in large enterprises across many industries

1965: A year of promise

* **Gordon Moore** of Intel formulated “Moore’s Law”
* The number of transistors/unit area of silicon would **double every two years** for the foreseeable future
* This had the effect of doubling computing power at **constant cost**, i.e. an **exponential** increase in compute/$

Fast forward to 2005

- Moore's law continues, but **no longer at constant cost**: signal/noise ratio prevents further voltage reductions and increases in clock speed; silicon designers move to dedicated processors and parallel SoC (Systems on Chip) e.g. 20 processors in iPhone 4
- Moore's Law has **changed everything**: cost/instruction, market size, device form factors and network bandwidth, opening up the era of "pervasive computing"
- And the **computer industry has changed radically** with few companies surviving from 1965

A 2005 Conundrum

- **OLTP survives and thrives**: by this date, it had become the **dominant mode of use** on servers, e.g.
  - Online financial transactions, e.g. Internet banking, insurance, travel reservations
  - Internet browsing
  - Online shopping, e.g. Amazon, eBay
  - Social media, e.g. Facebook, Twitter
- **How did this happen**, when so much else changed?

Did you do any of these today?

- Buy something in a supermarket?
- Use a cash machine, debit card or contactless card?
- Pay for something with a credit card?
- Make a telephone call?
- Travel by public transport?
- Watch catch up TV?
- Use electricity, gas or water?

Systems using one OLTP monitor handle > 100 billion (10^{11}) transactions (financial value > \$$10^{13}$) per day, in the US alone

Google processes 3.5 billion (3.5×10^{9}) searches per day

WHAT IS OLTP?

What is OLTP?

- Originally called "**OnLine TeleProcessing**", it denoted the use of terminals connected to a central computer via telephone lines
- The terminals were used by employees of airlines, travel companies, utilities, and banks to capture customer transactions at source and process them, rather than filing for later action
- Banks were the first to offer consumer terminals, e.g. Lloyds Bank Cashpoint (IBM 2780) in 1972
- Most networks were private and used a "star" topology

OLTP challenges in 1965

- Typical networks were small (~50 terminals), but a key problem was **handling concurrent activities** efficiently
  - network lines had **low bandwidth** (~1024 bits/sec) and could not be shared by different applications
  - processor hardware was **slow** (~1 MIPS)
  - accessing data on tape was **slow** (seconds/record)
  - software process scheduling was **slow**, had limited scalability, and applications were hard to write - mostly in assembler language

Hardware to the Rescue?

- Network limitations: higher speed **leased lines** between processing centres, improved modems, better protocols
- Processor speed: **Moore's Law**
- Data access: **"random access"** disks, more bytes/square cm
- **BUT** even the rate of improvement implied by Moore's Law **could not match** market demand and growth

Software challenges

* Existing operating systems, data management systems and programming languages were **designed for batch processing**:
    * **scheduling** an application process required allocation of macro-sized resources and could take **millions of instructions**, i.e. **seconds**
    * most **operating systems** could only handle **a few** concurrent jobs
    * **data management** mainly provided support for **sequential** files
    * **programming languages** did not support **network operations** and other activities required for OLTP

DB/DC systems emerge

* It quickly became clear that new software was needed for:
  * management of **Indexed Files** and **Data Bases**, which allowed direct access to specific records or sets of records within a data file (in milliseconds)
  * support for **Data Communications**, which enabled receiving and sending messages and control of telephone lines
  * rapid scheduling of **short application segments** which could be triggered by a message from a terminal and could create a response message
  * all these functions were required for a usable system, implemented as an **OLTP Monitor**, often for a particular industry or even a specific customer

Real Time Systems

- *"A real time system may be defined as … receiving data, processing them and returning results sufficiently quickly to affect the functioning of the environment …"*
  - James Martin, Programming Real-Time Computer Systems (1965)
- This showed a primary concern with **end user response time**, usually expected to be 2 - 3 seconds
- BUT recall that the **underlying batch accounting process** is real time, too, because business must be completed by end of each accounting period (usually overnight)

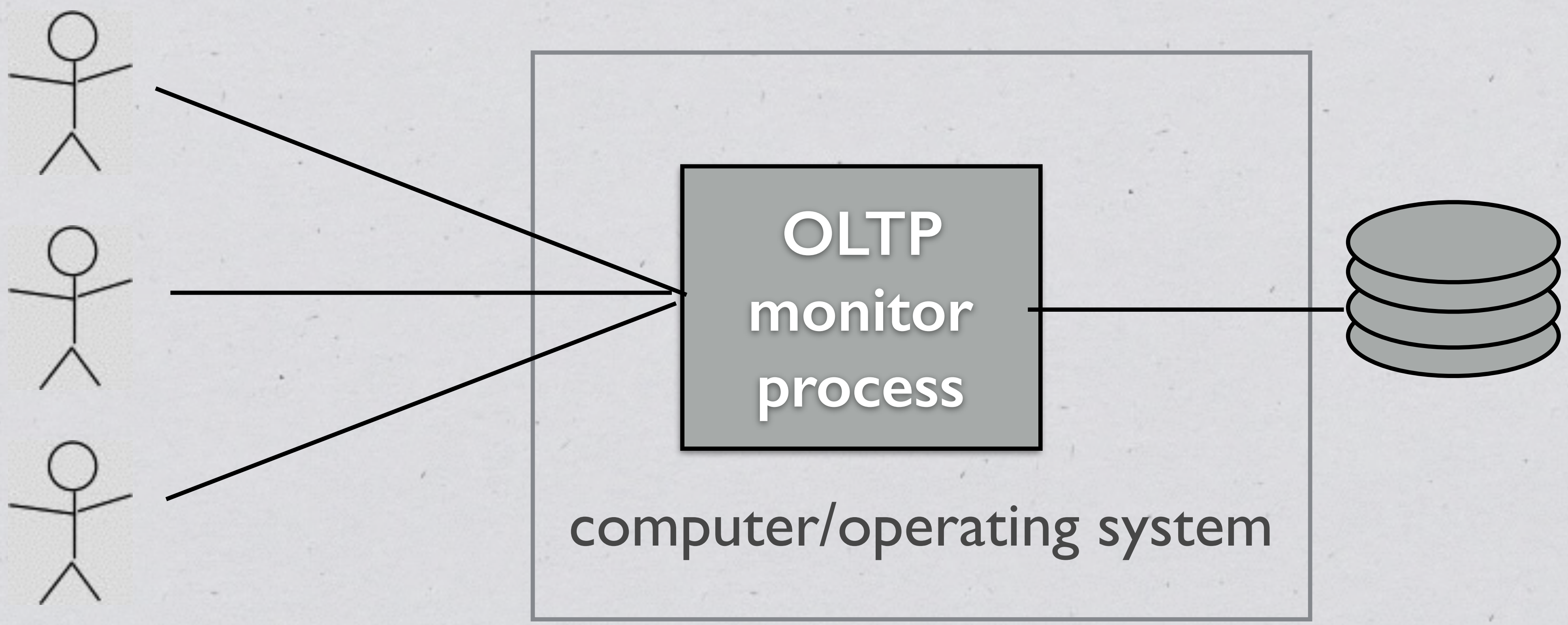
OLTP Monitors emerge

- Early OLTP Monitors based on IBM S/360:
  - **ACP** (Airline Control Program, 1969), for airline reservations
  - **BATS** (Basic Additional Teleprocessing Support) for UK banking
  - **IMS/DC** (Information Management system,1969) with **IMS/DB** for the NASA Apollo space programme
  - **CICS** (Customer Information Control System, 1969) for US utilities
  - **Shadow II** (1976) for UK travel agents
- ICL **TPMS** (1974) with **IDMS** for UK government systems

IBM Hursley Lab: home of CICS

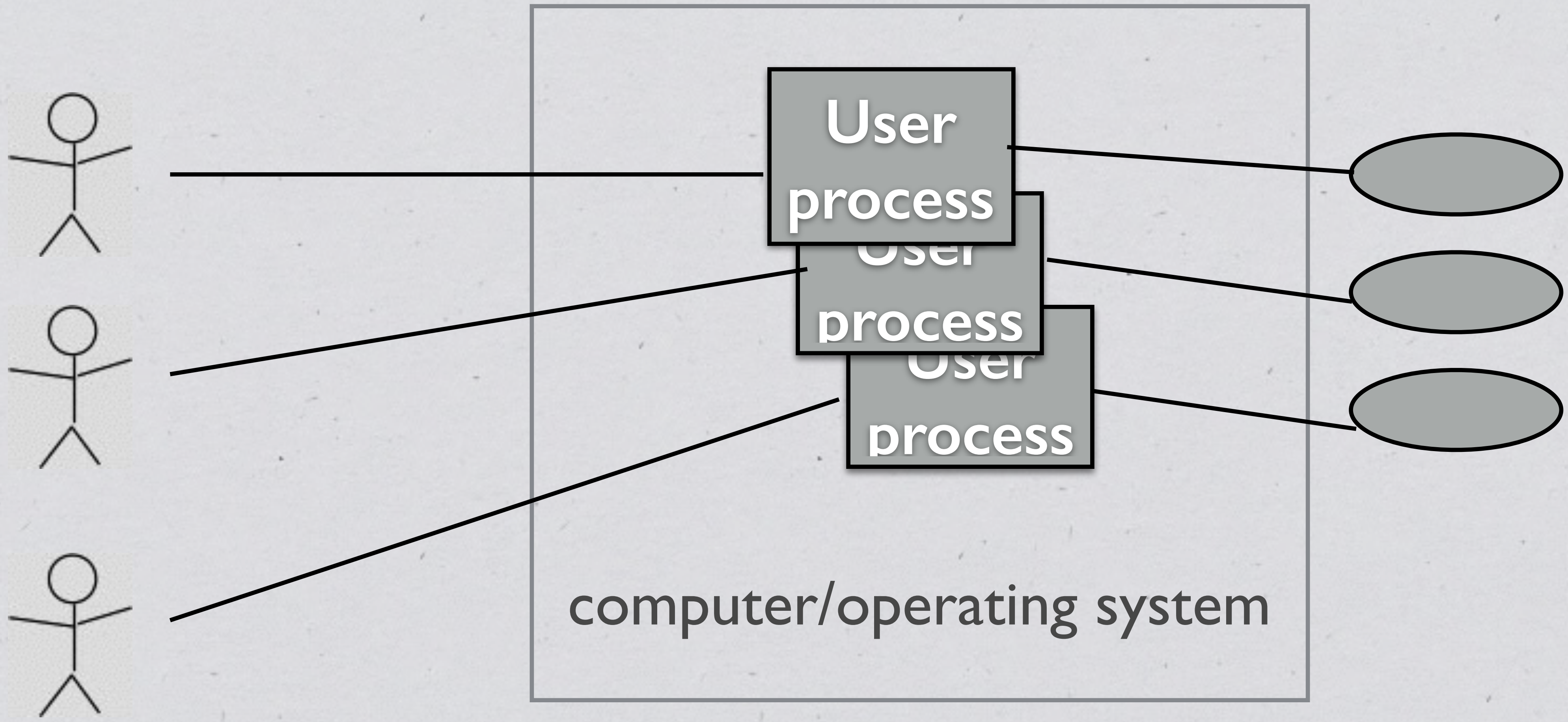
HOW OLTP WORKS

Early OLTP Monitor paradigm



One OS "process" does all the work of managing terminals and data accesses; application segments run on a monitor "thread" for each user but own no resources

Early Time Sharing paradigm

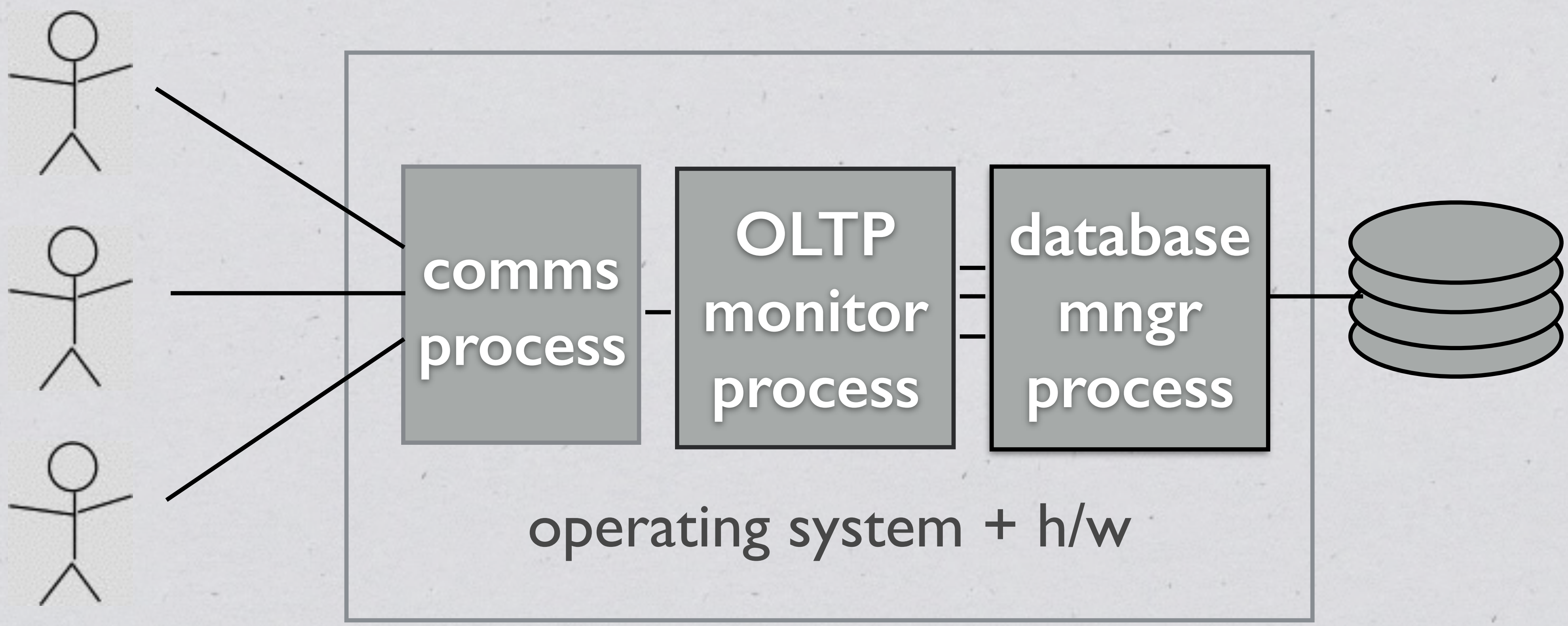


- Each end user has their own process and data; operating system gives each a “time slice” to share processor; no data sharing

Networking for OLTP

* There was a clear need to shield "applications" from data link control, traffic routing, and message assembly/disassembly
* There was also a need to **share the network** between different "applications" [where "application" means a class of customer applications running under an OLTP monitor or subsystem]
* This led IBM to develop SNA, a layered networking architecture implemented by VTAM/370 and programmable devices
* SNA became the model for OSI and other network architectures

Slightly more advanced OLTP



Comms process owns network; application segments run on one monitor thread per user; they share "access ports" to database manager process, which owns data resources

OLTP programming model

* TP Monitor acquires and retains shared resources
  * applications, memory, processes, threads, files, databases, communications channels, etc.
  * on **receipt of a transaction request message**, initiates application segment and provides concurrent access to these resources
  * frees resources when response message sent
  * so application segment is message in/message out, or "**stateless**"
* Larger applications ("pseudo-conversations") can be created
  * by **retaining some state data** in a "scratchpad area"/cookie or message
  * next segment retrieves state, processes message and issues response
  * different from conversational applications which retain all state

Application Programming

- OLTP application model **doesn't fit** with batch application programming or conversational interactive programming
    - Uses **modified runtimes** for High Level languages
    - OLTP Monitor provides **additional** statements and functions
    - So application language is a **modified form of HLL**, e.g. CICS/COBOL, Tuxedo/C
- Further mechanisms needed for large scale applications

WAS IT JUST MAINFRAMES?

Competition for mainframe TP

* In the 1970s and 80s, mainframes were the *de facto* business machines but other vendors saw opportunities to compete:
    * **Mainframe compatible** vendors, e.g. Amdahl, Fujitsu, produced machines that were faster than IBM's
    * **Specialist vendors**, e.g. Tandem, Stratus, produced highly reliable ["non-stop"] machines for financial systems
    * **Midrange** vendors, e.g. DEC, HP, produced machines that were cheaper for medium sized enterprises; many of these used a hybrid *Time Sharing* paradigm with a DBMS

The mainframe response

* Mainframe OLTP monitors ran best on **fast uni-processors**, which used water-cooled bipolar logic technology, but competitive processors were faster than IBM's and air cooled

* IBM's response was to move to **multi-processor systems** using cheaper (but slower) CMOS technology; this required huge changes to system software but succeeded in lowering costs

* Tandem systems were frequently used as **front ends** for mainframes, but the new multiprocessor mainframes provided more reliability so reducing Tandem's competitive advantage

The UNIX era

- **Bell Labs** produced the first versions of Unix and the C language compiler in 1972, but it wasn't a product until the 1980s
- Unix was widely used in universities and smaller enterprises for interactive and time-sharing systems, but **not for business**
- 1983 Bell Labs developed **Tuxedo** as a TP monitor for an internal application and, later, as a product
- Other Unix based TP monitors appeared in the 1990s, e.g. **Encina, CICS/AIX** for distributed processing

More competition for OLTP

- **Application vendors** had mainly targeted mainframe OLTP customers, with products based on OLTP monitors
- Some key application vendors, e.g. **SAP**, started to offer Unix versions of their products with **built-in OLTP functionality**, so no requirement for an OLTP monitor platform
- Most of these vendors developed their own OLTP function; a few licenced a monitor for inclusion with their application
- This became the chosen style for most **packaged applications** in the Unix environment

Distributed systems

- “Moore’s Law” improved processor speed much sooner than any improvements in network costs and bandwidth
- In the 1990’s fast dedicated long distance lines were still **only rated at 64 kbps** BUT cheap PCs were easily available
- By contrast, **Local Area Networks** (e.g. Ethernet, Token Ring, Netware) could achieve **up to 10 mbps**
- The meant it was usually cost effective to place **distributed processors** in branch offices and centres, e.g. supermarkets

Distributed vs. Centralised

* **Advantages** of distributed systems
  * Better response times for local tasks
  * Better availability for local applications
  * Cost savings by exploiting commodity technology
* **Disadvantages** of distributed systems
  * Increased latency for some tasks
  * Duplication of systems leads to potential sources of error
  * Increased systems management overhead

THE INTERNET REVOLUTION

The World Wide Web era

- Early Web Servers (1991) used a **similar paradigm** to early OLTP monitors:
  - single server process handled all requests
  - communication used request/response message pairs with connection broken after each request
  - static read-only data "pages" were held on disk
- BUT the WWW pioneers **knew nothing about Enterprise Systems** ...

Impact of WWW on Enterprise Systems

- By the mid-1990s, it was clear that the WWW could be used for transactional business, e.g. selling pizzas
- Web servers used a "TP Lite" (inquiry only) paradigm but **couldn't support applications** or handle updates
- **CGI exits** and links with a DBMS were introduced, plus "cookies" to enable pseudo-conversations
- Some commentators saw this as an **opportunity** for traditional OLTP monitors; others as creating a need for a **new breed** of "internet application servers"

Opportunity for OLTP on Web

* The WWW **greatly increased the market** for OLTP style transactional applications and lowered delivery costs:
    * free "any-to-any" network
    * web browser provides "virtual terminal"
    * access to much larger market
* Leading to demands for:
    * much increased scalability, broadband networks
    * better application programming models and methods

Applications servers emerge

- eBay and Amazon became leading commerce platforms by building scalable infrastructures to support their applications
- Other **open application servers** were developed to support applications written in Java, C# and related languages, eg:
  - WebLogic
  - WebSphere
- Many of these recreated the main features of established OLTP monitors, because of the requirement for scale

Mainframe OLTP response

* Established OLTP monitors **didn't support HTTP** or other web protocols, so couldn't communicate with web browsers, nor run applications in **popular web programming languages**

* Their immediate need was for "**gateway**" technology to enable connectivity, usually a special purpose monitor or WAS

* BUT their **reliability, scalability and mature applications** were key advantages which worked in their favour

* **A few OLTP monitors**, e.g. CICS, were also **enhanced** to support new protocols, languages and even greater scalability

2005 - and later

- The leading "**computer companies**" now include Apple, Amazon and Google, as well as Microsoft, Oracle, HP and IBM
- Internet-based OLTP is the **de facto standard** for most business applications, e.g. Travelport processes **1 Bn transactions/day**
- **Questions**:
  - What contributed most to this: Moore's Law? The World Wide Web? Broadband networks?
  - How did this pave the way for virtualisation, cloud applications, mobile, and the Internet of Things?

THANK YOU!