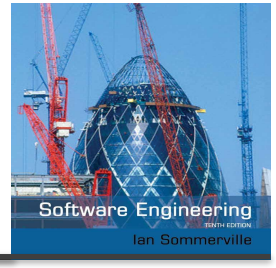
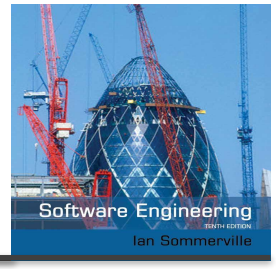


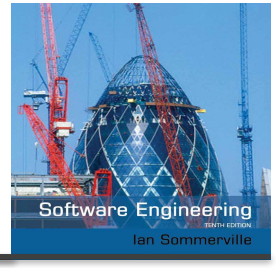
Application architectures

Previous review



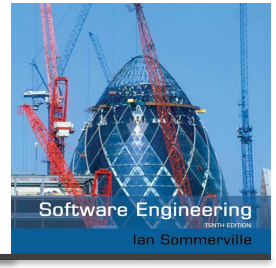
- **describes the patterns and techniques used to design and build an application**
- The architecture **gives you a roadmap and best practices to follow** when building an application, so that you **end up with a well-structured application**.
- Software design patterns can help you to **build an effective application**.

Application architectures



- **Application systems are designed to meet an organizational need.**
  - Somethings are common in all business like hiring people, invoice issuance, keep accounts etc.
  - That's why their system have much requirements in common
- For example, **all phone companies** need systems to connect & meter calls, manage network, issue bills to customers.
  - That's why **multiple phone companies tend to have a generic architecture** on which they add their customized requirements to make their product.
- A **generic application architecture** is an architecture for a type of software system that **includes principle characteristics of a class of systems**.
  - For e.g., real time systems might have generic arch. Models of different systems like data collection system, data monitoring system etc.
  - A system for supply chain management can be adapted for different suppliers, different goods, and contracts.

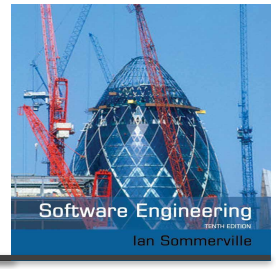
Use of application architectures



- As a starting point for architectural design.
  - If you are unfamiliar with the type of application you are developing then chose a baseline generic architecture and add your features into that to take it to final design.
- As a design checklist.

- As a way of organizing the work of the development team.
- As a means of assessing components for reuse.
- As a vocabulary for talking about application types.

Examples of application types



Data processing applications

- **Data driven applications** that **process data in batches without** explicit **user intervention** during the processing.

Transaction processing applications

- **Data-centered applications** that **process user requests** and **update information in** a system **database.**

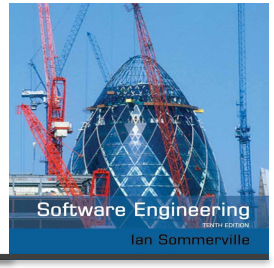
Event processing systems

- Applications where system actions depend on interpreting events from the system's environment.
- E.g., trigger an alarm in system on some sensor input etc.

Language processing systems

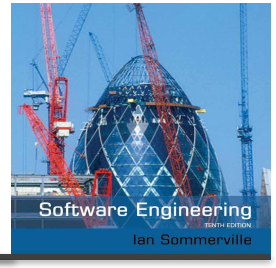
- Applications where the **users' intentions are specified in a formal language** that is **processed** and **interpreted by the system**.

Application type examples



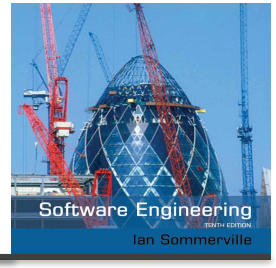
- Two very widely used generic application architectures **are transaction processing systems** and **language processing systems.**
- Transaction processing systems (web based business systems)
  - E.g., interactive banking systems, E-commerce systems, information systems & booking systems.
- Language processing systems (software development rely on these systems)
  - Users' intentions fed in as a formal language like any Programing Language.
  - They process the language into internal format and depicts as internal representation.
  - They are also used to interpret command languages like XML
  - E.g., Compilers (HLL to machine code)

Transaction processing systems

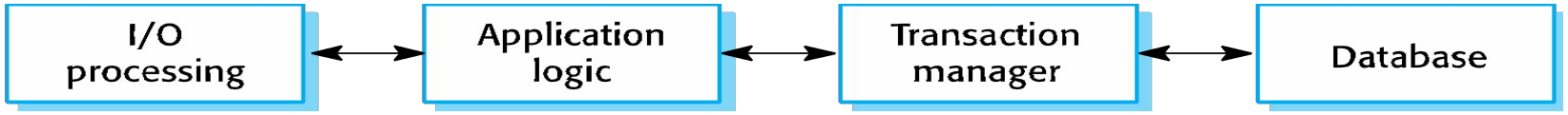


- **Database centered applications**
- Process user requests for information from a database or to update the database.
- It's the **most common type of interactive business systems**
- Technically, **transaction is a sequence of operation treated as atomic operation** to ensure consistency in case of failure of a transaction.
- All operations in a transaction must be completed before database changes are made permanent.
- From a **user perspective** a **transaction is**:
  - **Any coherent sequence of operations that satisfies a goal**;
  - For example - find the times of flights from London to Paris.
- Users make asynchronous requests for service which are then processed by a transaction manager.

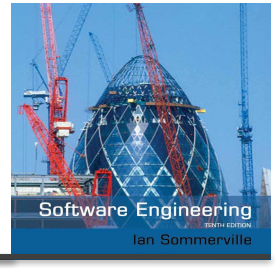
The structure of transaction processing applications



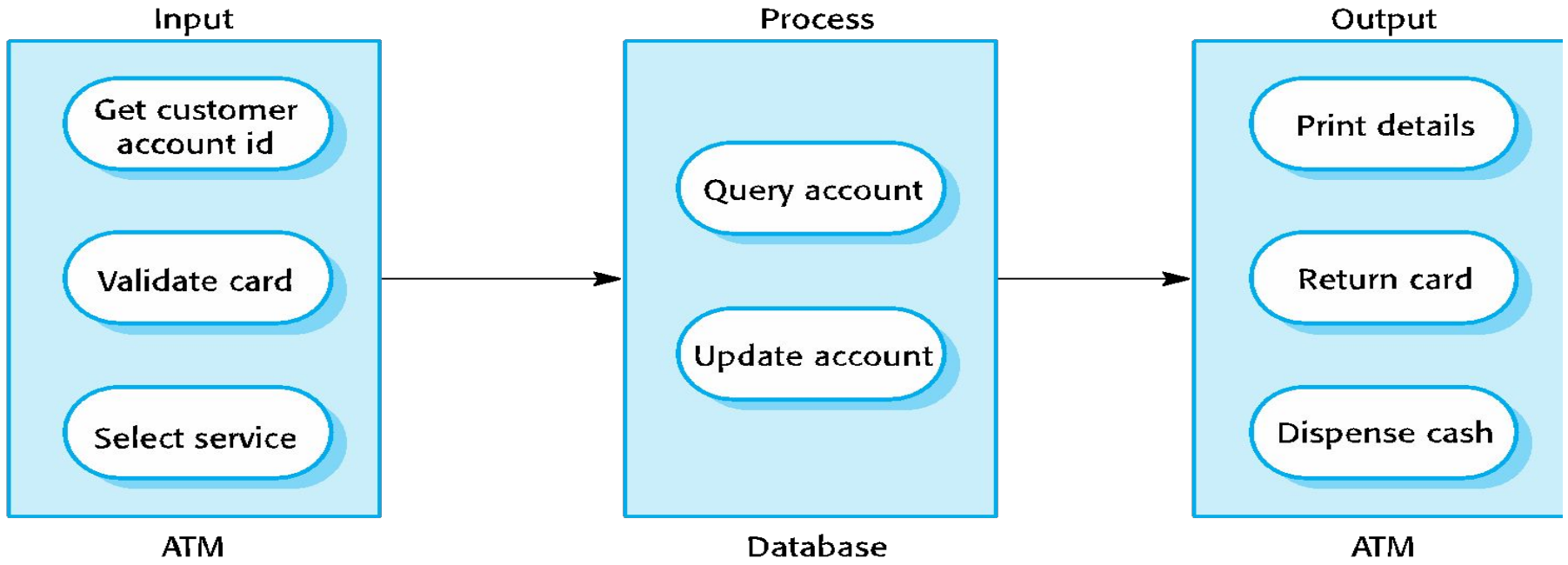
- User makes request using I/O component
- Request process by some application specific logic
- A transaction created and passed to transaction manager for processing.
- After successful transaction completion, a message is passed to application logic



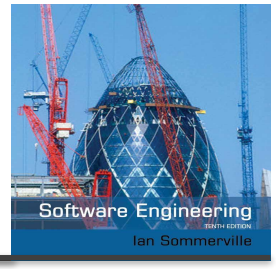
The pipe and filter architecture of an ATM system



- Transaction processing system could be modelled as pipe & filter architecture with system components responsible for input, processing & output.
- System composed of two cooperating software components: ATM and Processing logic
- Input output components are the installed software components in ATM whereas Process is the software component in Bank's database server.

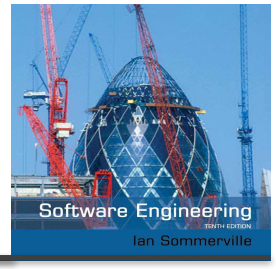


Information systems architecture

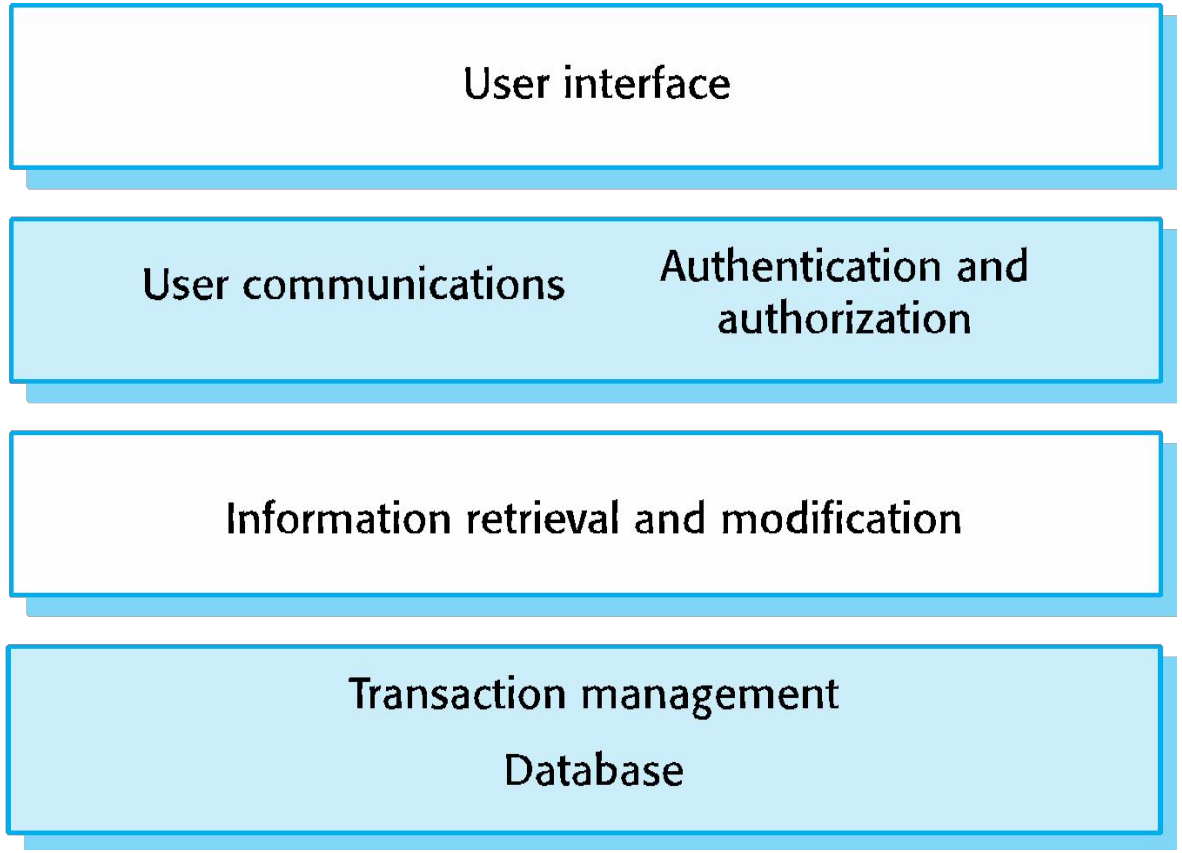


- Transaction processing systems provide information to Information systems for decision making. IS usually generate reports
- **These include transaction processing systems,** as interaction with these systems generally **involves database transactions.**
- Layers include:
  - The user interface
  - User communications , Authentication & Authorization ~ pass all user queries & check validations
  - Information retrieval ~ application logic for accessing & updating data
  - System database

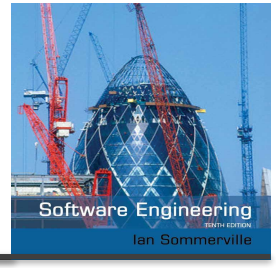
Generic Architecture of information system

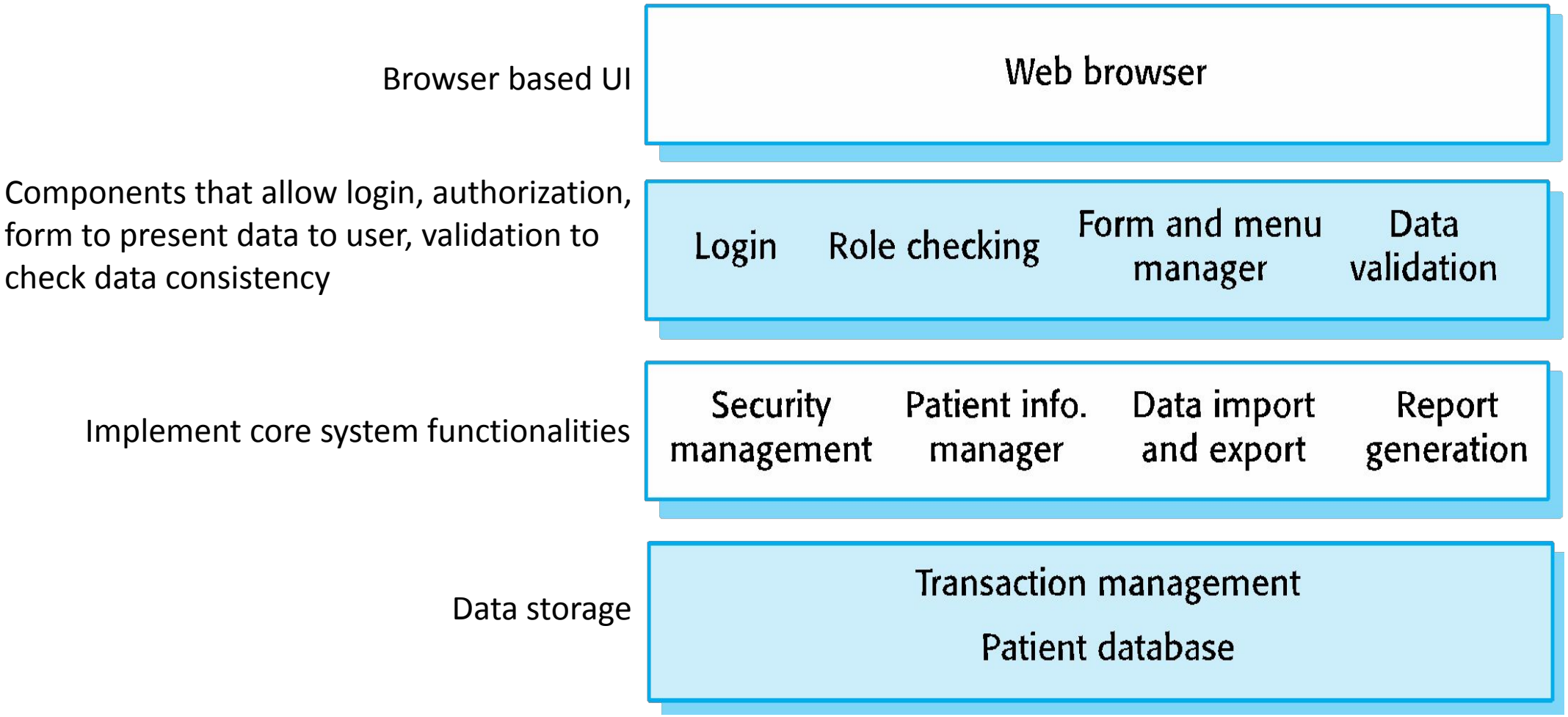


- Generic architecture
- Often implemented using a distributed with multi tier client-server architecture
- Segregated to web server (top most layer), application server (mid 2 layers) and the Database server (for bottom layer)

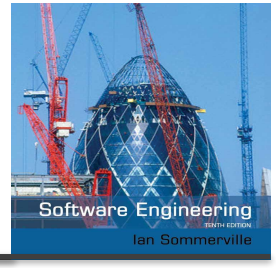


The architecture of the a Mentcare application



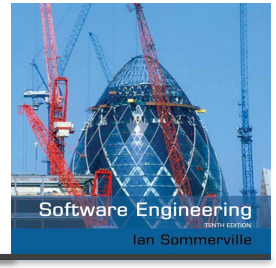


Web-based information systems



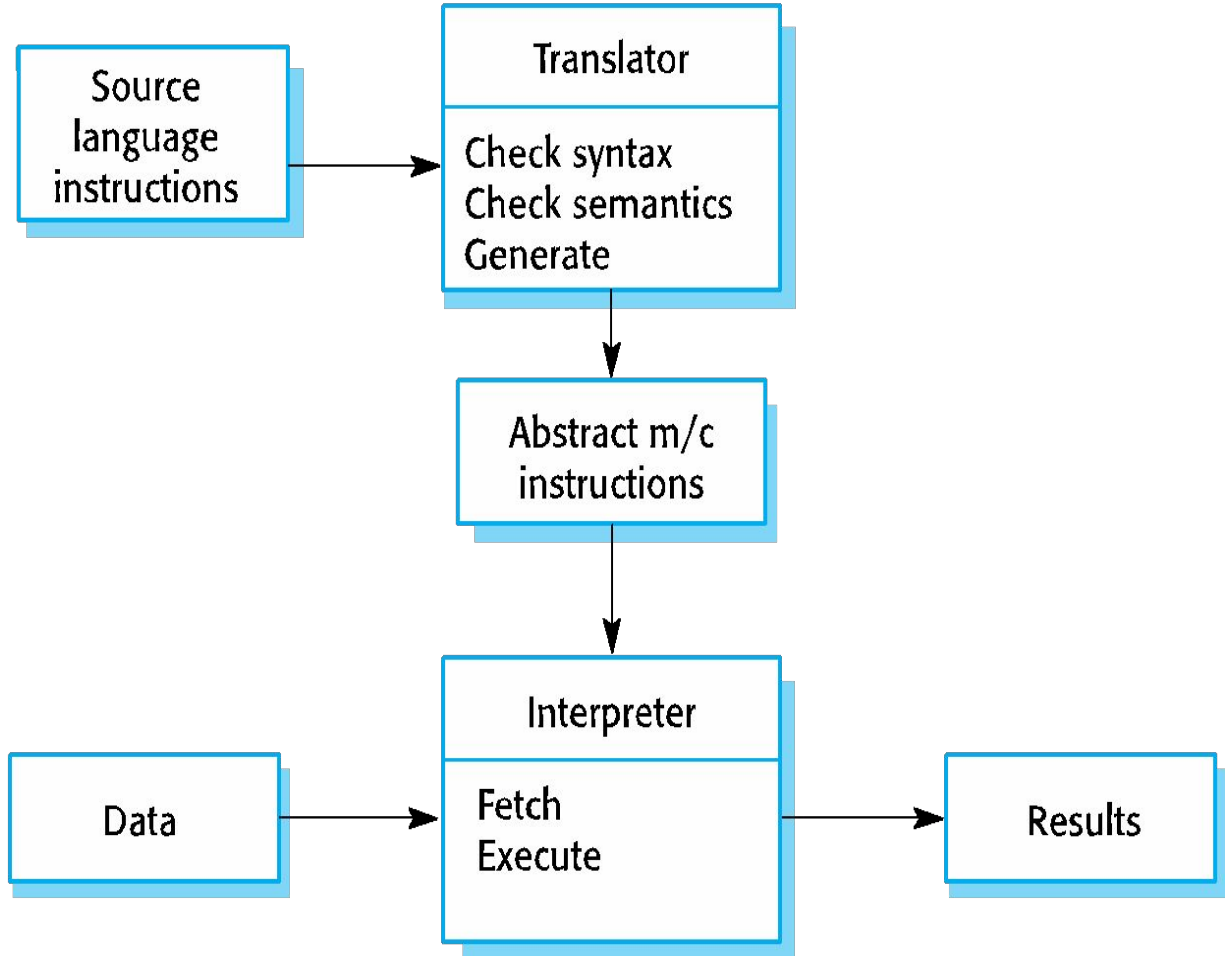
- **Information and resource management systems are now usually web-based systems** where the UIs are implemented using a web browser.
- For example, **E-commerce systems are Internet-based resource management systems** that accept electronic orders for goods or services and then arrange delivery of these goods or services to the customer.
  - In an e-commerce system, the **application-specific layer includes**
    - additional functionality supporting a '**shopping cart**', add multiple items in separate transactions but pay using 1 transaction at end.

Language processing systems

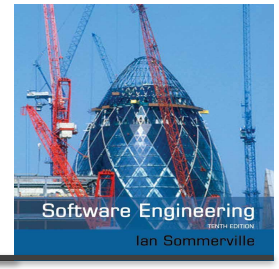


- Language processing systems are used to translate texts from one language into another and to carry out the instructions specified in the input language.
- They include a translator and an abstract machine that executes the generated language.
- Used: in systems where the easiest way to solve a problem is to describe an algorithm or describe the system data

The architecture of a language processing system

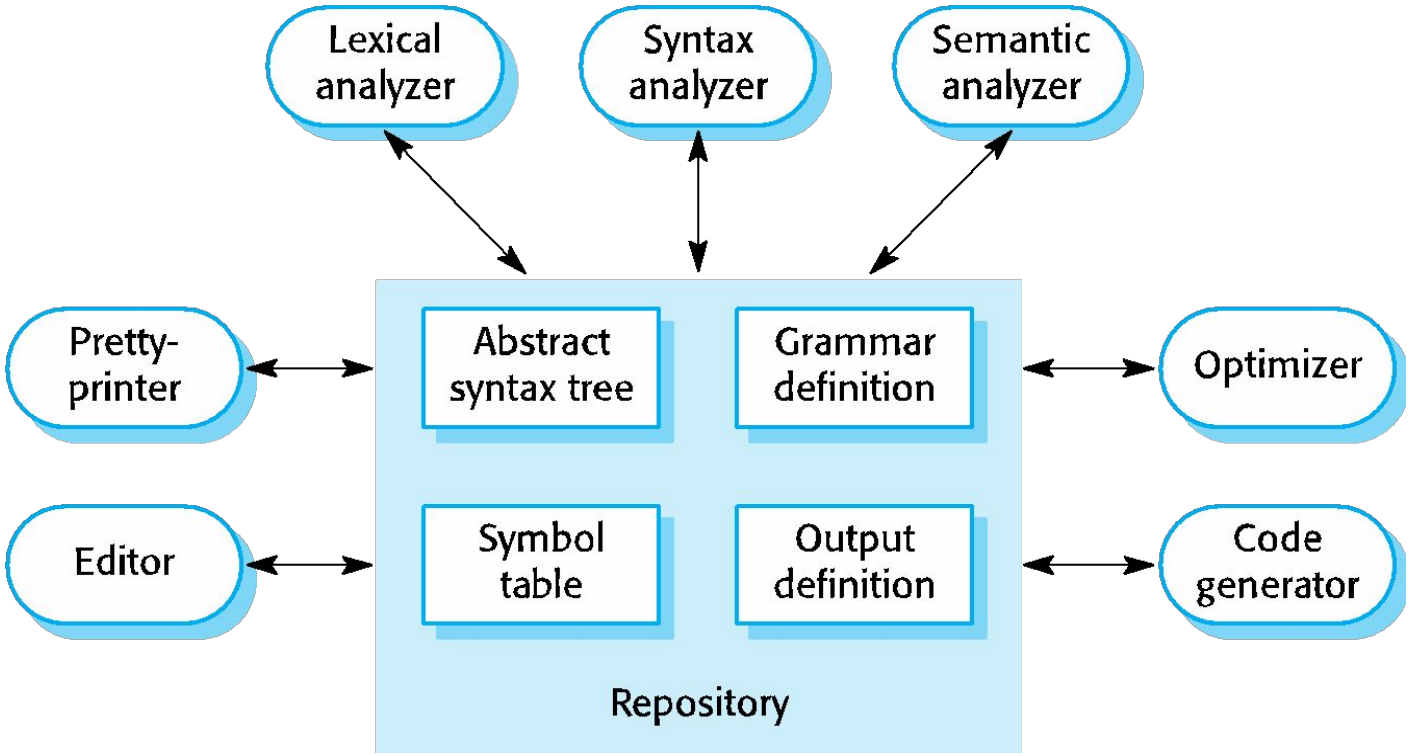


Compiler components



A lexical analyzer	• converts the High level input program into a sequence of Tokens..
A symbol table	• which **holds information about the names of entities (variables, class names, object names, etc.)** used in the text that is being translated.
A syntax analyzer	• check syntax if its correct
A syntax tree	• an internal structure to represent compiling program.
A semantic analyzer	• use info from the **syntax tree** and the **symbol table** • to **validate the semantic correctness** of input text.
A code generator	• traverse the syntax tree • generates abstract machine code.

A repository architecture for a language processing system (integrated set of programming support tools)



Grammar definition & output definition might be sometimes embedded in tools.

A pipe and filter compiler architecture with composite repository