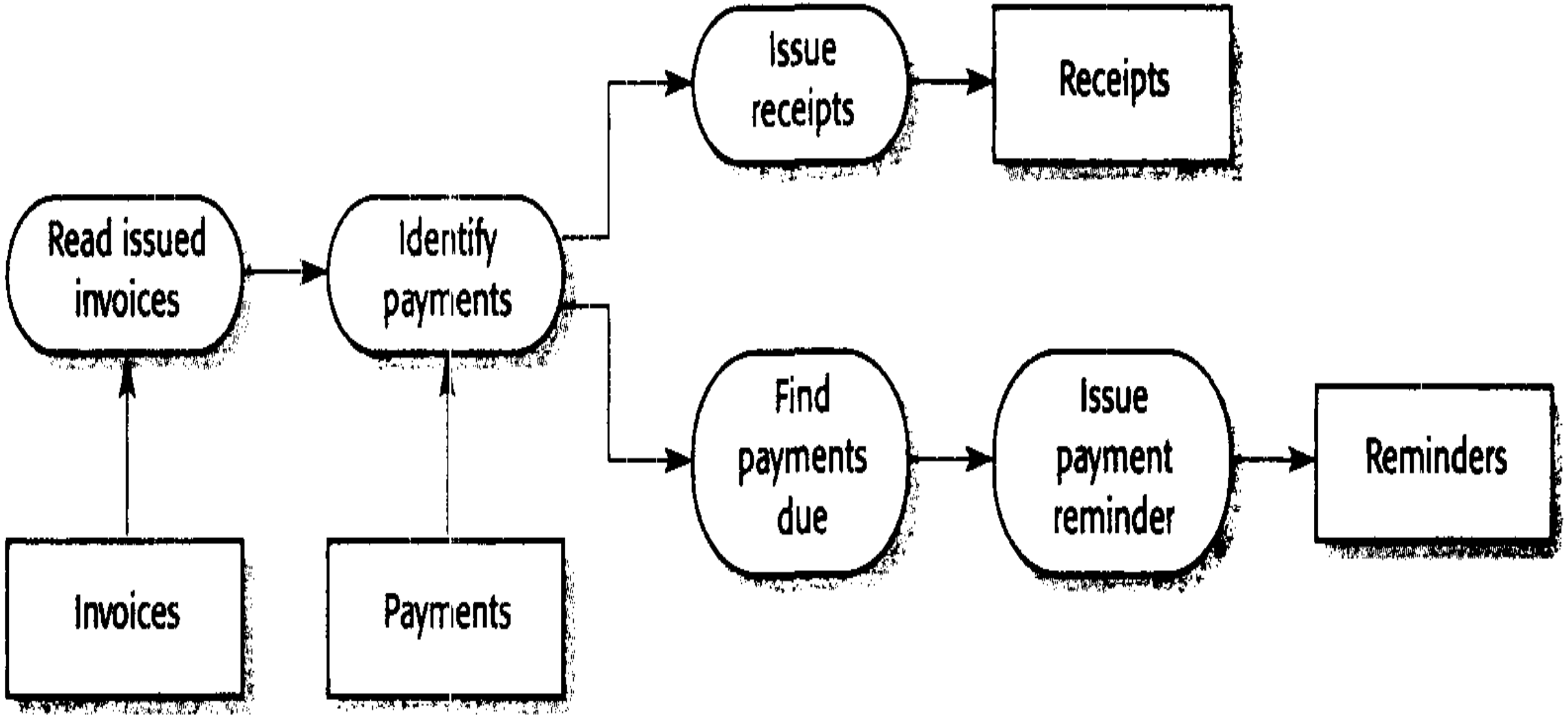
Function Oriented Pipelining

- Functional transformations process their inputs and produce outputs.
- Data flows from one to another and is transformed as it moves through the sequence.
- Each processing step is implemented as transform and data flows through these transforms.
- These transforms may execute sequentially or in parallel

Function Oriented Pipelining

- Advantages are supports reuse, simple to implement, adding new transformation is usually straightforward, intuitive in terms of input and output processing.
- Disadvantage is that there must be common format for data transfer.
- Interactive systems are very difficult to write using pipelining model because of the need for stream of data to be processed
- Following is the diagram for function-oriented pipelining

Function Oriented Pipelining



Control Styles

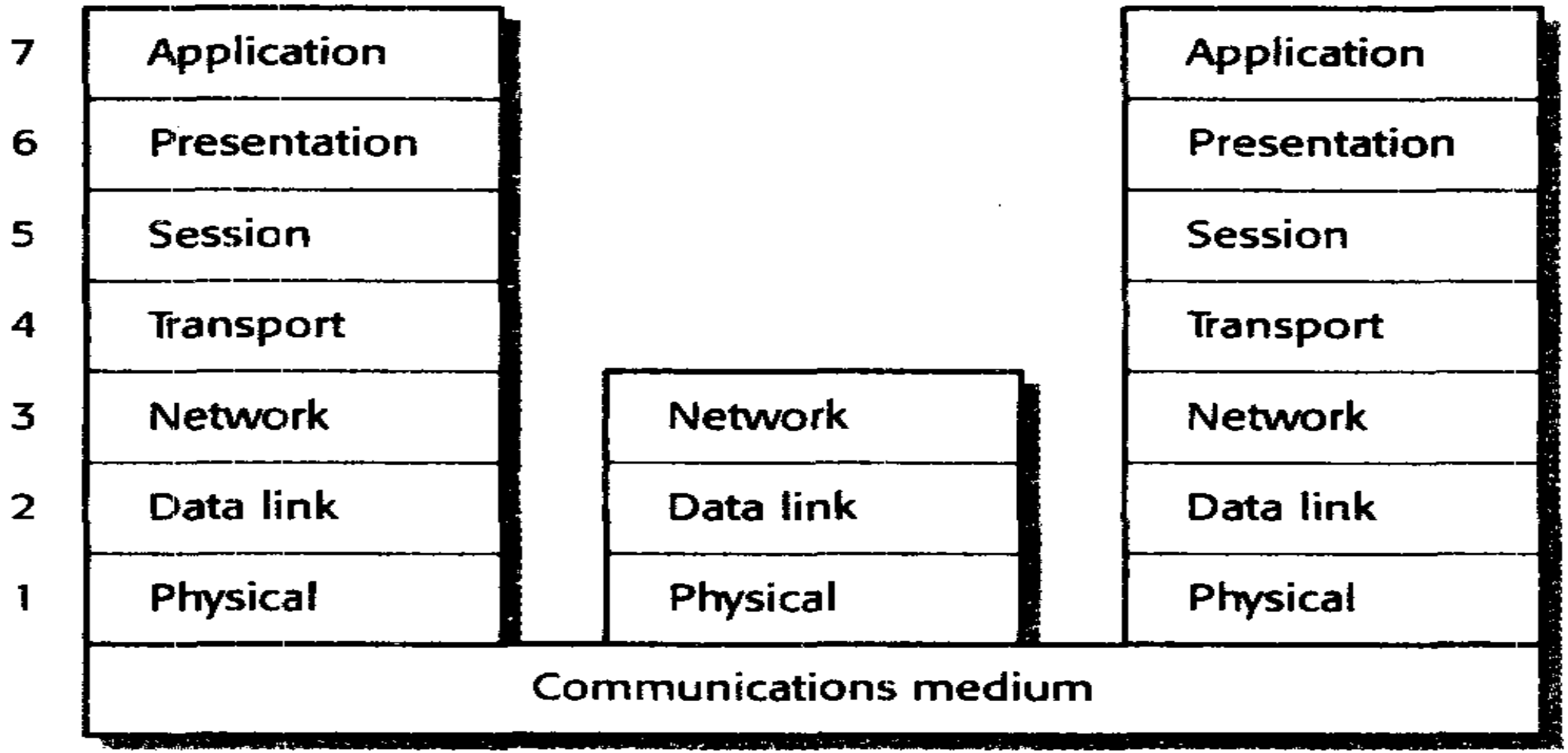
- They are used to control the information in the structural model with the help of some control model that supplements the structural model
- Two generic control styles are used in software systems
  - **Centralized Control** – One system has the overall responsibility of the system and may also delegate control to another sub system but expects to have control responsibility returned to it. The strength is relatively simple to analyses the flow of work and work out how the system will respond to particular inputs. Its weakness is even normal operations are awkward to handle
  - **Event Based Control** – Each system responds to some externally generated events coming from sub systems or from their environments. Advantage is evolution is simple that is any new sub system can be integrated with old system. Disadvantage is multiple registering may cause conflicts

Reference Architecture

Reference Architecture

- The previous models known as general models are applied to many classes of application
- Architectural model that are specific to a particular application domain may also be used. These models are called domain specific architectures.
- Two types of domain specific architectural model
  - **Generic models** – Abstractions from a number of real systems. They encapsulate the principal characteristics of the system such as data collection or monitoring systems
  - **Reference models** – They are more abstract and define larger class of systems. They are a way of informing designers about the general structure of that class of system. They are derived from a study of the application domain. They represent an idealized architecture that includes all the features that a system might incorporate
- The following is an example of OSI Reference model architecture

Reference Architecture



User Interface Design

- Computer system design encompasses a spectrum of activities from hardware design to user interface.
- Large organisations normally employ specialist interface designers for their application software.
- Therefore software engineers must take responsibility for user interface design as well as for the design of the software to implement that interface.

User Interface Design

- Important factors that engineer must consider are
  - People have a limited short term memory hence if you present users with too much information they may not be able to take all of it
  - When systems go wrong and issue warning messages and alarms this puts more stress on the users thus increasing the chances that they will make operational errors
  - Some people see and hear better than others, some people are colour blind and some are better at physical manipulation. Hence you should not design for your own capabilities and assume that all other users will be able to cope
  - Some people like to work with pictures, others with text. Direct manipulation is natural for some people but others prefer a style of interaction that is based on issuing commands to the system

User Interface Design Principles

Principle	Description
User familiarity	The interface should use terms and concepts drawn from the experience of the people who will make most use of the system.
Consistency	The interface should be consistent in that, wherever possible, comparable operations should be activated in the same way.
Minimal surprise	Users should never be surprised by the behaviour of a system.
Recoverability	The interface should include mechanisms to allow users to ecover from errors.
User guidance	The interface should provide meaningful feedback when errors occur and provide context-sensitive user help facilities.
User diversity	The interface should provide appropriate interaction facilities for different types of system users.