ECSE-322

Lecture 4

Data Structures – Messages

11 January 2008

Cool is on!

Office hour: Monday 10:35 TR 4105

messages + structures

information transmission

start/stop

encoding

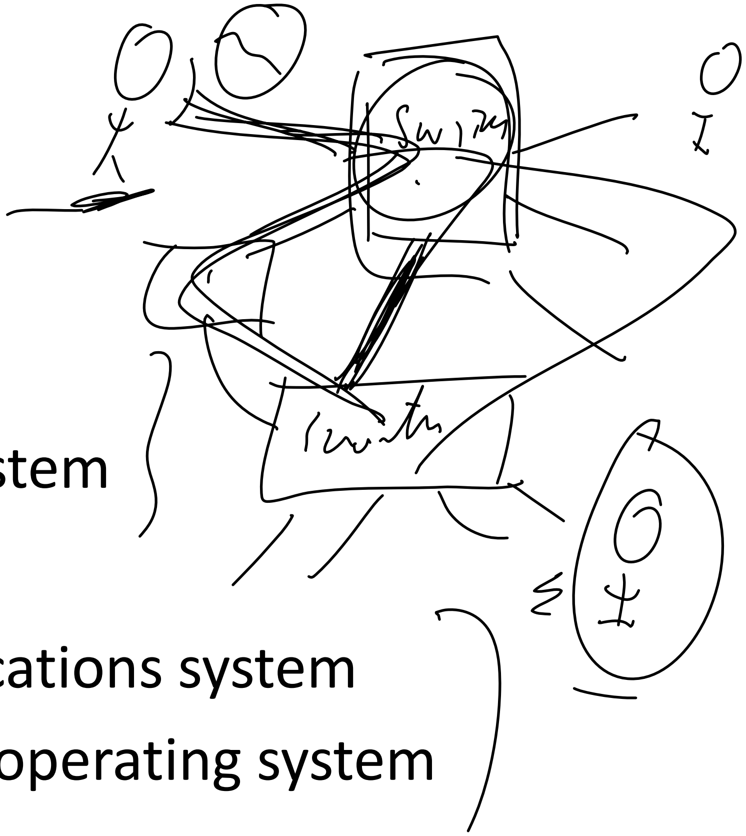
- synchronous vs asynchronous

Data Structures

- Methods of organizing data
  - A structure like any other in engineering
  - Requires design
    - What is the data for?
    - What operations will be needed?
    - What are the properties of the data being stored?
  - Provide means for finding particular data items
  - Allow information to be restored.

Data Structures

- Examples
  - Hardware implementations
    - Page buffer in the printer
    - frame buffer in a graphics system
  - Software
    - A packet switched communications system
    - The process structures in an operating system

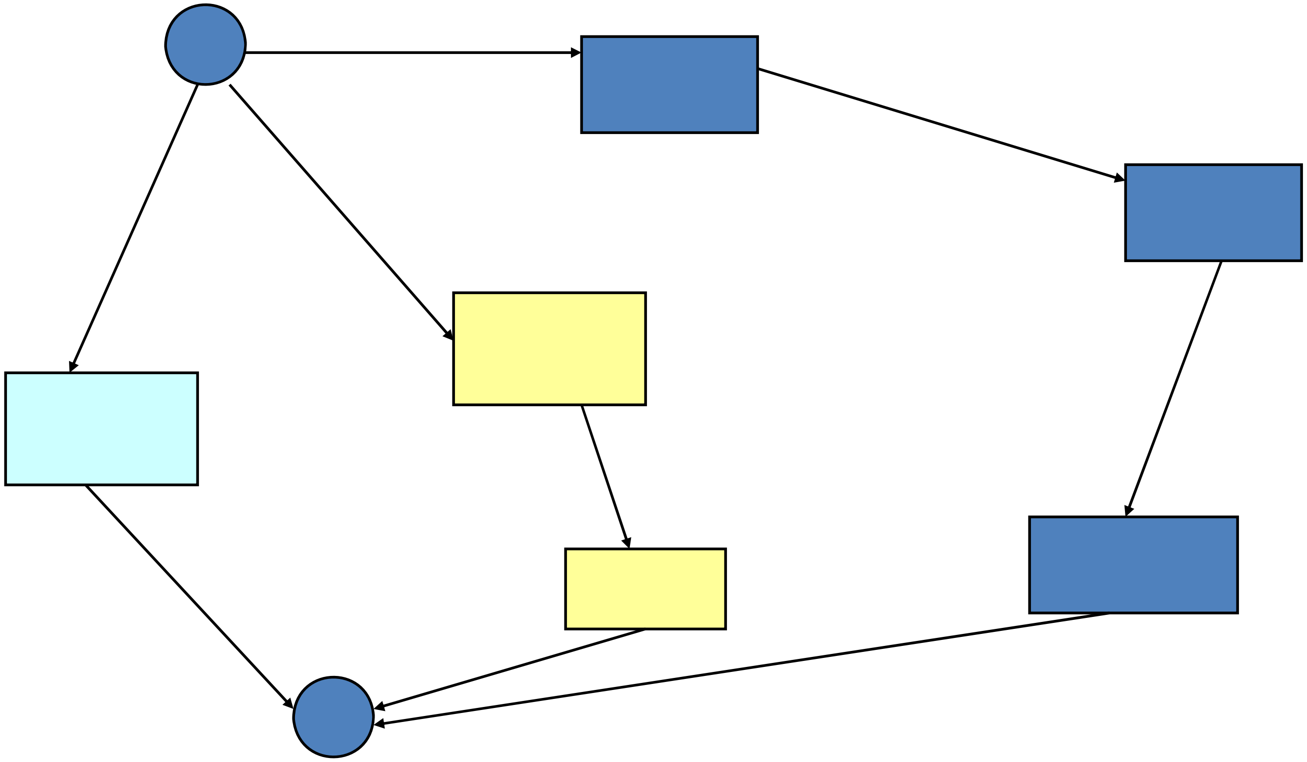


Example - A Packet Switched System

- Data is transmitted in blocks (packets)
- Each packet can be sent by a different route to the destination
- Each packet can arrive at a different time
- Requirement:
  - Design a data structure which will enable the original message to be put together correctly

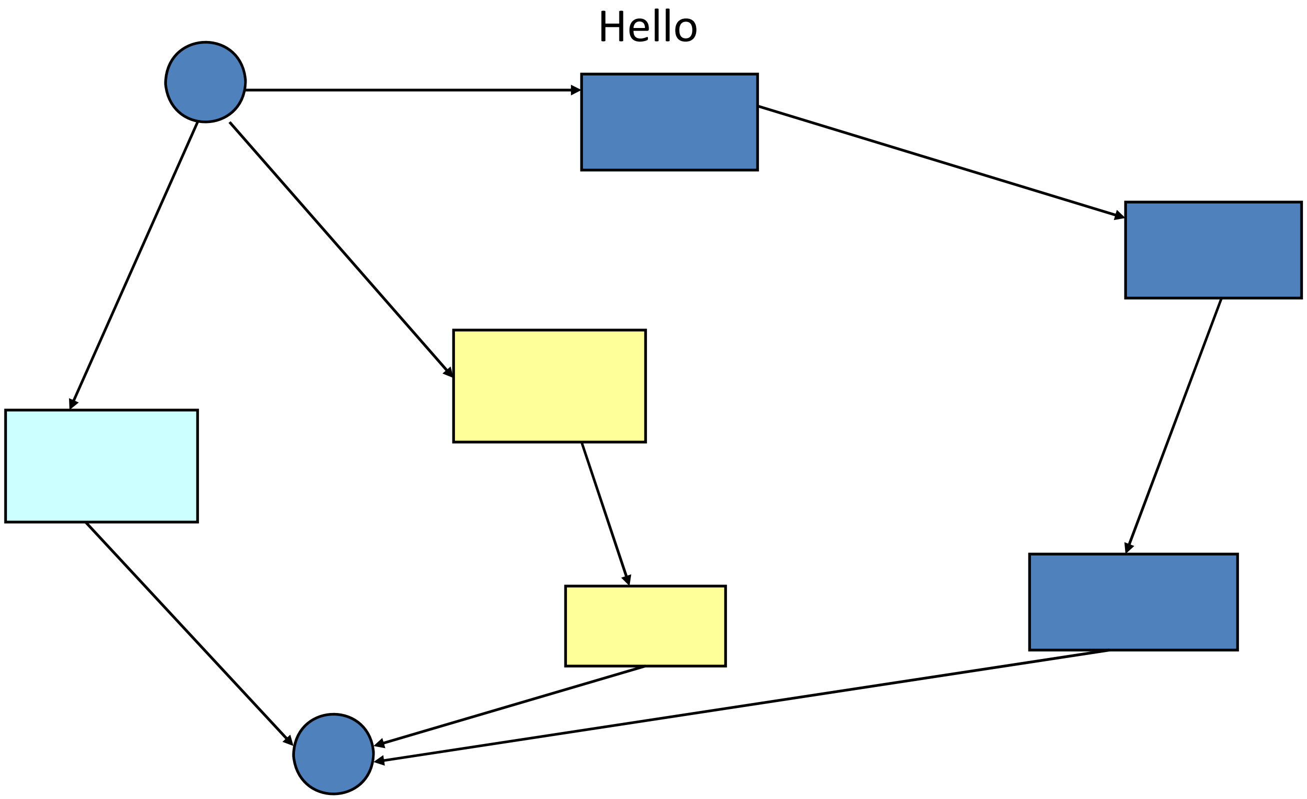
Example - A Packet Switched System

Hello World, please respond.



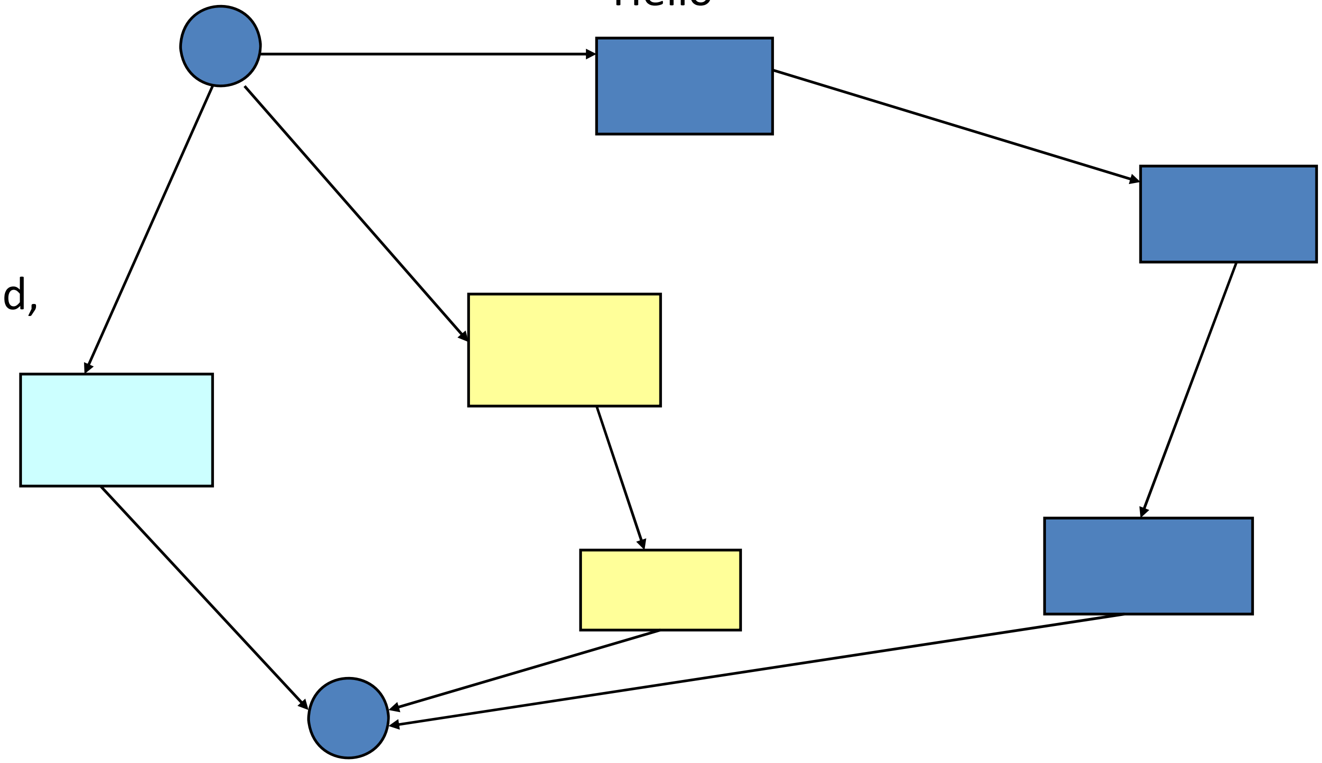
Example - A Packet Switched System

World, please respond.



Example - A Packet Switched System

please respond.

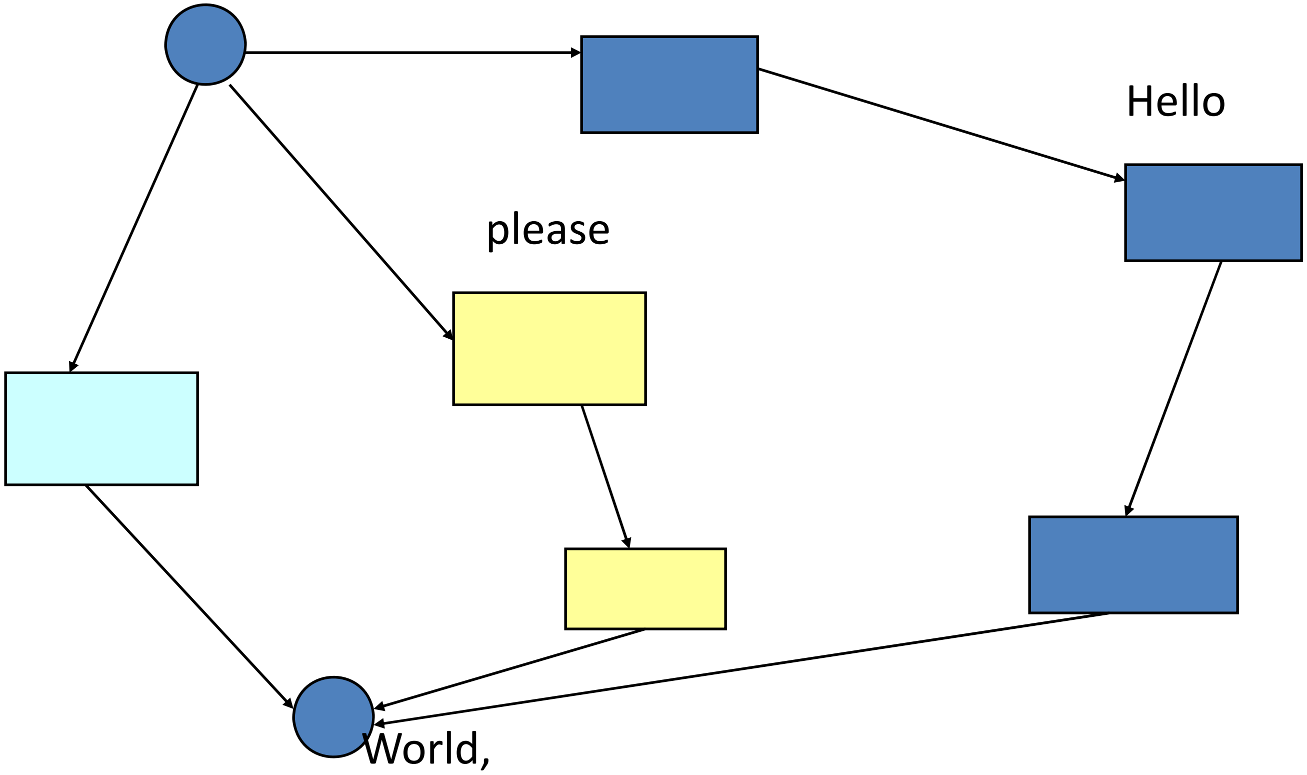


Hello

World,

Example - A Packet Switched System

respond.

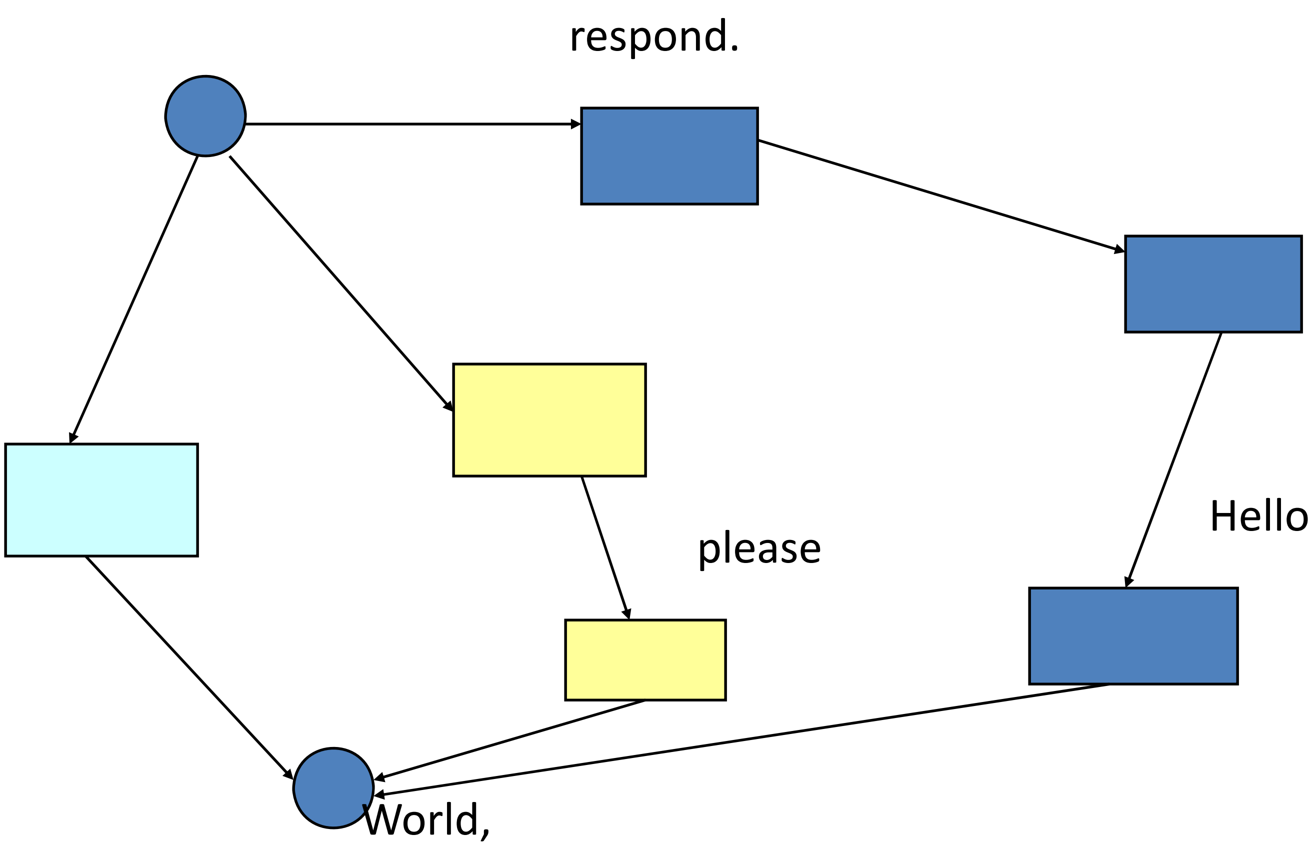


Hello

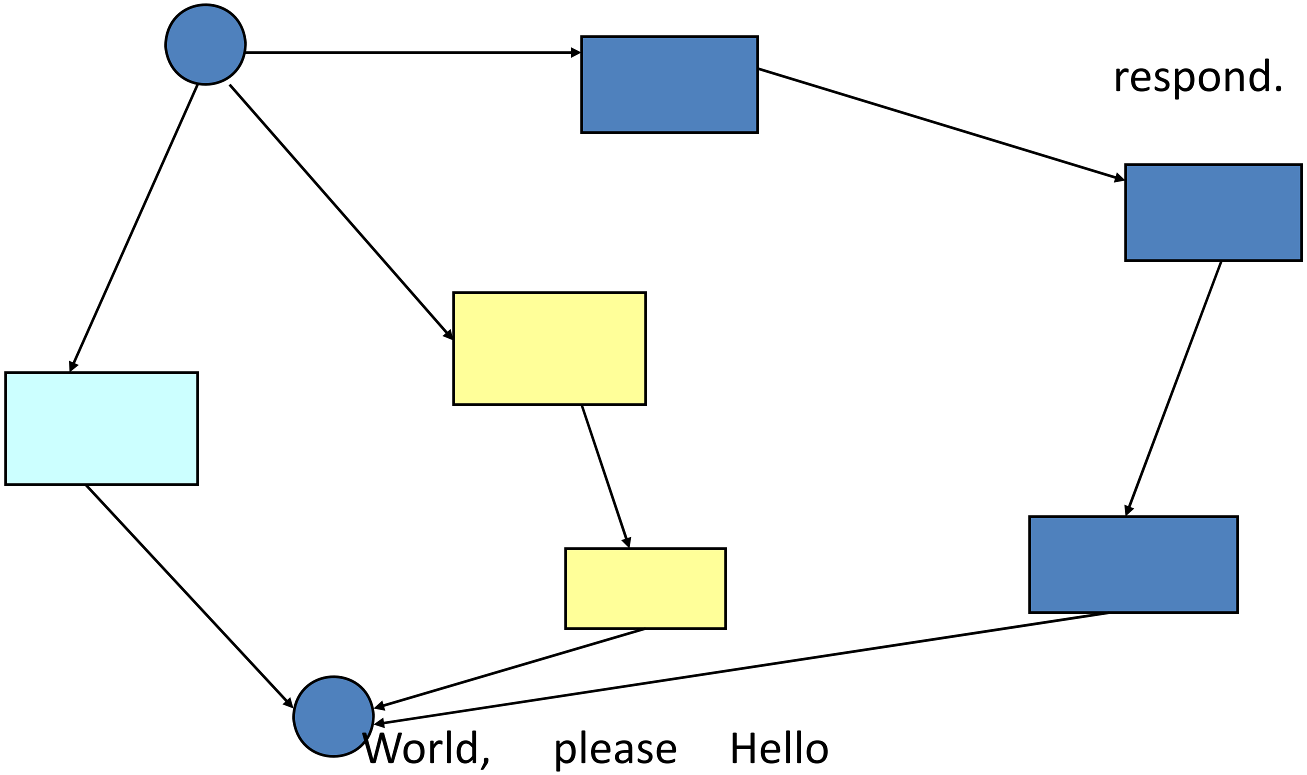
please

World,

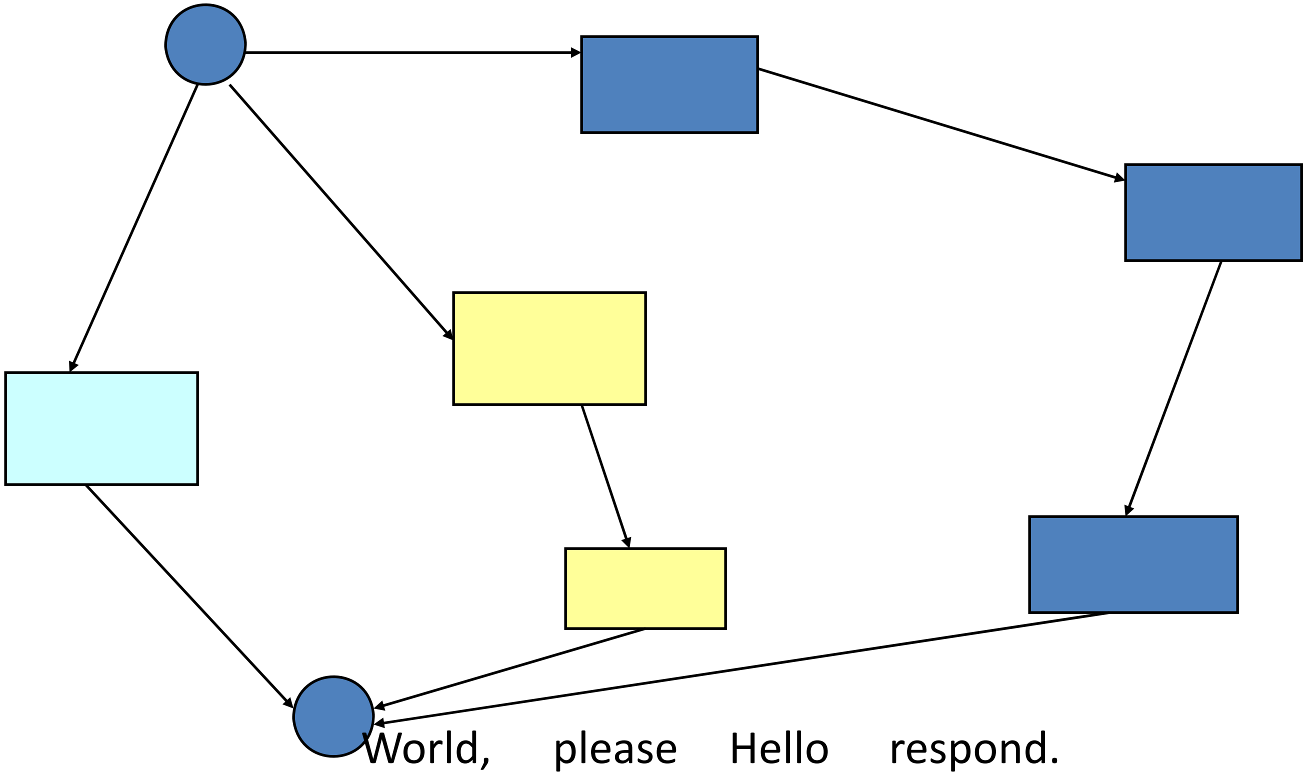
Example - A Packet Switched System



Example - A Packet Switched System



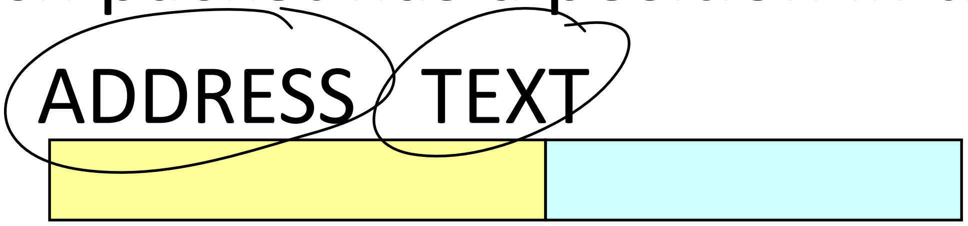
Example - A Packet Switched System



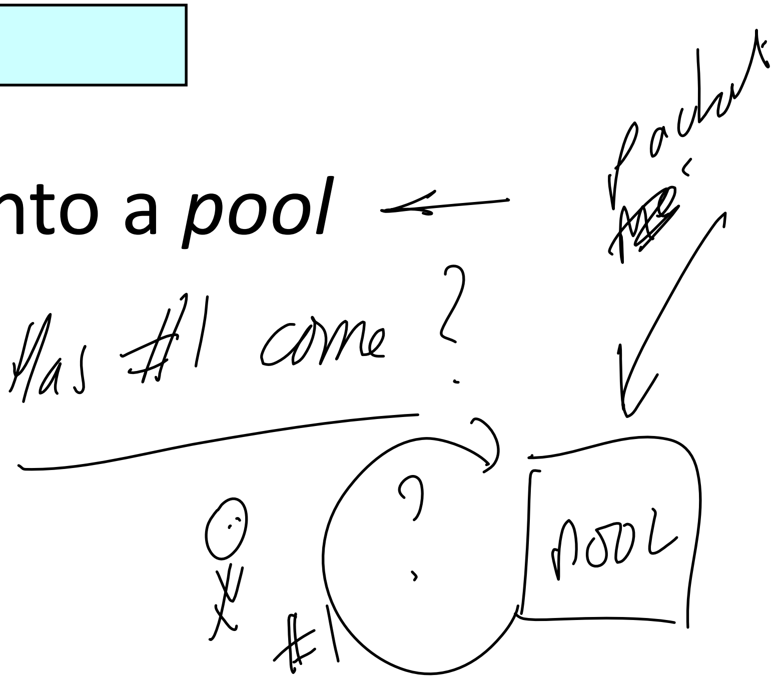
World, please Hello respond.

Example - A Packet Switched System

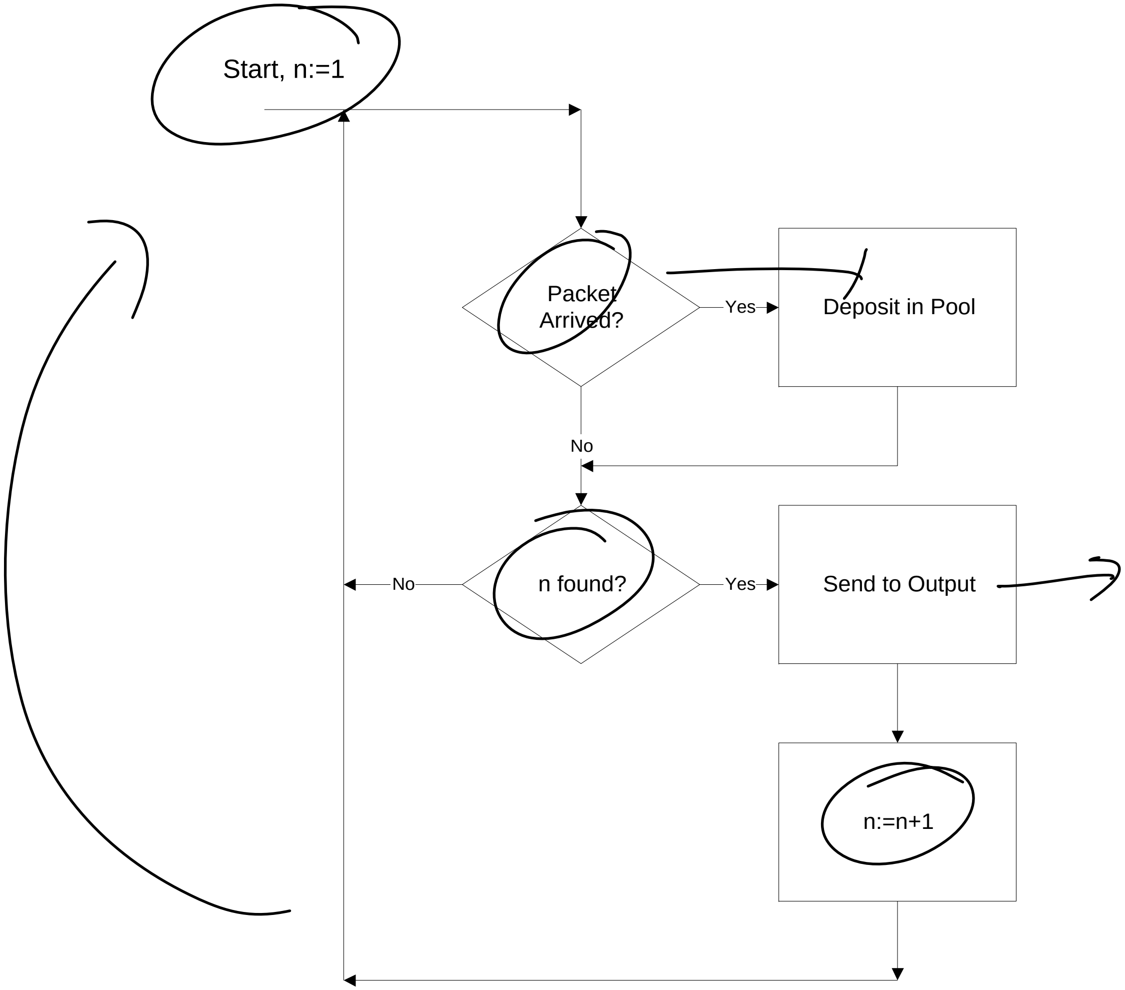
- Each packet is a text string
- Each packet has a position in the message



- An arriving packet goes into a *pool*
- The pool is sorted

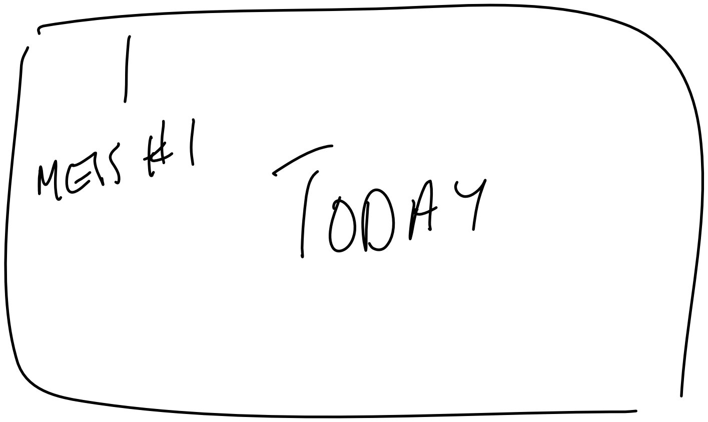


Example - A Packet Switched System



Example - A Packet Switched System

- What if several messages are sent to the same destination simultaneously?



Example - A Packet Switched System

- What if several messages are sent to the same destination simultaneously?
  - Add a second tag to indicate the message number.
- These properties define the *abstract data type* **pool.**
  - *Character strings of length one packet ending in a null*
  - *Message number*
  - *Packet number*

An Abstract Data Type

- Describes the form of the data
  - Component element types (e.g. characters)
  - A structure that relates the component element values (e.g. a linear arrangement)
  - A domain of allowable structures (e.g. from 0 to 80 characters in a packet)
- Defines how components may be accessed
  - A set of operations on the values in the domain

An Abstract Data Type

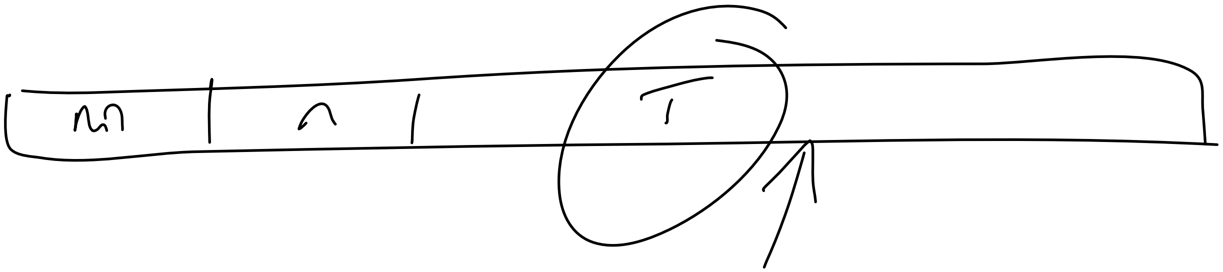
- The abstract data type pool structures data into
  - a key (the position in the message)
  - data (the packet itself)
- This structuring is necessary - a key always exists but may be implicit or, sometimes, the data may itself be the key.

A Data Structure

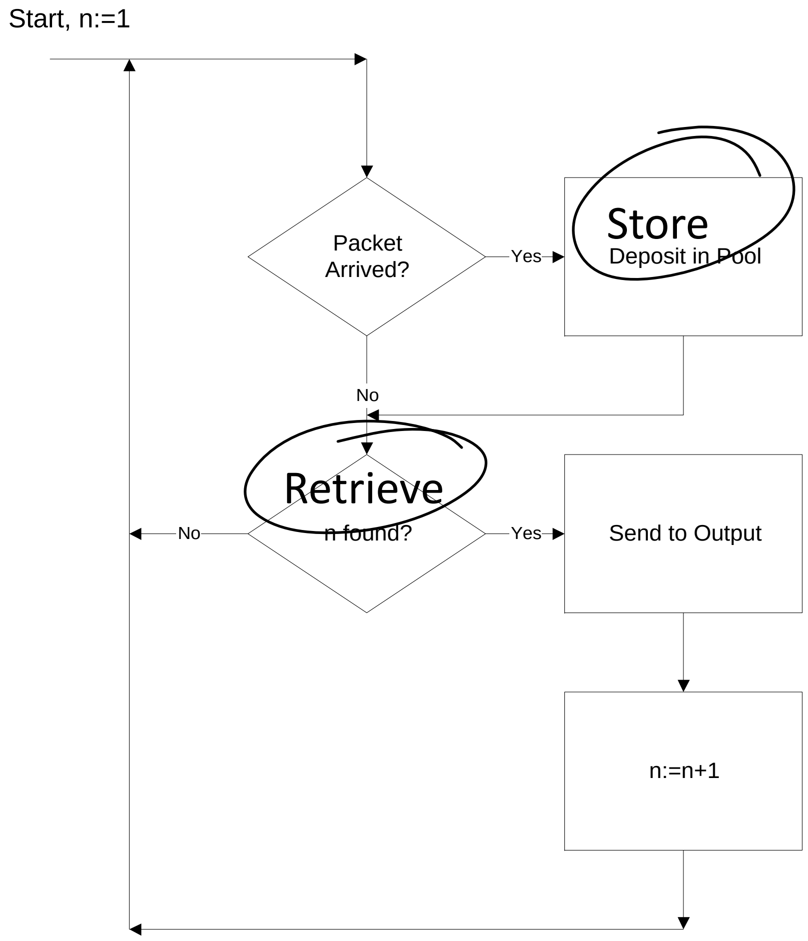
- The physical implementation of the data type.
- Maps the abstract data type onto the available environment.

Operations on the Data Type "Pool"

- Store (X)
  - Store a packet number and the associated text of packet X in the pool.
- Retrieve (m,n,T)
  - retrieves the text T associated with packet number n of message m from the pool, if it exists.



Operations on the Data Type “Pool”



The Data Structure

- The physical implementation of the data type
  - constructed from what is available in hardware
    - the elemental components are *bits*
    - from an abstract point of view we discuss *characters* and *numbers*
    - *bits* are grouped into larger structures
      - Bytes (8 bits)
      - Words (n bits)

The Data Structure

- Common values of n
  - 8, 16, 32, 64,…

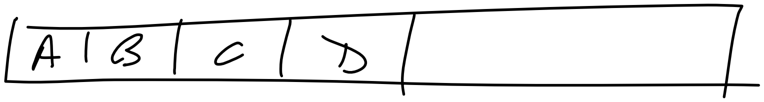
12
18
60

- Bits are too small to deal with..
  - Memory is constructed to work with individual bytes
  - Even though a working register may be several bytes wide, if each byte can be individually retrieved the machine is

Byte-addressable

Arrays and Vectors

- Ways of arranging collections of data of the same type



  - e.g. integers, real numbers, etc..
  - Each element is unique and located by a location (its *key*)
    - a_{ij}, b_k,..
  - The collection of elements is an *array*
  - If one index is used to locate an item (e.g. b_k), it is a *linear array* or *vector*