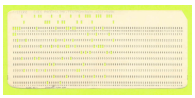
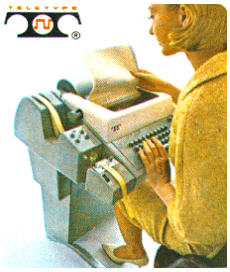
Moving Programming Languages from Batch to Interactive Systems Support

Reading #2:

"An Input-Output Model for Interactive Systems" by Mary Shaw, *Proceedings of the Conference on Human Factors in Computing Systems (CHI86)*, 1986, pp. 261-273.

Classical Batch I/O





GUI I/O

Interactive I/O

- Interactive I/O is different
  - Input is driven by events generated under the control of a human being rather than a program. Must synchronize timing. “Real-time control” problem.
  - Input is an interactive process requiring feedback. (Input is conventionally treated as a simple parsing problem.)
  - Output device is a continuous sensor or observer of the application software
  - Output device displays 2 or 3 D graphic material

Problem cont.

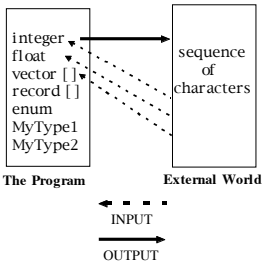
- Need a model that treats I/O as a problem of converting
  - between the data types of the program and
  - some suitable types for direct transmission to available I/O devices

Solution

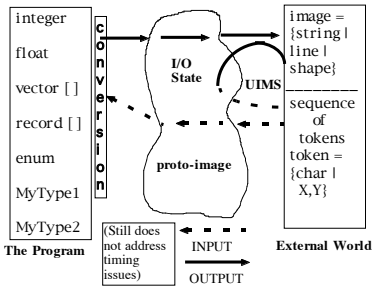
- Create separate I/O state (in addition to application program state)
- Separate I/O in application from I/O handled by the operating system

Classical Batch I/O

- Simple formatted input-output
- Consults format information for each value as it is converted



2-D Bit mapped Graphic Display, Keyboard & Pointing Device I/O



From Batch to Interactive I/O

A Formal Description

- Primitive I/O (model 1)
  In: seq-of-char --> P_i
  Out: P_i --> seq-of-char
  where P_i is a primitive type

  Example: INPUT X or PRINT X
- Add Formatting (model 2)
  In: seq-of-char x F_{in} --> P_i
  Out: P_i x F_{out} --> seq-of-char
  where P_i is a primitive type; F is formatting
  – Example: PASCAL read(x) or print(x:5:2)
  prints a floating point with 2 decimal places

From Batch to Interactive I/O
Model 3

- Add I/O state such as page numbers
  In: seq-of-char x F_{in} x IOState --> P_i x IOState
  Setup: IOState x . . . . --> IOState
  Out: P_i x F_{out} x IOState --> seq-of-char x IOState
  where P_i is a primitive type; F is formatting

  Example: FORTRAN
  100 FORMAT ('New page header', (/10(F10.2,2X)))
  200 WRITE (6,100) V

  prints a vector V beginning on a new page, 10 elements/line

From Batch to Interactive I/O
Model 5

- Add 2-D, interactive input
  In: seq-of-token x F_{in} x IOState --> P_i x IOState
  Setup: IOState x . . . . --> IOState
  QueryStyle: IOState x . . . . --> {F_{in} , F_{out} , F_{comp} }
  Compose: ProtoImage x F_{comp} x IOState --> Image x IOState
  Out: P_i x F_{out} x IOState --> Image x IOState
  where P_i is a primitive type; F is formatting

  Example: scrollbar widget

From Batch to Interactive I/O
Model 6

- Add user-defined types, 2D display, interactive input
  In: seq-of-token x F_{in} x IOState --> {P_i , T_i} x IOState
  Setup: IOState x . . . . --> IOState
  QueryStyle: IOState x . . . . --> {F_{in} , F_{out} , F_{comp} }
  Compose: ProtoImage x F_{comp} x IOState --> Image x IOState
  Out: {P_i , T_i} x F_{out} x IOState --> Image x IOState
  where P_i is a primitive type; F is formatting; T_i is user-defined type
  NOTE: Must have a mechanism for registering these definitions with I/O control so they can be appropriately invoked.
  Example: Java new class called "fancy scrollbar"

From Batch to Interactive I/O
Summary

- Add I/O state to program
  - Actual output of system influenced by information about the state or history of the input and output transactions
  - Example: page numbers
- Add sensitivity to event timing
  - Feedback from system must be synchronized with input from the user
  - Screen must be kept continuously updated if stored values change
  - Support asynchronous input from user
    - Processing of "terminate this process immediately" must not wait until the current process terminates on its own.

From Batch to Interactive I/O
Summary

- Must support graphics, video and sound as I/O types
  - Graphics plus text and other "natural data types"
  - Continuous image
  - Data changing with time, i.e. animation, video, sound
  - Allocation of space on the display
- Interactive input must provide feedback to user
- Allow user-definable data types to extend to I/O

Implications

- Decoupling of application from interface
- Strong linkage between display and program
  - Display reflects current program state at all times
- Freedom without license
  - Uniformity of interface style is an advantage, but users may want and need to tailor the interface to their own organization and style