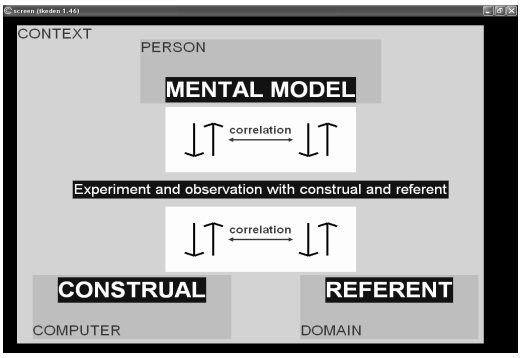
Empirical Modelling

- empirical = based on observation and experiment
- empirical = given in experience
- *modelling* because it is intended to support an activity that relies upon establishing a correlation between the experience offered by the computer and some external experience moment by moment ... and thus is (as if) carried out in a situation in which there is a referent

Model making as construing



Modelling state

- spreadsheet
- "now" + "being-in-the-moment"
- state-as-experienced
- phenomenology
- identity crisis: Bose-Einstein condensation
  http://www.colorado.edu/physics/2000/bec/#
- dependency: entanglement
  http://www.signandsight.com/features/614.html
- pragmatism: James and Dewey

Background and History

A *definitive notation* = a simple formal language in which to express definitions

A set of definitions is called a *definitive script*

Definitive notations different according to

types of the variables that appear on the LHS of definitions and **operators** that can be used in formulae on the RHS. These are termed the *underlying algebra* for the notation.

The definitive notation concept

Todd relational algebra query language ISBL
Brian & Geoff Wyvill's interactive graphics languages
spreadsheets
style definition in word processors

The term "definitive notation" first introduced by Beynon

"Modelling with Definitive Scripts" is fundamental to EM
[Rungrattanaubol's PhD Thesis: **A treatise on MWDS**]

Related developments

spreadsheets with visualisation mechanisms

spreadsheet-style environments for end-user programming (e.g. AgentSheets)

generalised spreadsheet principles in application-builders (e.g. ACE)

"object-linked embedding" in Windows

What does *definitive* mean?

definition has a technical meaning in this module
definitive means "definition-based"

"definitive" means
more than informal use of a programming technique.

Definitive notations are
a means to *represent state* by definitive scripts
and *how* scripts are interpreted is highly significant.

Significance of interpretation …

Miranda *can* be viewed as a definitive notation over an underlying algebra of functions and constructors
BUT this interpretation emphasises
program design as a state-based activity
rather than
declarative techniques for *program specification*.

[cf. 'admira' application and contrast with KRC]

Definitive notations

The tkeden interpreter uses many definitive notations

eden: scalars, strings, lists

DoNaLD: for 2-d line drawing

SCOUT: displays, windows, screen locations, attributes

EDDI: relational tables and operators

ARCA: edge-coloured digraphs in n-space

DoNaLD: a definitive notation for line-drawing

Donald = a definitive notation for 2-d line-drawing

underlying algebra has 6 primary data types:
integer, **real**, **bool**ean, **point**, **line**, and **shape**

A **shape** = a set of points and lines

A **point** is represented by a pair of scalar values {x,y}.

Defining shapes in DoNaLD

Two kinds of shape variable in DoNaLD:
these are declared as **shape** and **openshape**

An **openshape** variable S is defined componentwise
as a collection of points, lines and subshapes

Other mode of definition of shape in DoNaLD is
shape RSQ
RSQ=rotate(SQ)
- illustrated in definition of vehicle in VCCS model.

Agents and semantics

Archetypal use of MWDS: human-computer interaction
"single-agent modelling"

Variables in a definitive script represent
- the values that the user can observe
- the parameters that the user can manipulate
- the way that these are linked indivisibly in change
definitive script can model physical experiments

[cf the role of spreadsheets in describing and predicting]