Modelling with definitive scripts

Empirical Modelling as Construction

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), development tools (WPF)
“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, boolean, 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]
Modelling with definitive scripts

Modelling with different motivations

"Room as physical artefact with mass in time and space"

"Room as architectural drawing"

"Room as EM teaching artefact"

Script of definitions

Script with specific range of interactions

About Definitive Scripts

roomYung1989

roomviewerYung1991

room3dMacDonald1998

roomYung1989

About Definitive Scripts

room3dsasamiCarter1999

room3dsasamiCarter1999
Modelling with definitive scripts

---

**Observables, Dependency, Agency**

The observables, dependencies and agency that are topical relate to the situation and the way in which a script is being interpreted.

In the architectural drawing, don’t observe time.

In the physical room, observe mass, time, force.

In teaching EM, we observe the screen display itself and seek to interpret “absurd” definitions.

---

**Observables**

Observables are entities whose identity is established through experience whose current status can be reliably captured by experiment.

*Can be physical, scientific, private, abstract, socially arbitrated, procedurally defined etc.*

---

**Dependency and Agency**

An *agent* is an observable (typically composed of a family of co-existing observables) that is construed to be responsible for changes to the current status of observables.

A *dependency* is a relationship between observables that - in the view of a state-changing agent - expresses how changes to observables are indivisibly linked in change.

---

**Single Agent modelling**

In the primary and most primitive form of Empirical Modelling, the modeller is the only state-changing agent – though they may act in the role of different agents: e.g. room user or designer, architect, Empirical Modelling lecturer.

The dependencies between observables are then those that are experienced by the modeller acting in the situation: they express the way in which changes to observables are connected.

---

**Negotiated and evolving interpretations**

The situation surrounding the interpretation of a script is never completely closed or well-specified. The modeller always has to exercise discretion to achieve a degree of closure. Situations can blend.

Definitions stabilise as meanings are negotiated.

Stable definitions reflect established experience.

Skills and insights can give rise to new definitions.

---

**Illustrative examples**

Definitions stabilise as meanings are negotiated.

*The model of the desk drawer gets improved.*

Stable definitions reflect established experience.

*The door location and mechanism gets fixed.*

Skills and insights can give rise to new definitions.

*We connect the door opening with the light coming on, or learn to use a touch-sensitive switch.*