

Model making as construing

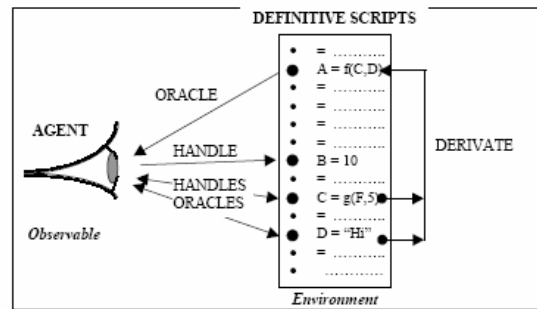
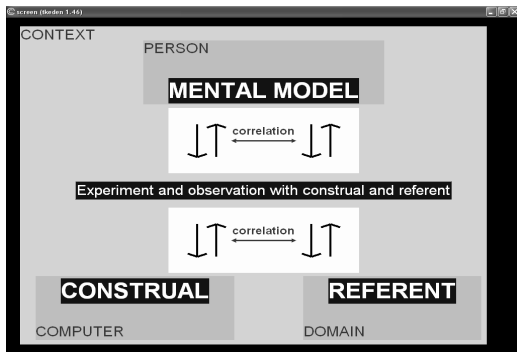


Figure 2-18: Definitive script as observer's model of state ('one-agent' modelling)

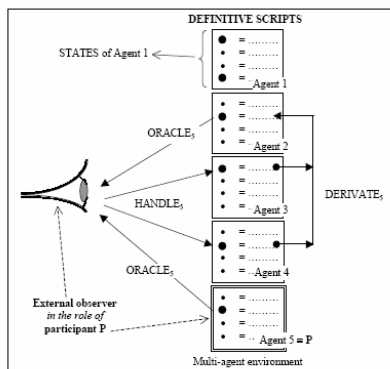


Figure 2-19: Definitive script as observer's model of state ('multi-agent' modelling)

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

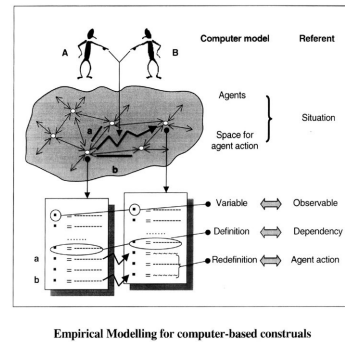
Agents

Agents are responsible for state-changes:
 meta-agents: e.g. the model builder
 agents determining model behaviour
 Observables mediate agent actions/interactions

Use 'LSD notation' to specify perceptions and protocol (= *privileges*) of agents

Examples

meta-agent: software developer; architect
 agent: users, devices; room user, door



Virtues of a definitive script

- represents view (cf spreadsheet)
 - variables correspond to observables
 - hides invisible activity
 - can represent indivisibility in action
- ... *when interpreted with agent protocol*
- allows experimental basis of knowledge
 - reflects different status of parameters
- ... *supports open-ended incremental and distributed development*

Roles for modelling with definitive scripts

- Definitive scripts support artefacts that help developers*
- to identify reliable interactions with their environment
 - to recognise when there is a working understanding
 - enable complex co-operative behaviour
 - to construe complex system behaviour as agent interaction
- ... *formal approaches neglect the empirical basis for knowledge of reliable systems that embraces activities of all these kinds*

EM vs traditional modelling

conflate concerns	separate concerns
represent via metaphor	represent symbolically
support ambiguity	expect/impose precision
encourage customisation	promote standardisation
expose empirical roots	hide empirical foundation
are shaped by construal	discard explanation

... **Key concept: Modelling based on 'definitive scripts'**

EM for Systems development

'Concurrent system in the mind of the external observer'

- identifying an objective perspective
- circumscribing agency
- identifying reliable generic patterns of interaction

- Concurrent engineering design task ...

EM as pre-system development

Making the transition from
uncircumscribed ill-conditioned, loosely
regulated interactions

to

circumscribed precisely prescribed well-
regulated reliable behaviours

Illustration from railway history ...

... Tamworth 1870 accident