An Agent-oriented Framework for Concurrent Engineering

Valery Adzhiev*

Meurig Beynon†

Alan Cartwright‡

Simon Yung†

* *Flight Research Institute*

Zhukovsky, Russia

† *Dept. of Computer Science*

‡ *Dept. of Engineering*

University of Warwick
Coventry CV4 7AL

Acknowledgements to EPSRC, Royal Society

Content of talk

- Concurrent Engineering (CE): Motivating Issues

  Knowledge representation in CE
  Conventional approaches

... need new modelling paradigm
can new modelling techniques address issues? ...

- The Abstract Definitive Machine (ADM)

  agent-oriented modelling
  definitive representations of state

- Virtues of the ADM

  Coordinating agent interaction in the ADM
  Case study: railway system modelling

... our new approach has considerable promise
How can we adapt it for concurrent engineering?

- A Computational Model of the CE process

  The virtual prototype
  The hierarchy of design agents
  Design frames
  Time: schedules, milestones, history

Knowledge representation

– issues for Concurrent Engineering:

Need to assist design team to

- represent many alternative views;
- distinguish and synthesise knowledge of fundamentally different kinds;
- deal effectively with concurrency, inconsistency and conflict;
- record human decision-making and negotiation;
- express the concept of a consensus view.

Elaboration of each issue ...

- *represent many alternative views;*

  many modes of observation of a design product, needing diverse representations

- *distinguish and synthesise knowledge of fundamentally different kinds;*

  ... different source, nature, status:

  source:

  physical law / superior / customer

  nature:

  theoretical /empirical / open to expt

  status:

  amenable to change / private / shared

... characteristics shape design interactions
e.g. in communication, experiment, research

Elaboration of each issue (cont.) ...

- *deal effectively with concurrency, inconsistency and conflict;*

  design is explored and evaluated by different criteria, but agreement may be essential for furthering the design

- *record human decision-making and negotiation;*

  the basis for judgement may be aesthetic

  no objective criteria in conflict resolution

- *express the concept of a consensus view:*

  Concurrent Engineering

  ⇒ **integrity** of the cooperative activity

  what is current status of the design?

  how do we represent the consensus view?

Conventional ways of computer modelling

[typically]

need comprehensive models of state & behaviour

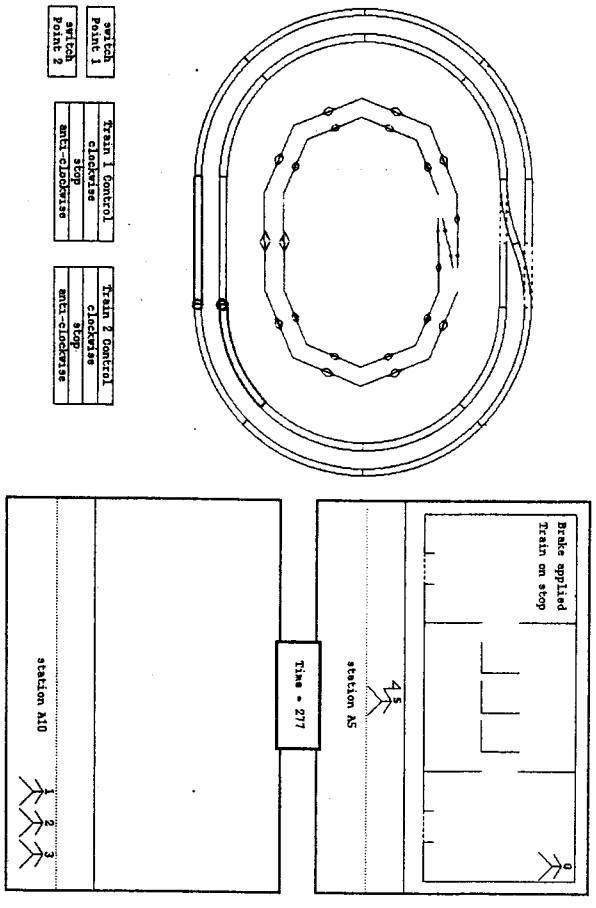
presume consistency and perfect knowledge

are not view-oriented

are aimed at *automating* the design process

rather than *supporting the human designers*

... *motivates alternative modelling paradigm*



Definitive scripts

Use scripts of definitions to represent state

Use redefinition to specify change of state

Scripts make use of definitive notations:

SCOUT - window layout

DoNaLD - line drawing

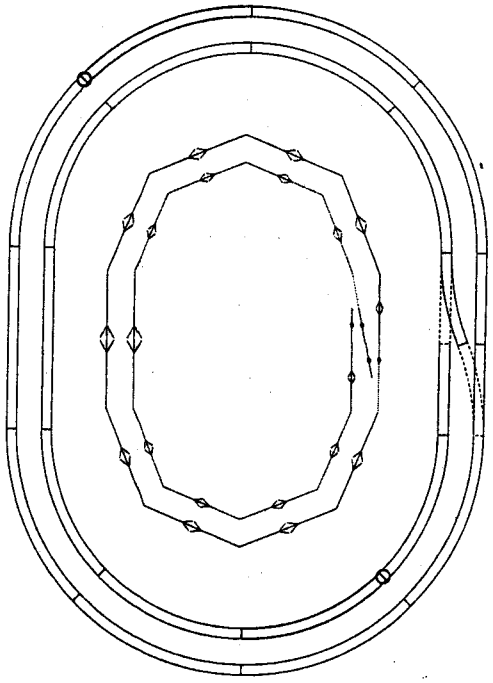
ARCA - combinatorial graphs

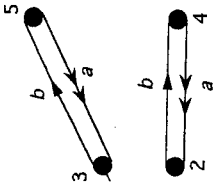
ARCA model of set of points as an example

Virtues of the **definitive script**

- represents view (cf spreadsheet)
- variables ↔ observables
- hides invisible activity

  / represents indivisibility






```
# points have 5 vertices and 3 colours
mode Point1 = 'abc'-diag 5

Point1Stts = 1 # status of the points, 1 = closing the loop
# defining the edges
a_Point1{1} = 1; b_Point1{1} = 1; c_Point1{1} = if (Point1Stts == 1) 2 else 3
a_Point1{2} = 2; b_Point1{2} = 4; c_Point1{2} = if (Point1Stts == 1) 1 else 2
a_Point1{3} = 3; b_Point1{3} = 5; c_Point1{3} = if (Point1Stts == 1) 3 else 1
a_Point1{4} = 2; b_Point1{4} = 4; c_Point1{4} = 4
a_Point1{5} = 3; b_Point1{5} = 5; c_Point1{5} = 5

Scale = 100   # scaling factor for locating vertices
Point1!1 = A!1   # defining locations of the vertices of Point1 in
Point1!2 = A!1 - [Scale / 2, 0, 0] # terms of the location of the first vertex of
Point1!3 = A!1 - [Scale / 2, Scale / 10, 0] # A, the graph representing the outer loop
Point1!4 = A!1 - [Scale, 0, 0]
Point1!5 = A!1 - [Scale, Scale / 5, 0]
```

A definitive script for the connectivity of a set of points

Agents

agents are responsible for state-changes

have meta-agents: such as the model builder

also have agents to define model behaviour

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

e.g. stationmaster

Examples

meta-agent: Track Layout Designer

agent: stationmaster, guard

Virtues of the definitive script + **protocol**

- experimental basis of knowledge
- reflects different status of parameters

The Abstract Definitive Machine

Definitive store D + Action store A

entity = set of definitions + set of actions

agent-oriented interpretation

LSD specifications for agents ↔ entities

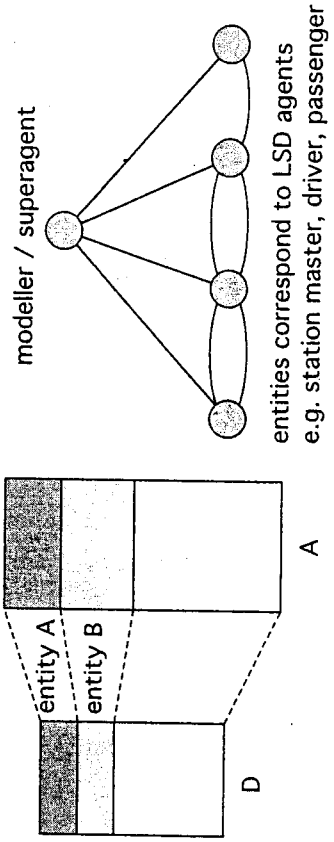
animation of agents (as in entities) by explicit scheduling and synchronisation

observation of the model as dictated by the modeller (= superuser)

Virtues of the execution model

- comprehensible states
- scope for intervention and redirection
- conflict detection and resolution

Evident in superuser-driven mode of execution



The ADM Machine Model

Prospects for addressing each issue ...

- *represent many alternative views;*

  diverse representations e.g. track layout, connectivity graph via definitive notations DoNaLD, SCOUT, ARCA etc

  incorporate behavioural models:

  simulate motion of train

  simulate arrival-departure protocol

  supply operator interface

- *distinguish and synthesise knowledge of fundamentally different kinds;*

  * track layout as conceived by designer
  * connectivity determined by layout
  * points setting as specified by operator
  * stationmaster interprets door open
  * content of model: trains crash at points
  * form of display & interaction interface

  ... *explanatory modelling*

Prospects for addressing issues (cont.)

- *deal effectively with concurrency, inconsistency and conflict;*

  concurrent action ↔ parallel definition

  one passenger *opens* , another *shuts* door

  track layout is redesigned – new topology

- *record human decision-making and negotiation;*

  assess aesthetic qualities of layout

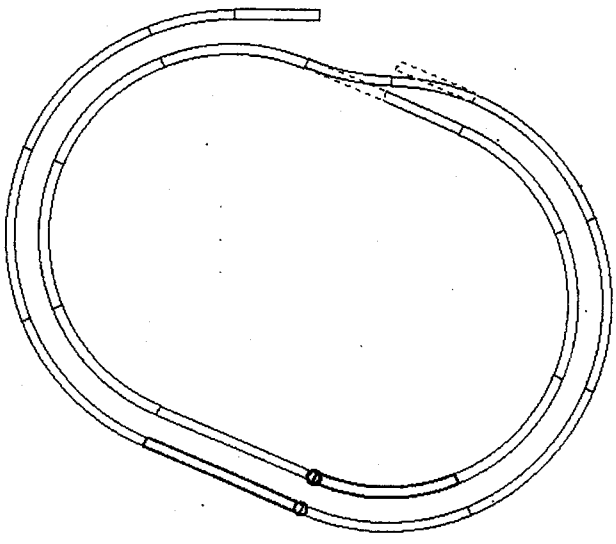
  no objective criteria in conflict resolution
  e.g. will door be shut or open?
  alter layout / connectivity graph?

- *express the concept of a consensus view:*

  Reconciling the track layout with the connectivity graph involves coordination

  Interpreting the physical train location from abstract location needs convention

.... *motivates coordination of views*



The Concurrent Engineering Design Process

Hierarchy of design agents

General principle

Two perspectives for a design agent
upward view: negotiation with authority
downward view: management of subagents

Top of the hierarchy

Customer (C) & Principal Manager (PM)

Customer

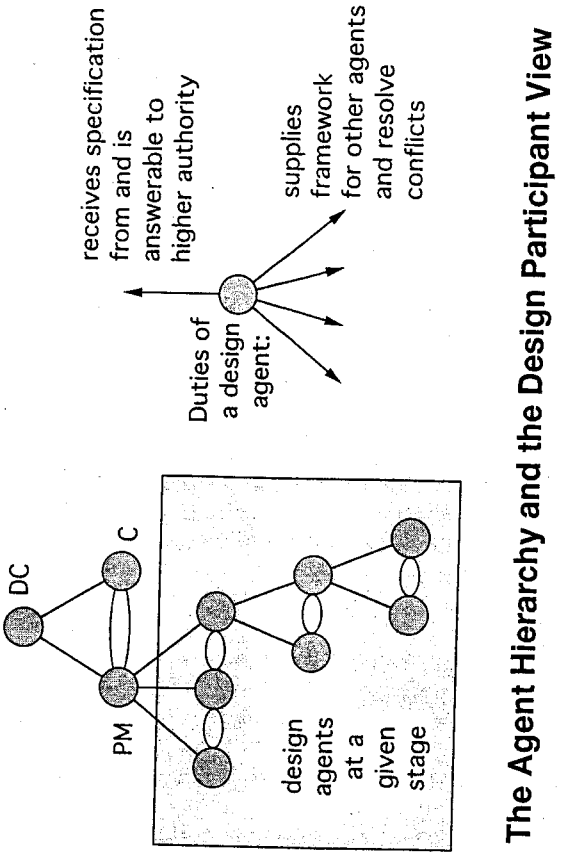
role: supplies requirements specification
[which may be developed / transformed dynamically during the design process]

Principal Manager

role: represents the consensus view
negotiates over context for design, and customer requirement

Hierarchy of agents below principal manager

Upper levels: Coordination and Management
Lower levels: Experimentation and Detail



The Agent Hierarchy and the Design Participant View

The Evolution of the Virtual Prototype

Virtual Prototype: consensus of design team on current status of product being developed

Have a *design frame* ↔ each agent

design frame of an agent
↔ current status of design as seen by agent

design frame of PM
↔ current consensus about design

design frames are developed in stages

development involves selection of parameters appropriate to current stage

Ingredients in developing frames: Expectation, Elaboration, Alteration, Novelty and Learning [Minsky]

Lower level activity ≡ situated modelling
cf. track layout design
Higher level activity ≡ ADM programming
cf. overall railway model management

Concurrent Engineering: Computational Model

Within the hierarchy of design agents

Each design participant

- is given parameters and constraints, patterns of interaction, privileges, goals by design coordinator at higher-level

  interface for experiment and interaction may also be supplied

- conflicts between view and those of peer agents are arbitrated at higher level

- supplies framework for (or "programs") agents lower in the hierarchy, specifying:
  which agents are active
  how they interact
  when the agents act
  synchronisation / scheduling of tasks
  rules to resolve interagent conflicts
  storage and retrieval strategies

... cf superuser programming in the ADM