Definitive Methods for Programming and Parallelism

How to approach parallelism?

1. Adaptation to existing parallel machines

"Powerful" parallel architectures exist, but who can program them?

Hardware technology outstrips software technology?

Will future developments be hardware or software driven?

Definitive approach:

So far primarily *model the application*, but could *model the machine*

[no clear distinction between modelling devices and system context]

Examples of this in a small way:
cf tank.e / tank.s, SCOUT for OpenWindows, Peco Rail, Systolic Arrays

SAND: systolic array definitive notation devised by Sockett in 1991

How about the Parsytec??

2. Translation techniques from a definitive specification

Interface with existing machines via an appropriate "high-level" language

Generate efficient code in OCCAM from high-level specification

Is this really just strategy 1 in disguise?

Definitive approach:

Use definitive methods as a requirements capture agent-oriented rather than object-oriented analysis technique.

Ingredients:

ATC moves the plane vs low-level divisible protocol
Could treat as preliminary phase in OOD for distributed machine

definitive state-transition models to Petri nets, CCS (Paul Ness)
[small-scale case studies, as explored on Concurrency WorkBench]

Possibly related projects

translation from EDEN to procedural language
[most successful effort yet is to procedural EDEN]

translation from ADM to procedural language

admira: a definitive miranda

3. **Architecture software-driven by (generalised/adapted) ADM model**

Presume definitive script good for parallelism in following respects:

- order of definitions immaterial
- dependency is explicit
- parallel redefinition is conceptually simple

How can we exploit this? (if at all)

[cf efficient parallel implementation of spreadsheet challenge]

VN bottleneck connected with need when updating procedural vars to update all the dependent variables ?? ... how about having background processing in memory maintaining updates, shipping only key actions from processor to memory? Obligatory update on call-by-need. Possibility of clever maintenance of update by virtual calls for values.

Possibly related project

definitive interpretation of definitive interpreters