Seminars for the S4 module 1993

M4 The Definitive Programming project reviewed

Orientation on concurrent systems software cf S1, S2, S3
Emphasis in S4 on **reactive systems requirements modelling**
    programming / software development / modelling

Issues for concurrency and parallelism
Baldwin:
    software technology for parallel programming is in bad shape

The broader perspective for the definitive programming project
The 4 areas:
    Programming Paradigms
    Software Engineering
    AI and Applications
    Foundations
Quotations from Backus, Kent, Harel, Cantwell-Smith

The Programming Paradigms perspective

Backus: criteria for evaluating programming languages
Problems of representing state central to parallelism

Agent-oriented modelling
    for requirements specification in reactive systems
From comprehensive description to feasible prescription

Modelling the interface using definitive scripts
    modelling the door wrt user and architect protocols

Agent-oriented modelling over definitive representations of state
0-agent, 1-agent, multi-agent models in definitive programming

Overview of the project: topics, personnel, development of ideas

T4 Programming as Modelling: the context

Brooks: No Silver Bullet

The four areas:
- the AI and Software Development perspectives

Cantwell-Smith: form vs content – the lessons of logic
- Context-dependence
- Interaction between form and content
- More discriminate modelling

Harel: Biting the Silver Bullet
Issue:
- is there a fundamental distinction between
- 1-person programming and reactive systems engineering?

Relevance of form-content controversy to Harel's position
Motivation for linking first and second factors in programming

Object-oriented programming as a case-study

W4 Programming as Modelling revisited

Development of the programming as modelling concept
Modelling agent interaction as key to Programming as Modelling

Models of physical phenomena: classical physics vs modern physics
- mechanical analogies in classical physics
- object-oriented models in modern CS

Significance of an observation-oriented approach
The role of observation and experiment

From Observation and Experiment to Definitive Scripts
The historically primary uses of *variable* and *function*

Bertrand Russell and Mevdevev

Virtues of definitive scripts
- dealing with first and second factors
- form and content: the filing cabinet vs digit example
- context-dependence
- more prescriptive modelling
- useful role in reactive systems specification

Th4 From Modelling to Programming

Summary of our orientation towards programming as modelling

Computer models in the reactive system development process

The VCCS is not a program

The Paradox of Experiment
know exactly what (I expect) will happen
have no idea (within preconceived context) what will happen

Interpretation of same experiment on two sides of an Act of Faith

Programming as
conventions for interpretation
+
manipulation and observation of a reliable state-changing device

1-person programming from this perspective

Computational agents in the reactive system context
What kind of an object can be used to carry out computation?
Motivating example: the GCD folk-dance routine

Summary

distinction between sequential and concurrent programming
and 1-person and reactive systems programming

the act of faith as the bridge between practice and theory

physical experiment dictates
functional dependencies -> definitive script
independent centres of change -> agents

=> new principled approach to programming
based on multi-agent models of physical systems
principled use of functional abstraction