CS405 Intro to EM

Theme 3: Software and system development

Further motivation and orientation for Empirical Modelling …

Critical issues in computer science …

1. **The "software crisis"**
   No silver bullet (Fred Brooks)
   Biting the silver bullet (David Harel)
2. **The logicist debate**
   A critique of pure reason (Drew McDermott)
   Two lessons of logic (Brian Cantwell-Smith)
3. **The current status of database theory**
   DB systems or DB theory – or "Why don't you teach ORACLE" (Mike Ridley)

The "software crisis"

Fred Brooks: No Silver Bullet … (1986/1995)

… reviews every aspect of technical and conceptual progress in mastering software development, and concludes that none of these advances is addressing the essence of the problem …

Biting the silver bullet

Response by David Harel (1986)

- one-person programming vs reactive systems
- need for visualisation for cognitive support e.g. statecharts, message sequence charts
- Play-In scenarios

Biting the silver bullet 2

Amir Pnueli: Reactive systems are systems whose role is to maintain an ongoing interaction with their environment rather than produce some final value upon termination. Typical examples of reactive systems: air traffic control system, programs controlling mechanical devices such as a train, a plane, or ongoing processes such as a nuclear reactor.

The logicist debate

McDermott – a Critique of Pure Reason

Celebrated renunciation of faith in logic as basis for AI

Responses collected in Comput. Intell. Vol 3 1987

Brian Cantwell Smith – a non-logicist stance …

The two lessons of logic

Lesson one: the irreducibility of content to form

content relations aren't computed: how symbols 'reach out and touch someone' - almost a total mystery

Lesson two: a single theoretical stance

The two factors (form and content) are relatively independent, but have to be ultimately related: a single, unified theory must provide an account of both factors

Formal logic tenets *not* for AI (1)

1. Use can be ignored
2. Locally the two factors can be treated independently, even though they must be globally related
3. Language and modelling should be treated completely differently (as in "promiscuous" modelling: can substitute model of X for X with theoretical abandon)

Formal logic tenets *not* for AI (2)

1. *Use can be ignored*

   cf. natural language – context dependence, resolving 'now', pronouns etc

2. *Locally the the two factors can be treated independently, even though must be globally related*

   'if the first factor could be cleft from the second factor, would make sense to write things down first and build programs second' (McD) – not like thought

Formal logic tenets *not* for AI (3)

3. Language and modelling should be treated completely differently

Whole new theories of representation and correspondence will be required:

- explaining computational practice
- promiscuous modelling "pernicious where fine grained questions are concerned"
- representations in current computational systems range continuously from linguistic to virtually iconic

The theory of databases perspective on representing state …

The CS perspective on database theory …

M J Ridley, University of Bradford (TLAD WS 2003)
DB systems or DB theory – or "Why don't you teach ORACLE"

"The modern age of databases may [NB! - WMB] have started with relational databases which despite their ubiquitous nature we should remember were looked on as of only academic interest at one time. What surely marks this era [the early 1970s] out is the sound theoretical basis of that database model and the progress that was made because of this.

… But what age are we in now?"

Perspectives from academia 2

M J Ridley, University of Bradford (TLAD WS 2003)
DB systems or DB theory – or "Why don't you teach ORACLE"

"… most people would agree [that] … we need a theoretical framework in which to talk about DBs if the result is to be productive … The questions that seem to me to need answering are: What is that framework? …

… Is it the simple relational model? How important is that framework? (In practical, educational terms how much of the syllabus is this?) Are there competing theoretical frameworks that need considering?"

Perspectives from academia 3

M J Ridley, University of Bradford (TLAD WS 2003)
DB systems or DB theory – or "Why don't you teach ORACLE"

"*… most people would agree [that] … we need a theoretical framework in which to talk about DBs if the result is to be productive …*

But is there still a firm theoretical foundation for what we teach? Does that matter? Are we in danger of training students in the features of a particular system (and a convenient web interface for it) rather than educating them about DBs in the broader sense? …"

Perspectives from academia 4

M J Ridley, University of Bradford (TLAD WS 2003)
DB systems or DB theory – or "Why don't you teach ORACLE"

"What is the relationship of Database teaching to other aspects of the curriculum such as the study of data structures and object-orientation? And are there theories of these subjects? … "

"… is there are more general object theory that informs programming and software development generally. If so can relational theory be located as a subset of that mainstream or does database stand out as an exception?"

Perspectives from academia 5

M J Ridley, University of Bradford (TLAD WS 2003)
DB systems or DB theory – or "Why don't you teach ORACLE"

Ridley's own answer to these questions:

"… we are in danger of losing sight of firm theoretical ground especially as we are under pressure to include yet more additional facilities of DBMSs … more so as object systems have failed to achieve commercial success commensurate with their early promise and the pressure of the web has shifted the focus of much database work."

C.J.Date: Why relational?
from *Relational Database Writings 1985-1989*

Purpose of the paper ...

... a succint and reasonably comprehensive summary of the main advantages of the relational approach

… concerned with **technical** not business advantages

… to evaluate relational models in DBs fully we must also consider the most fundamental issues

Why relational? - [Chris Date] 1

Perceived advantages of relational DBs:
simple data structure
simple operators
no frivolous distinctions
SQL support
the view mechanism
sound theoretical base
small number of concepts
the dual-mode principle
physical data independence
logical data independence

Why relational? 2

Perceived advantages of relational DBs (cont.):
ease of application development
dynamic data definition
ease of installation and ease of operations
simplified database design
integrated dictionary
distributed database support
performance
extendability

Issues raised in database manifestos in 80s

Classical applications for databases presume

uniformity: many data items of similar format

record orientation: fixed length records are the basic data item

fields are atomic: assume "1NF "

short transactions: typically fraction of a second, no human interaction

static conceptual schemes: infrequent change of DB scheme, mode of change is restricted

Issues for database development

Modern database demands

enormous volumes of data

high-performance e.g. for multi-media, real-time

support for metaphor e.g. visual image not table

concurrent access, distributed data

closer integration of data access and programming

support for modern data abstractions:

objects, inheritance, aggregation

applicability to design environment needs: incremental intensional change

Issues that emerged in the 80s ...

In context of new applications:

CAD, CASE, multi-media, GIS, expert DBs

had the development of new features:

rule-based integrity, triggers, knowledge emphasis

complex objects

complex data structured hierarchically

⇒ OODBs + nested relational models

behavioural data

interdependent deletion of tuples

store behavioural info in DB

⇒ methods and rule-based approaches

Issues that emerged in the 80s

behavioural data

interdependent deletion of tuples

store behavioural information in DB

⇒ methods and rule-based approaches

meta-knowledge

application data obeys complex rules

"a/c's pay interest only if balance > X"

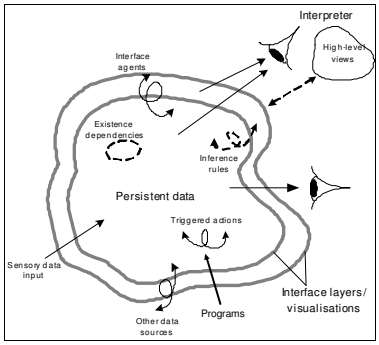
⇒ rule-based approaches

long transactions

need for undo, as in exploratory design

conflicts amongst transactions

Modern context for general data modelling



Issues for database development

How to avoid "back to the future"?

need theoretical foundation

need qualities of declarative query

need principles to handle abstraction at many levels: data independence

need to support interaction of agents at high-levels of abstraction

need to retain/enhance the form-content relationships that relational DB design theory introduces

Connections with Empirical Modelling 1

- reactive systems
  - modelling agency and interaction
- visualisation of conceptual state
- non-logical representation
  - new semantic possibilities for linking form and content afforded by dependency
  - coherent account of form-content relation

Connections with Empirical Modelling 2

- non-logical representation
  - context-dependence
  - linking of form and content
  - not promiscuous modelling
- strong link with RDBs + views (cf. ISBL)
  - generalising functional dependency
  - potential for enriched data representation

Connections with Empirical Modelling 3

Most critical issue of all …

- prospects for unifying activities that are empirical in nature and involve pre-theory experiment with processes that are informed by theory and amenable to abstract representation using logic … optimisation, inference, formal definition

Connections with Empirical Modelling 4

in effect … unifying theory and practice

… hence our concern for a potential "new foundation for computing": EMpaper 027: *The Interpretation of States*

… see also 028: *New Paths for Programming in Theory and Practice*