Empirical Modelling in support of constructionist learning:

A Case Study from Relational Database Theory

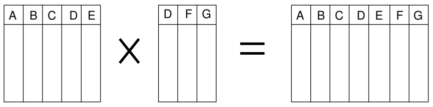
Meurig Beynon and Antony Harfield
Department of Computer Science
http://www.warwick.ac.uk/go/em

Theme of the talk

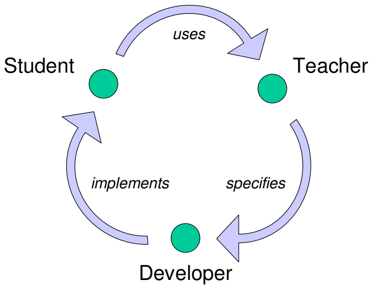
- Developing educational software should be closely linked to learning within the subject domain -> constructionism.
- Traditional approaches to computer programming are not good at this.
- Suggest that Empirical Modelling addresses this by model-building that is based on identifying patterns of observables, dependencies and agency that are significant in the domain.
- Illustrate this with reference to a specific case study.

A case study in relational databases

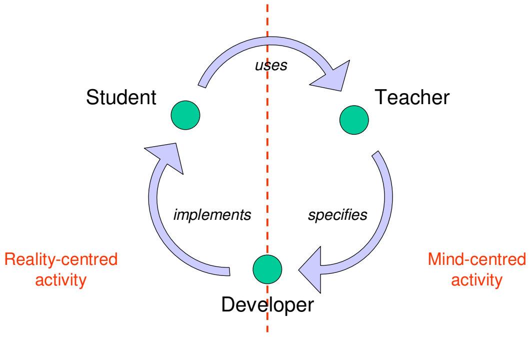
- An educational artefact for learning about ‘lossless join decomposition’ (from an undergraduate course in Computer Science).
- Enable the student to test relational decompositions experimentally, and develop a method for identifying lossless joins.



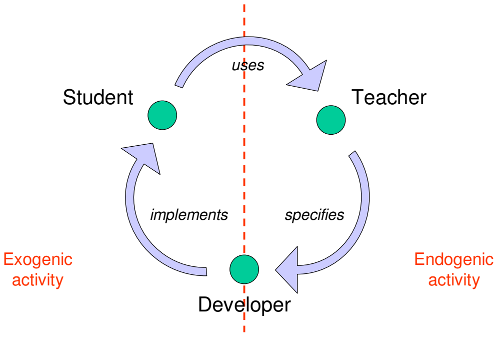
Developing educational software

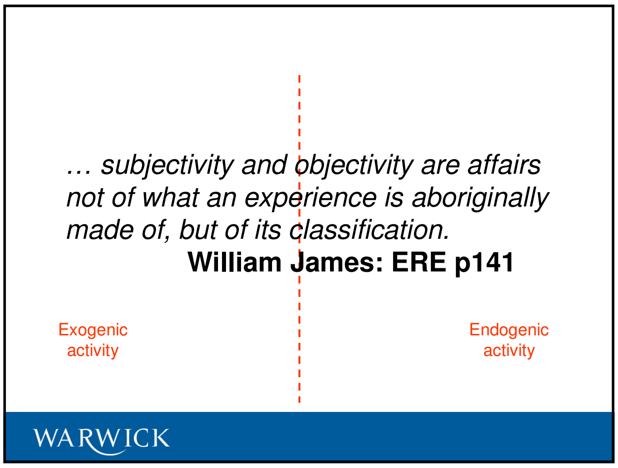


Developing educational software



Developing educational software



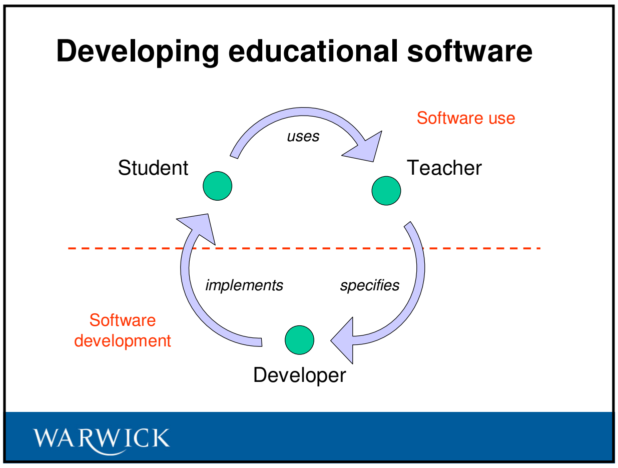


… subjectivity and objectivity are affairs not of what an experience is aboriginally made of, but of its classification.

William James: ERE p141

Exogenic activity

Endogenic activity



Developing educational software

Software use

Student

uses

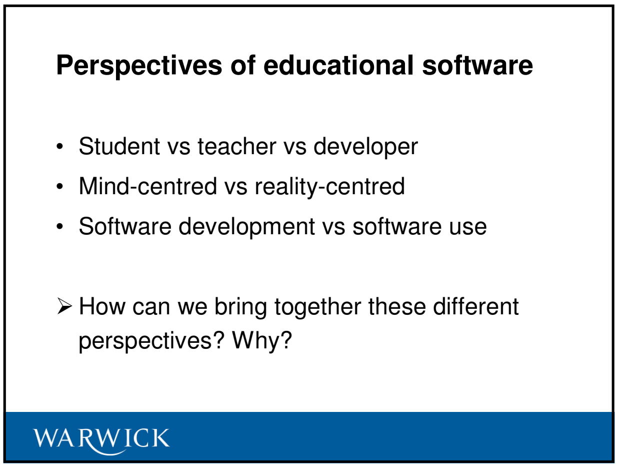
Teacher

implements

specifies

Software development

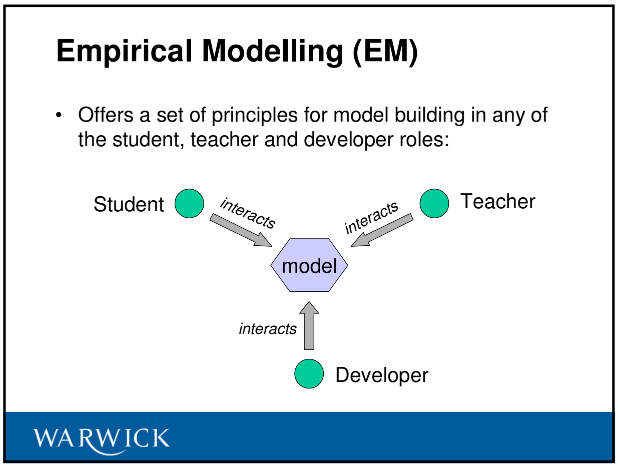
Developer



Perspectives of educational software

- Student vs teacher vs developer
- Mind-centred vs reality-centred
- Software development vs software use

➢ How can we bring together these different perspectives? Why?



Empirical Modelling (EM)

- Offers a set of principles for model building in any of the student, teacher and developer roles:

Student

interacts

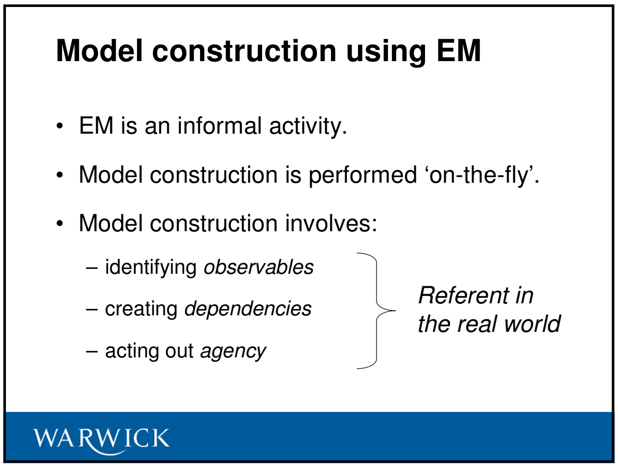
interacts

Teacher

model

interacts

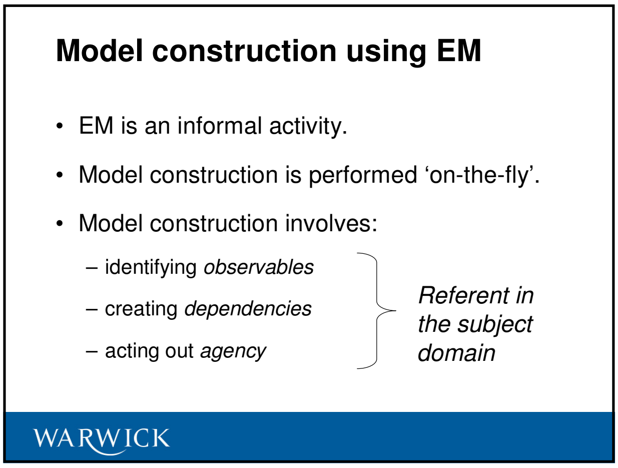
Developer



Model construction using EM

- EM is an informal activity.
- Model construction is performed 'on-the-fly'.
- Model construction involves:
  - – identifying *observables*
  - – creating *dependencies*
  - – acting out *agency*

Referent in the real world



Model construction using EM

- EM is an informal activity.
- Model construction is performed 'on-the-fly'.
- Model construction involves:
  - – identifying *observables*
  - – creating *dependencies*
  - – acting out *agency*

Referent in the subject domain

EM and spreadsheets

- Spreadsheets are useful tools for educational software (Baker and Sugden).
- Dependency is a key concept.
- Model construction takes place in the subject domain.
- Less difference between student, teacher and developer.

➢ EM tools are like generalised spreadsheets

EM and spreadsheets

- Spreadsheets are useful tools for educational software (Baker and Sugden).
- Dependency is a key concept.
- Model construction takes place with close attention to the subject domain.
- Less sharp distinction between the student, teacher and developer perspectives.

➢ EM tools are like generalised spreadsheets

Lossless Join Decomposition

Let R be a relation scheme, ρ a decomposition of R and F a set of functional dependencies of R. Suppose that the sub-schemes in ρ are $\{R_1, R_2, \ldots, R_k\}$.

ρ has lossless join if every extensional part r for R that satisfies F is such that $r = \Pi_1(r) \bowtie \Pi_2(r) \bowtie \ldots \bowtie \Pi_k(r)$, where $\Pi_i(r)$ denotes the projection of r onto R_i.

Informally: r is the natural join of its projections onto the sub-schemes $R_1, R_2, \ldots, R_k$.

Illustrating lossless join

Consider the relation

SUPPLIERS (NAME, CITY, AGENT, ITEM, PRICE)

Semantics: *Each supplier is based in a city, and the enterprise responsible for setting up the database has an agent for each city.*

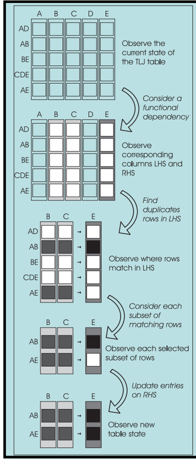
The set F of functional dependencies is generated by:

$S \rightarrow C$, $C \rightarrow A$, $S\ I \rightarrow P$

... each supplier sited in one city

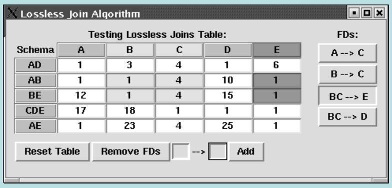
... each city has one agent serving it

... each supplier sells each given item at fixed price



A
B
C
D
E
AD
AB
BE
CDE
AE
Observe the current state of the TLJ table
Consider a functional dependency
Observe corresponding columns LHS and RHS
Find duplicates rows in LHS
Observe where rows match in LHS
Consider each subset of matching rows
Observe each selected subset of rows
Update entries on RHS
Observe new table state

An observation-oriented model of the testing lossless join algorithm (constructed using tkeden)



Testing Lossless Joins Table:

Schema	A	B	C	D	E
AD	1	3	4	1	6
AB	1	1	4	10	1
BE	12	1	4	15	1
CDE	17	18	1	1	1
AE	1	23	4	25	1

```
project_table_LHS_FD is project(current_table, makestrlist(FDs[current_FD][1]));
project_table_RHS_FD is project(current_table, [FDs[current_FD][2]]);
pattern_duplicate_rows is index_duplicated(tail(project_table_LHS_FD));
newcol is transformcol(makelistcol(project_table_RHS_FD), pattern_duplicate_rows);
newtable is apply_current_FD_current_table(current_table, newcol);
```

Listing 1: Observables and dependencies in the TLJ construal

Lossless and lossy decompositions

$SCAIP = SIP \bowtie SCA = SIP \bowtie SC \bowtie CA$ *lossless*

$SCA \subseteq SA \bowtie CA$ and $SCA \neq SA \bowtie CA$ *lossy*

… have possibility that Fred is agent for Hull and York, and that PVC is a supplier based in Hull. Then:

(Hull, Fred) * (PVC, Fred) = (PVC, Hull, Fred)

(York, Fred) * (PVC, Fred) = (PVC, York, Fred)

The second join is not in the relation SCA.

So this decomposition is not lossless join.

Testing Lossless Join (step 1)

1. Construct a table
   with n columns (corresponding to attributes)
   with k rows (corresponding to sub-schemes)

Initialise the table at row i column j
by entering α_j if attribute A_j appears in R_i
and by entering β_{ij} otherwise

NB α's represent joinable tuples, padded out to R by β's

Testing Lossless Join (step 2)

2. Repeatedly modify the table to take account of all dependencies until no further updates occur

   i.e. if X → Y and two rows agree on all the attributes in X then modify them so that they also agree on all attributes in Y. Explicitly, change attributes in Y thus:

   - if one symbol is an α_i make the other an α_i
   - if both symbols are of form β_{*j} make both β_{ij} or $\beta_{i'j}$ arbitrarily.

On termination declare lossless join if and only if one of the rows is $\alpha_1\alpha_2 \ldots \alpha_n$.

Testing Lossless Join – an illustrative example

Verify the decomposition SCAIP = SIP ⋈ SC ⋈ CA is a lossless join ....

Initial table

$$\begin{array}{llllll} & S & C & A & I & P \\ SIP & \alpha_1 & \beta_{12} & \beta_{13} & \alpha_4 & \alpha_5 \\ SC & \alpha_1 & \alpha_2 & \beta_{23} & \beta_{24} & \beta_{25} \\ CA & \beta_{31} & \alpha_2 & \alpha_3 & \beta_{34} & \beta_{35} \end{array}$$

Functional dependencies are S → C, C → A, S I → P

Lossless Join Decompositions 12

Illustrative example

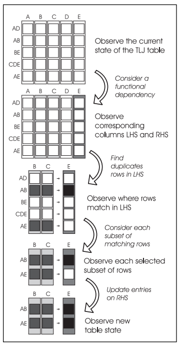
Functional dependencies are S → C, C → A, S I → P
and from these arrive via stage 2 of algorithm at table:

$$\begin{array}{llllll} & S & C & A & I & P \\ SIP & \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 \\ SC & \alpha_1 & \alpha_2 & \alpha_3 & \beta_{24} & \beta_{25} \\ CA & \beta_{31} & \alpha_2 & \alpha_3 & \beta_{34} & \beta_{35} \end{array}$$

at which point no further dependencies apply.

Row 1 shows that the result is lossless

The case study



Observe the current state of the TLJ table

Consider a functional dependency

Observe corresponding columns LHS and RHS

Find duplicates rows in LHS

Observe where rows match in LHS

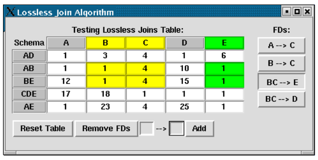
Consider each subset of matching rows

Observe each selected subset of rows

Update entries on RHS

Observe new table state

```
project_table_RHS_FD is
   project(current_table,
           FDs[current_FD]);
```



Testing Lossless Joins Table:

Schema	A	B	C	D	E
AD	1	3	4	1	6
AB	1	1	4	10	1
BE	12	1	4	15	1
CDE	17	18	1	1	1
AE	1	23	4	25	1

WARWICK

Advantages of the observation-oriented model

Pattern of observation used to construct the model is central to learning the TLJ algorithm: the steps involved in constructing the model correspond to exercises that might be used to scaffold the learning of the algorithm.

The model can be used to trace activities outside the scope of the algorithm, such as errors made by students in its application.

The semi-automated model gives useful support to activities such as developing examination questions based on the TLJ algorithm.