Science & Technology Facilities Council
e-Science

A pseudo object DB model
and its applications on a highly complex distributed architecture

Peter Colclough
Gilles Mathieu
STFC

IEEE/IARA DBKDA 2009,
March 1-6 2009, Cancun, Mexico

Outline

- Scope and context
- Initial motivations
- Model description
- Application example
- Ongoing and future work

Scope and context

- What are we exposing here?
  - A methodology for designing DB schemas
    - Non application specific
    - Works on Oracle, MySQL, Postgres, DB2…
  - A technical solution to apply this
    - Set of tools and standard code
  - Proofs that it works
    - We use it, this is not a theoretical dream

Naming convention

In the rest of the presentation, the Model described here will be referred to as:

Pseudo Relational Object Model (PROM)

Initial motivations

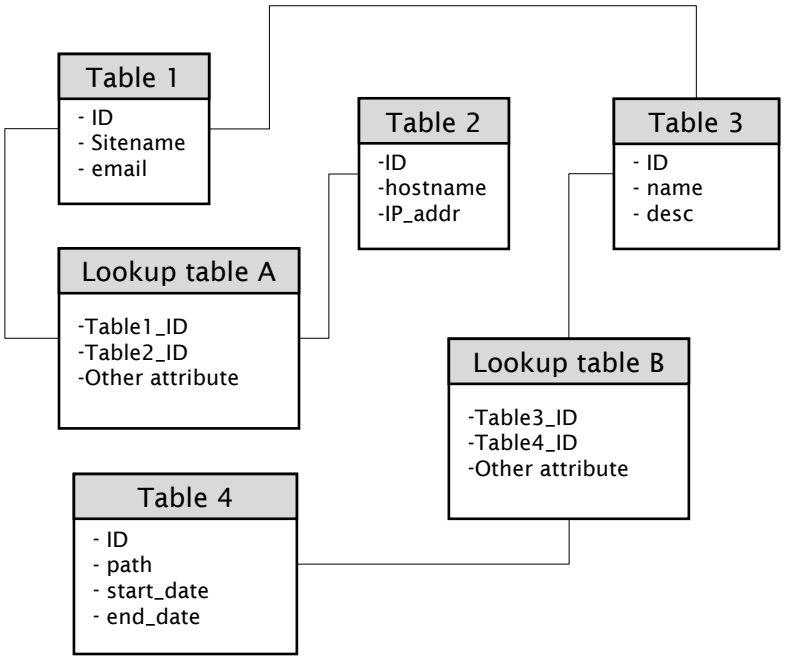
- Limitation of standard RDBMS
  - When schemas have to change quickly
  - When working on large distributed DBs
- Need of a model that:
  - Allows for flexibility
  - Doesn't imply rewriting interfaces

The concept

- Propose a “constraint free” model by:
  - Removing all physical relations
  - Storing them as meta data
    - Table names
    - Object types and actual objects
    - Link types and actual links

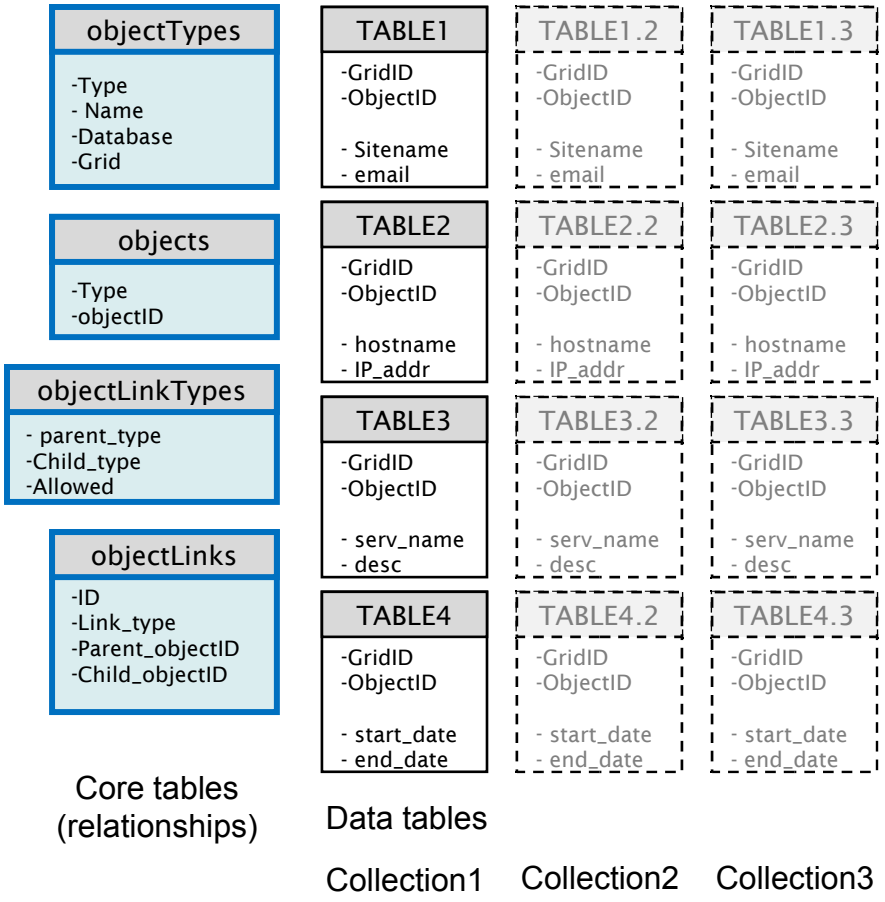
PROM "big picture" (1)

Standard relational model



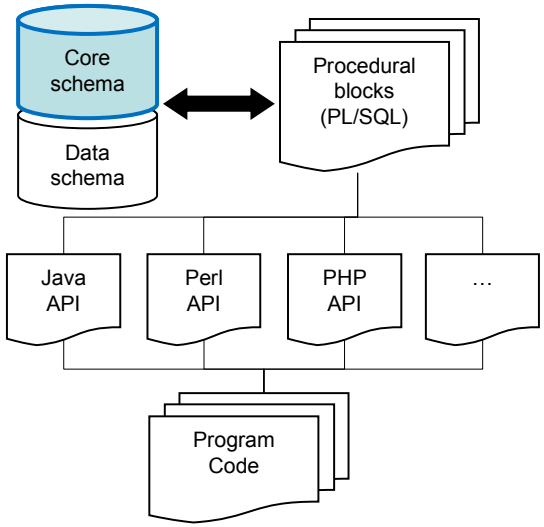
- Physical Data Tables
- Hard coded relationships and constraints

PROM



PROM "big picture" (2)

- Meta data separated from actual data
- Homogenous access through Standard APIs
  - Generic functions for select, insert, delete, update…

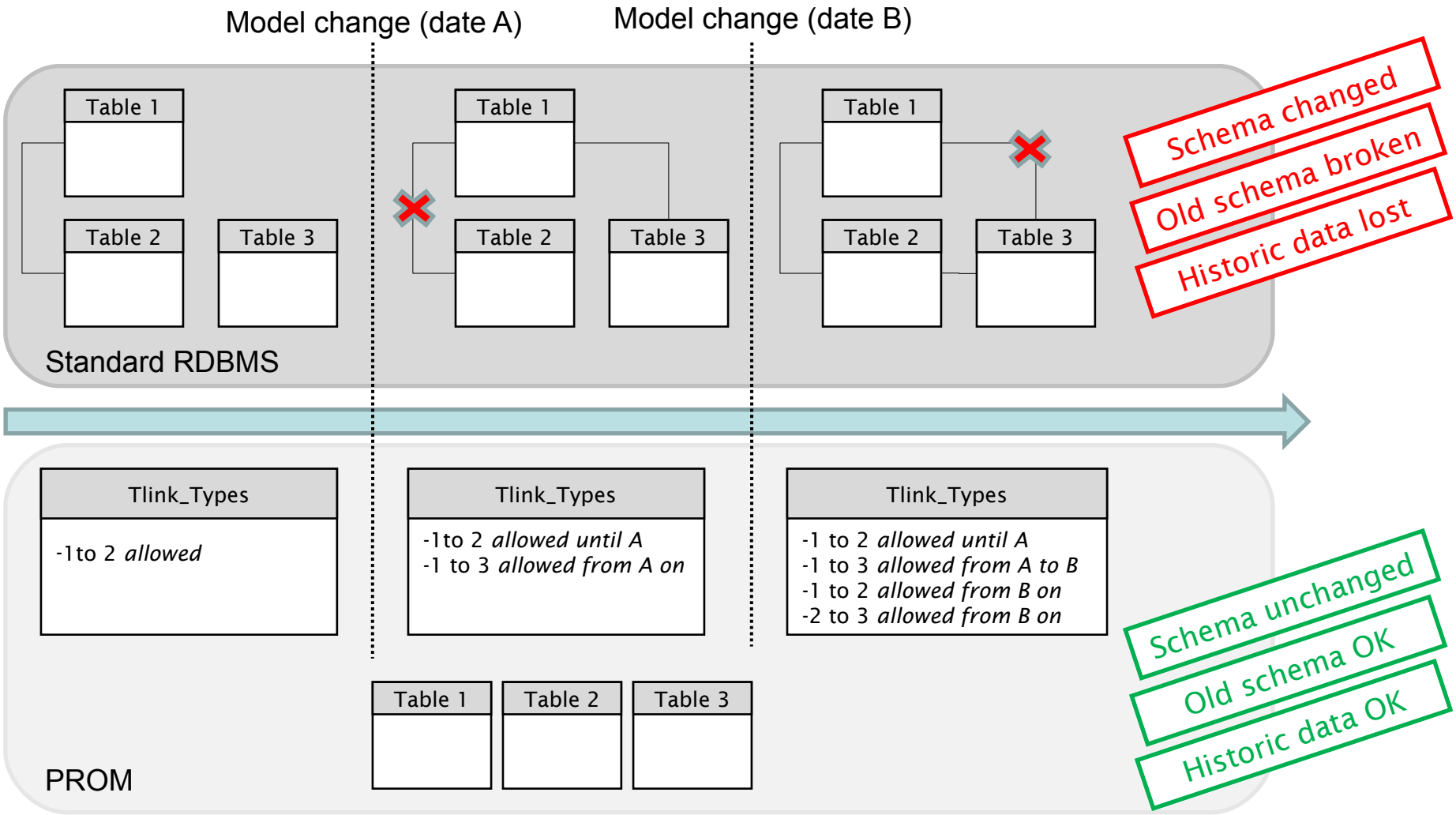


PROM: the "time" factor

- Meta data have dates on and off:
  - Gives the validity of the information
    - A relationship is actually removed by setting it off at a certain date
  - Allows to keep history
    - Historic data and relationships are kept
    - "snapshots" can be retrieved at any time

Different schema versions can work concurrently !

PROM: the "time" factor



Cost comparison

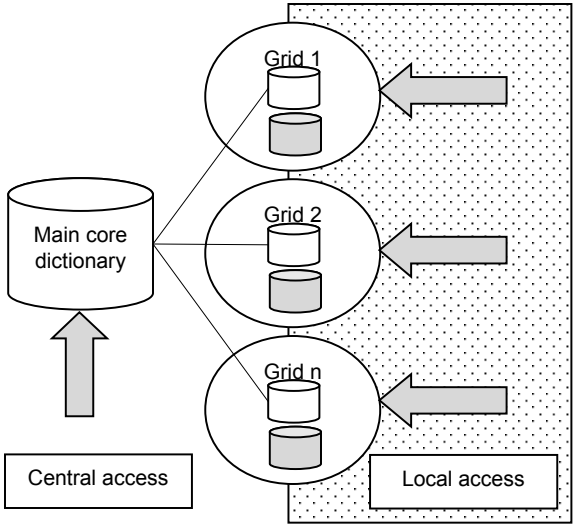
Modification	Standard RDBMS	PROM
Add or modify fields in existing tables	-Modify SQL code -Modify client code -**Breaks old schema**	-Modify client code -**Old schema still valid**
Add a new table	-Write more SQL code -Modify client code	- Modify client code
Add a n-n relationship between existing tables	-Add a lookup table -Modify SQL code -Modify client code -**Increases model complexity**	-Add a metadata entry -Modify client code -**No added complexity**
Remove a n-n relationship between existing tables	-Remove lookup table -Modify SQL code -Modify client code -**Breaks old schema**	-Flag metadata entry -Modify client code -**Old schema still valid**

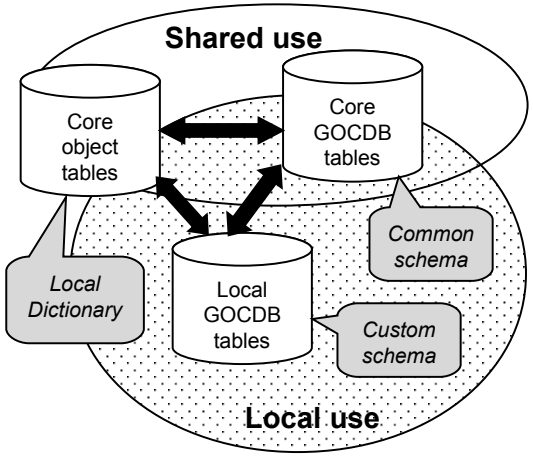
Application example

- Grid Operations Centre DB (GOCDB)
  - Official data repository for:
    - Enabling Grids for E-SciencE (EGEE)
    - Worldwide LHC Computing Grid (WLCG)
  - Stores grid topology information
  - Current Evolution needs
    - Distribution per country
    - Central view and regional customisation

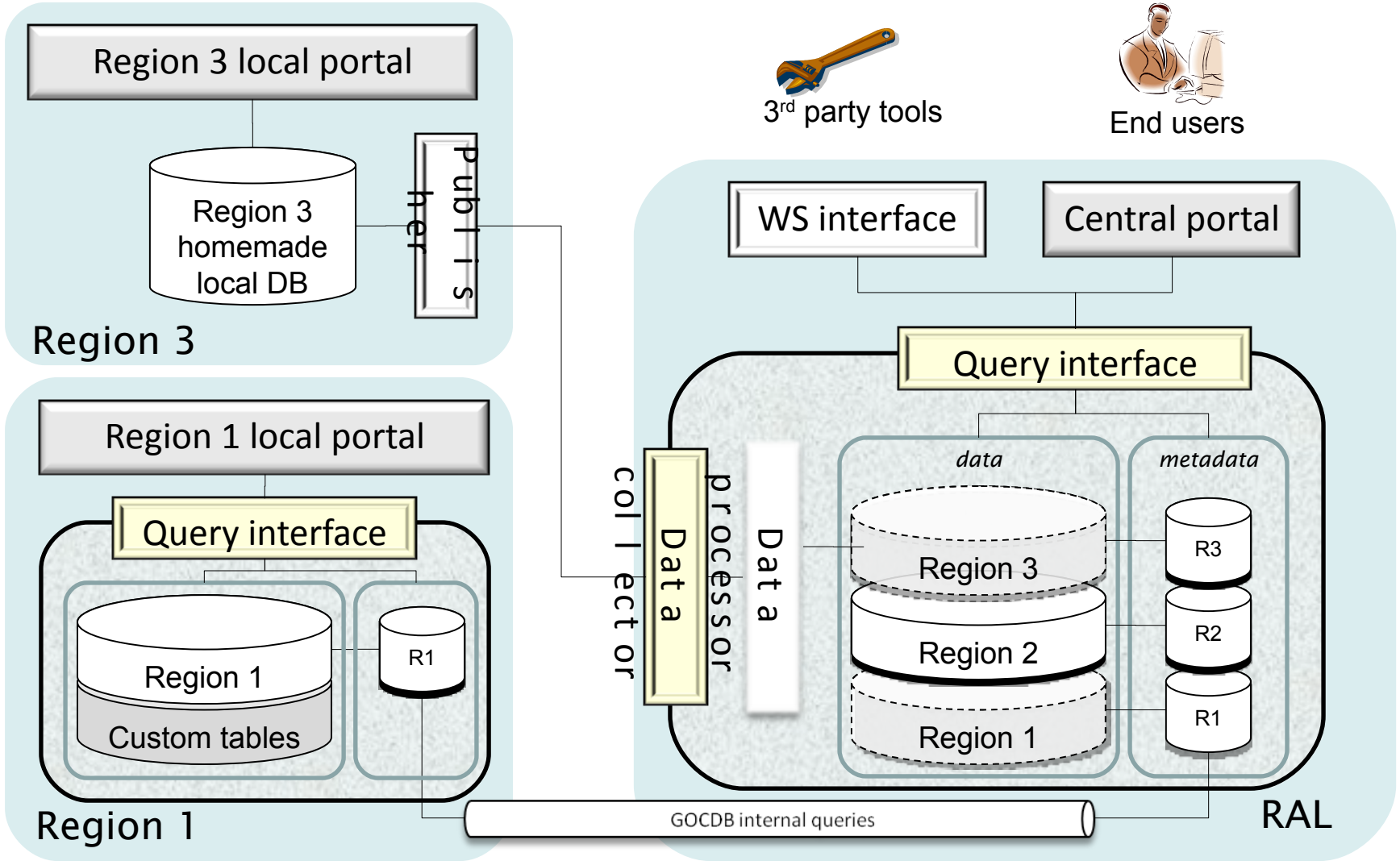
Proposed approach

- Logically split the central DB into regional instances
- Allow both local customisation and central coherency





The big use case



Ongoing and future work

- Allow heterogeneous link types
  - Inter-DB links
- Increase API access
  - Add languages
- Add admin functionalities
  - Checks, warning and monitoring

For more details…

- About PROM
  - Peter Colclough - biton@compuserve.com
- About GOCDB
  - http://www.grid-support.ac.uk/content/view/406/290/
  - Gilles Mathieu – gilles.mathieu@stfc.ac.uk