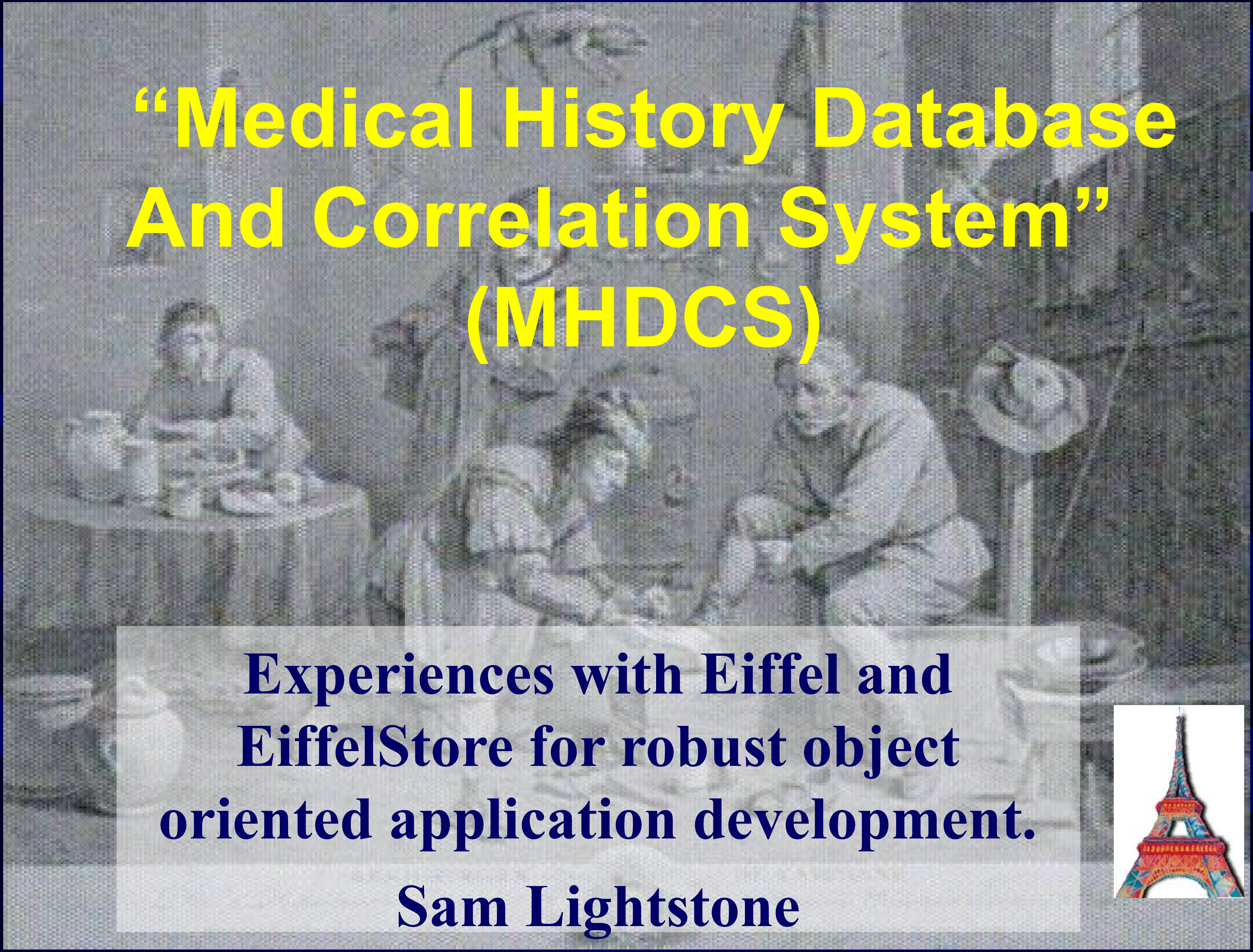


“Medical History Database And Correlation System” (MHDCS)

Experiences with Eiffel and EiffelStore for robust object oriented application development.

Sam Lightstone

Agenda

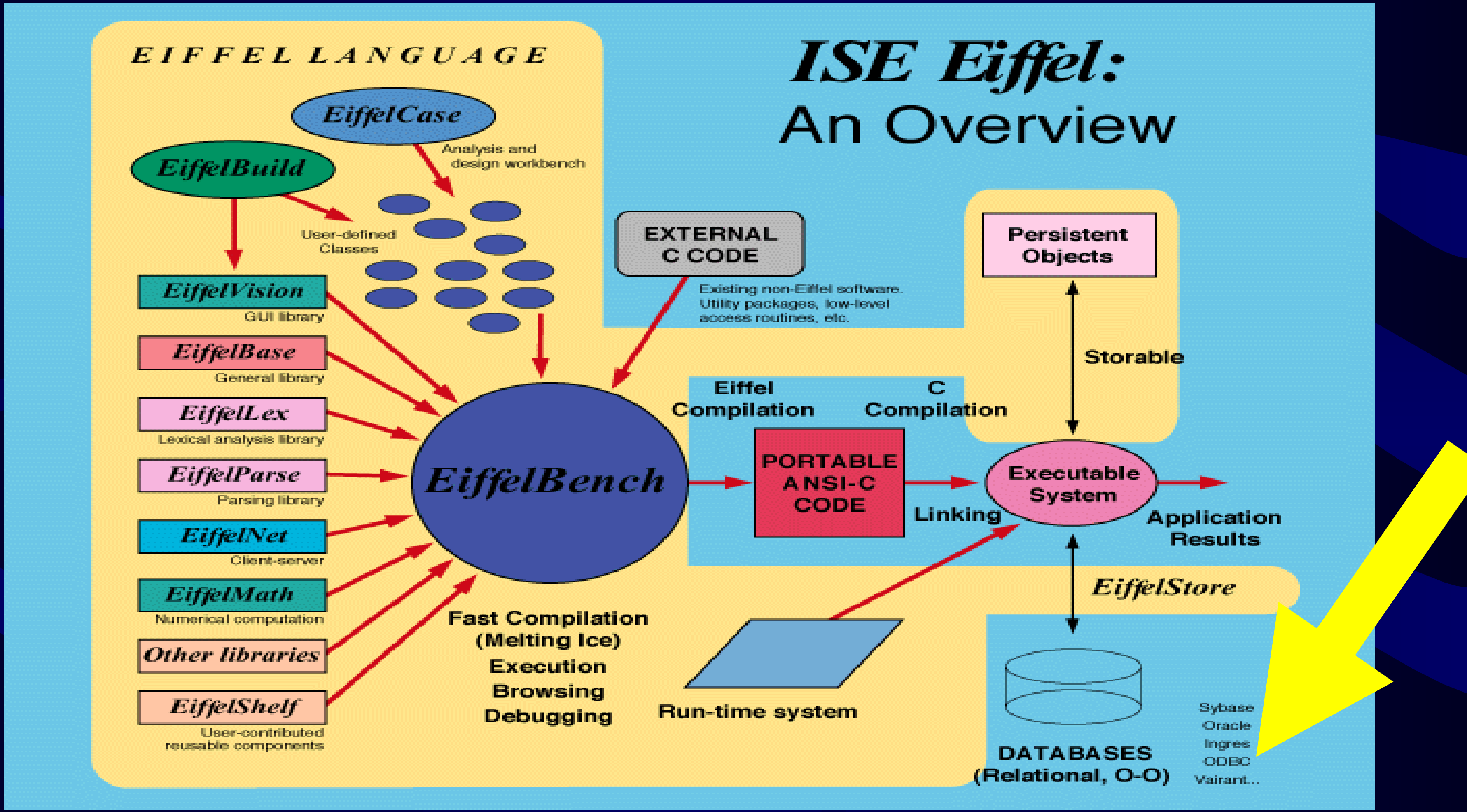
- Project intro
- EiffelStore
- Software demonstration
- Design overview
- EiffelWow & EiffelFrust
- Lessons learned
- Conclusion

Project overview

- A rocket weighs 1200 kg and carries 200kg of fuel. It expends 5kg of fuel per second. The rocket leaves the surface of the earth at 400 m/s and accelerate at 4 m/s^2 until fuel is exhausted. Write a program which calculates….

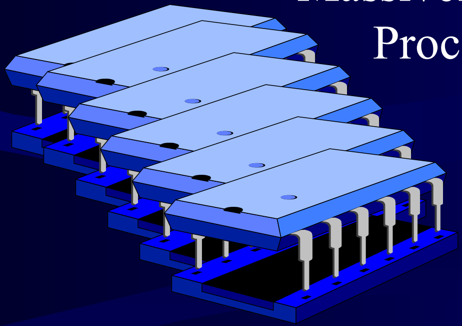
Sound familiar?

Something caught my eye...



Why use a relational database?

Extensive support for Symmetric Multi-processors, and Massively Parallel Processors



Native support for parallel asynchronous disk I/O, heavily tuned asynchronous prefetching.

- Multi-user, high concurrency.
- Crash recovery.
- Backup & Restore and ROLLFORWARD recovery.
- Super reliability.
- Super scalability to thousands of disks and hundreds of CPUs.
- Free world class data processing, such as data sorting, indexing.
- Built in client-server capability, authentication, clustering and fail-over.

Project summary

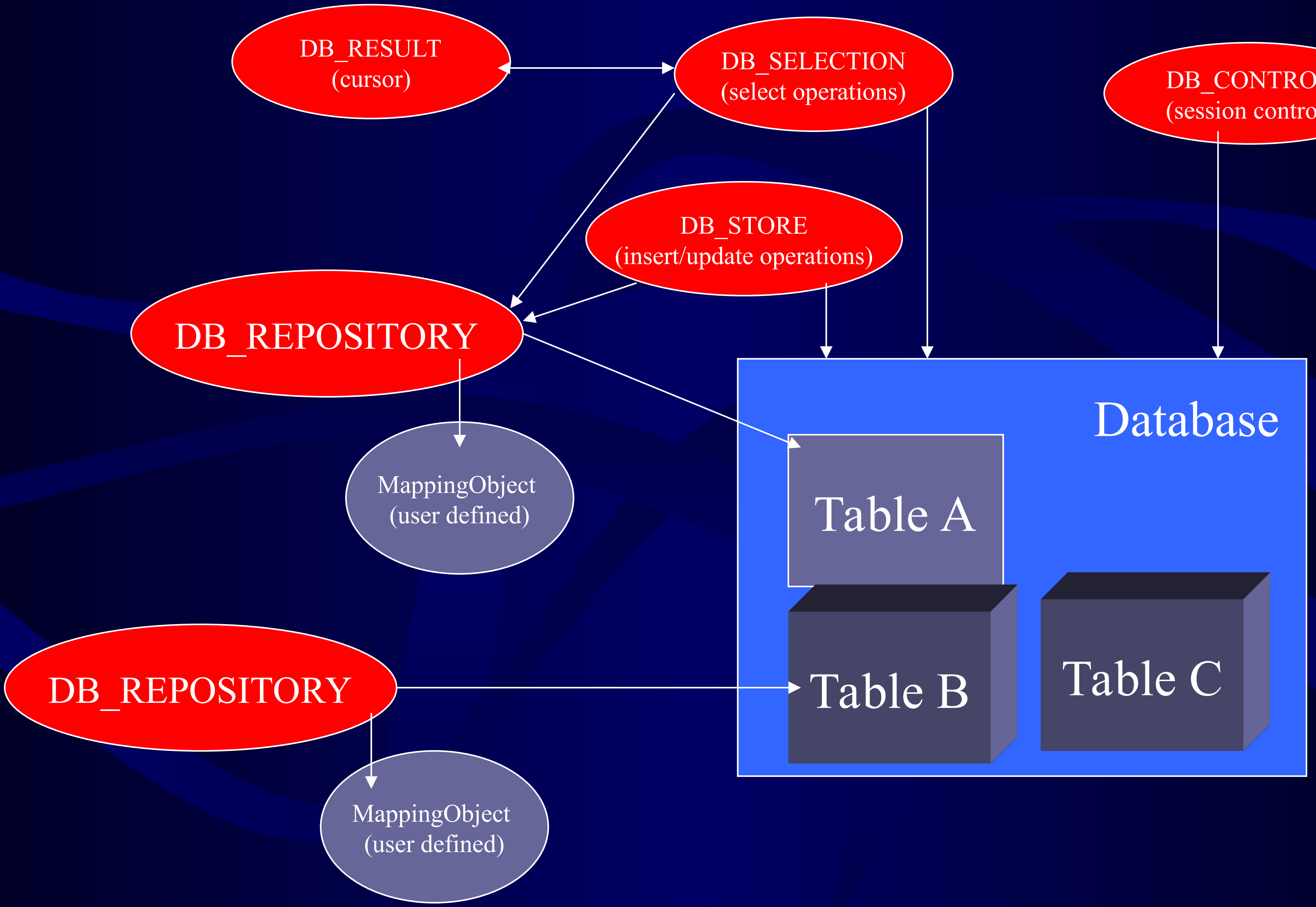
(more details to come in the demo...)

- A medical database application.
- Used by medical practitioners.
- Stores consultation data in relational tables.
- Correlates patients with relatives.

EiffelStore

- The EiffelStore Application Programming Interface (API) is DBMS-independent
- The list of currently supported DBMSs: ODBC (dozens of database products, in particular on Windows); Oracle (Unix, Windows); Sybase (Unix, Windows); Open Ingres (Unix, Windows); MATISSE object-oriented and multimedia database.
- Conceptually organized as databases and repositories.

EiffelStore



DB_RESULT (cursor)

DB_SELECTION (select operations)

DB_CONTROL (session control)

DB_STORE (insert/update operations)

DB_REPOSITORY

MappingObject (user defined)

Database

Table A

Table B

Table C

DB_REPOSITORY

MappingObject (user defined)

Demo modeling

- Two relational tables, with associated indexes.
- One table holds OHIP users. The other holds medical consultation records.
- For demonstration purposes, the database is pre-loaded with 58 OHIP users, consisting of students from this ConGESE course, fictional parents, and children.
- A medical event generation routine was developed in C to generate 5.5K medical consultation records for these 58 OHIP users.

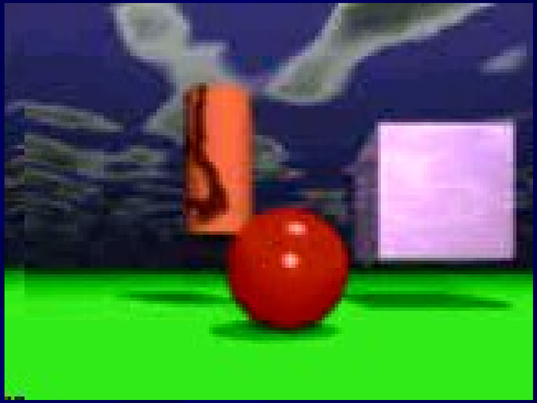
Table: “ohip_users”

Contains set of known OHIP patients. Each record contains name, date of birth, parents’ OHIP numbers.
The database enforces uniqueness of each patient’s OHIP number.

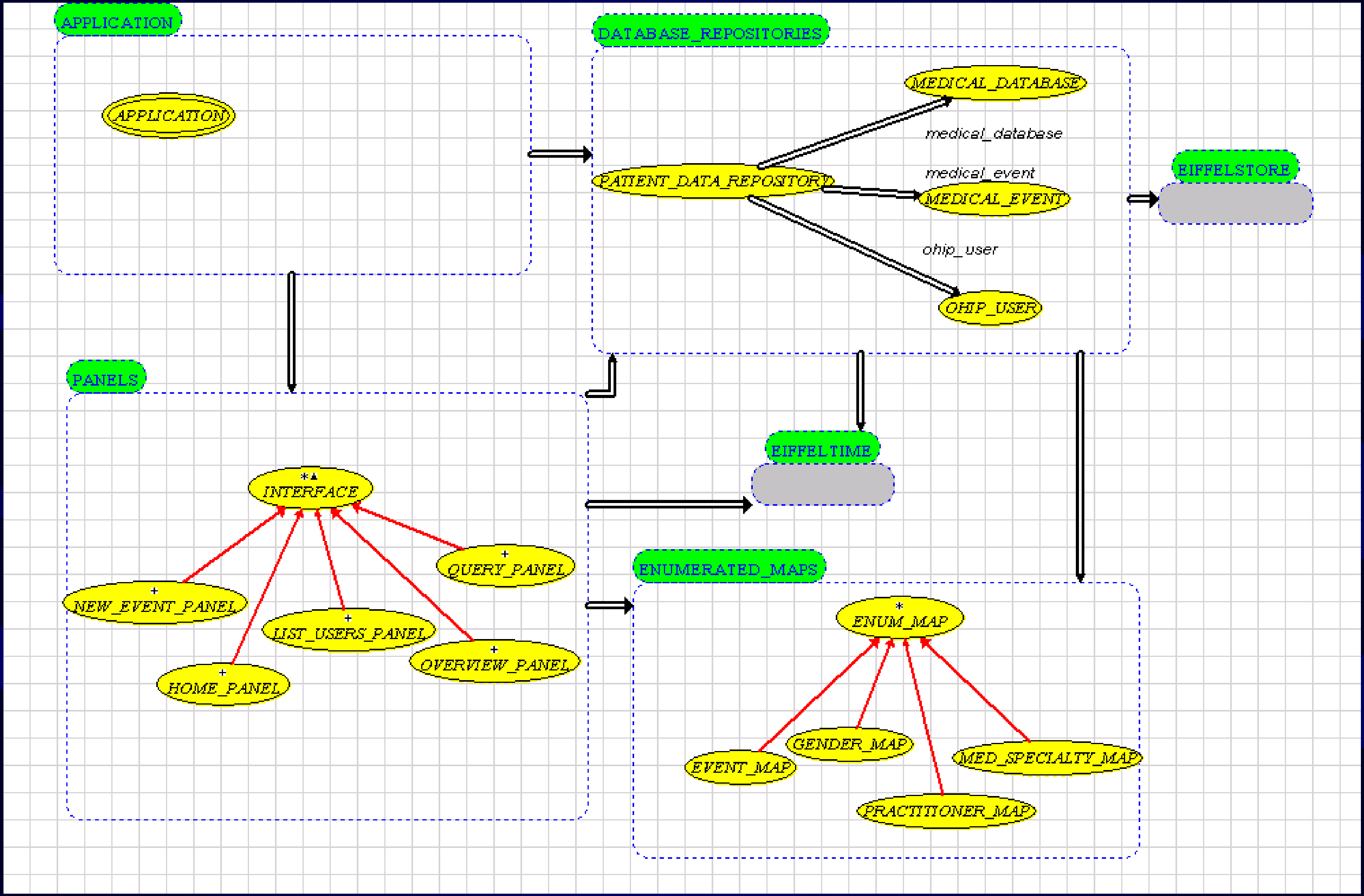
Table: “medical_events”

Contains medical consultation records. Each record includes: Practitioner, hospital, patient OHIP number, consultation type, diagnosis, comments, date/time.

It’s now time for our feature presentation….



System Design



Interesting Design Issues

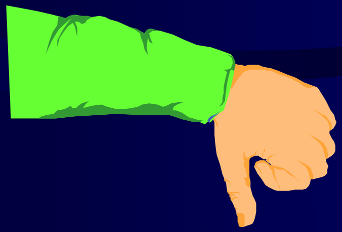
- What is a panel?
- Multi-panel transition.
- Single choice principle for enumerated types, their operators and external forms.
- Best correlation method to find relatives?

EiffelWow

- Pure OO. I like it!
- Design by contract, powerful technique. Note the self testing aspect.
- Melting Ice technology… super cool!
- EiffelStore, great OO interface to relational DB. More on this later.
- Beautiful suite of tools.
- Rich class library.
- A brilliant conceptual framework and implementation suite. Full marks for good ideas.

EiffelFrust

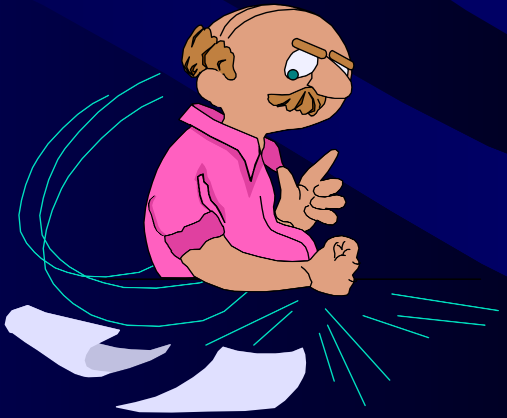
- Compilation tells first error only!
- Indent through “tab” convention, limits use of editors.
- Terse commenting style is too terse. For example: relative parameters. (b relative to a or a relative to b?)
- Frequent traps, exists using Case & Bench. This improved with Bench 4.4.
- Counter-intuitive GUI, but I got used to it.



EiffelFrust

(bugs, bugs, bugs...)

- No support for DOS EOF character!!!
- io.readchar doesn’t work (had to write my own).
- EiffelStore… too many to mention. It didn’t work at all.



EiffelFrust

- Hard to believe Eiffel has existed since 1985. It does not demonstrate the maturity of an aged system.
- Eiffel development environment is a bad example of Bertrand Meyer's OO Software Construction methodologies.

> ***"In software development, reliability should be built-in, not an afterthought."***
> Bertrand Meyer, President, Interactive Software Engineering Inc.

Lessons

- There is such a thing as Eiffel. It has some brilliant ideas.
- Componentry reuse, although wonderful in principle, is development and serviceability risk.
- System design using ADTs yields a neatly organized system framework.
- Contracts are potentially powerful, but practically limited. Hard to use comprehensively. See your local patent office for examples of software patents…
- Design by contract yields a large potential quality boon by its self testing nature.
- The "single choice principle" is singularly important.
- Most of the important Eiffel principles are not limited to OO SC.

Conclusion

- Medical History Database and Correlation System. A fun Eiffel project.
- EiffelStore, nice range of DBMS support. Nice object mapping.
- Robustness is a major issue for the Eiffel development environment.
- Software engineering is useful and important. It is not limited to OO.