Directions in Applications and Systems

Tuesday, March 25, 2002 Michael Minock

What is the point?

The relational model has triumphed.

- Banks use it to record account information and transactions.
- Airlines use it for their booking and reservation systems - or do they?
- Small businesses use it for simple inventory, billing and accounts receivables operations.
- Hospitals use it for medical records.
- ...

What is the point?

A fair question.

Why do we need more data models? Isn't the relational data model good enough?

And how often are 'advanced models' used to do the things that relational systems can already handle?

So we must make sure that the applications that we envision really do require extra modeling power. If they do not, then, chances are, a mature relational system is the best choice.

So this class is (implicitly) also about the relational data model. And what its limits are.

Some Motivating Examples

- We need to integrate information over a large complex, multi-concept information space with information at different levels of aggregation and of different types (image, audio, text, web-page, attribute-based). (E.g. intelligence gathering, market analysis, environmental, etc.)
- We need top performance. Any extra bit of performance lets us scale to bigger data sets. And handling bigger data sets gives some type of advantage. (E.g. telecommunications switching equipment, scientific modeling)
- We need to deeply model many types of objects and their sub-categories. Objects may also have complex behaviors. Objects may be mutable and have complex lifetimes.(E.g. models of virtual worlds).

Examples continued

- We have complex *ancestor* or *bill-of-sales* type queries. Over relational models these queries require a program to be written in the host language. E.g:
  - "Does Bush have any royal ancestors?"
  - "Are there any parts within my car that were manufactured in Chile in the late 1980's?"
- We need to be able to track both the times when financial transactions occurred and the time that corrections to the transactions were issued. We want to search this historical information looking for evidence of fraud.
- We have travel destination data and we wish to issue queries like finding all the non-crowded beaches that are walking distance from a train stop and are greater than 5km from a sewage treatment plant.
- Uncertainty - We only have approximate information on the positions of a set of objects we are tracking (E.g. birds). Some areas within our scope of observation have no information whatsoever. The user wants to know the likelihood of an object (a bird) being within a certain sector. How do we handle this?

The right model-application fit

There are countless examples (start thinking about generating some of your own).

The techniques and models that we describe in this class will give you an idea of how to tackle the above problems - and with which model(s).

Your system choice will depend what is available or your capability to build your own 'system' from scratch.

Before we start with the issue of Semantic Modeling, let us take a look at what some leading database researchers are saying about the direction that *systems* are taking.

The Asilomar Report on Database Research 1998.

A group of leading DB researchers met in Asilomar, California.

Driving Forces:

- the Web
- the need to unify program logic and databases
- hardware advances

The Web

Databases technology has contributed little, thus far, to core web protocols

DBs appear in peripheral roles:

- high-end web servers
- E-commerce servers
- content repositories
  - personalization (e.g. personalized mannequins)
  - wherever data modeling tasks are significant.

XML - a step backward?

XML is essentially a hierarchical data model

XML opens the way to client side data caches with updates (yikes!)

But XML is very influential...

XML Schema seems to be an acknowlegment from W3C

Unifying Data and Programs

Traditional databases housed data.

Relational systems incorporated triggers and stored procedures as an after thought.

We would like to be able to have DBs serve as repositories of active components.

This would facilitate the inter-operation and rationalization of organizations large, complex applications

And this helps in the integration, evolution, migration, and replacement of application systems.

Thus DBs should be more 'application aware'.

Hardware Advances: Scale up to Mega-servers

Moore's law (Bits per unit area of silicon doubles every year - since 1962) is creeping up on traditional assumptions

Within 8 years, a terebyte of main memory as a buffer to a 100 terabyte DB.

Hot tables and all indices in main memory. B+ trees not necessarily the best indexing strategy any more.

Hardware Advances: Scale down to appliances

There will be billions of people using trillions of gizmos.

Each gizmo maintains some information store.

Gizmos will interconnect with gizmo servers.

Will traditional client server architectures or three-tiered middle-ware architectures endure?

Plug and Play

Human attention is the most precious resource now.

Databases should accommodate **no knobs** operation.

Data bases should:

- be self-tuning
- perform information discovery - like DNS
- supply *meta-data*

Federate Millions of Databases

1000 or more sites all providing views over a global schema.

Each site is locally autonomous.

The global system must present semantically richer answers:

- evidence accumulation
- incompleteness information
- inconsistency information

This requires new integration tools.

This requires new distributed processing algorithms.

The Grand Challenge

The Information Utility: Make it easy for everyone to store, organize, access, and analyze the majority of human information online.

It will be on the web, but will we be able to get at it?

And what does this have to do with advanced data models?

A lot - but not everything.

The data models should be studied in relative isolation.

These Asilomar issues are important when we begin to speak about where and how advanced data modeling will be used. And what additional system features might be present.