

# OpenRecord 0.1

a tool for web-based peer production

august 2005

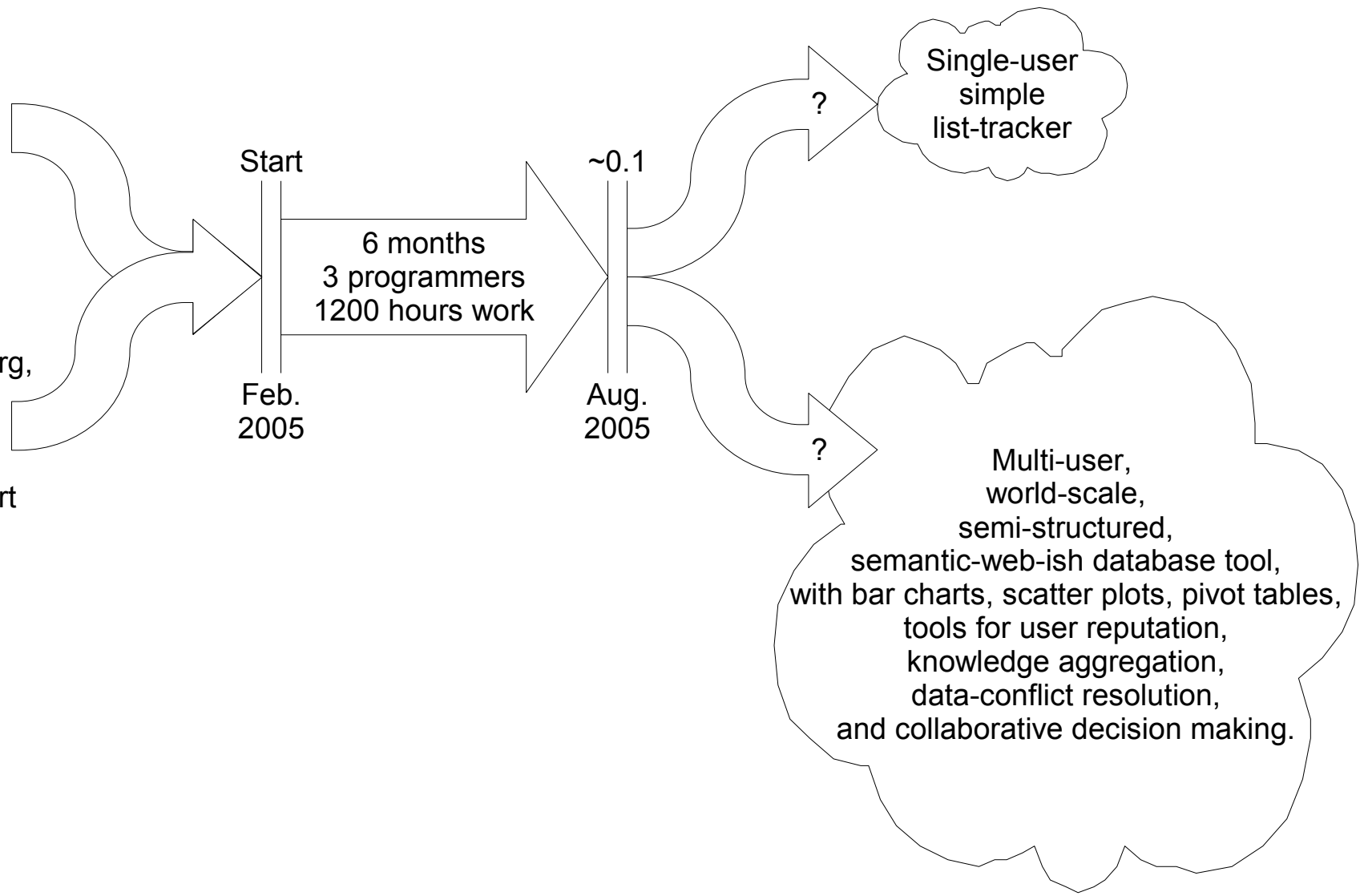
# Project History

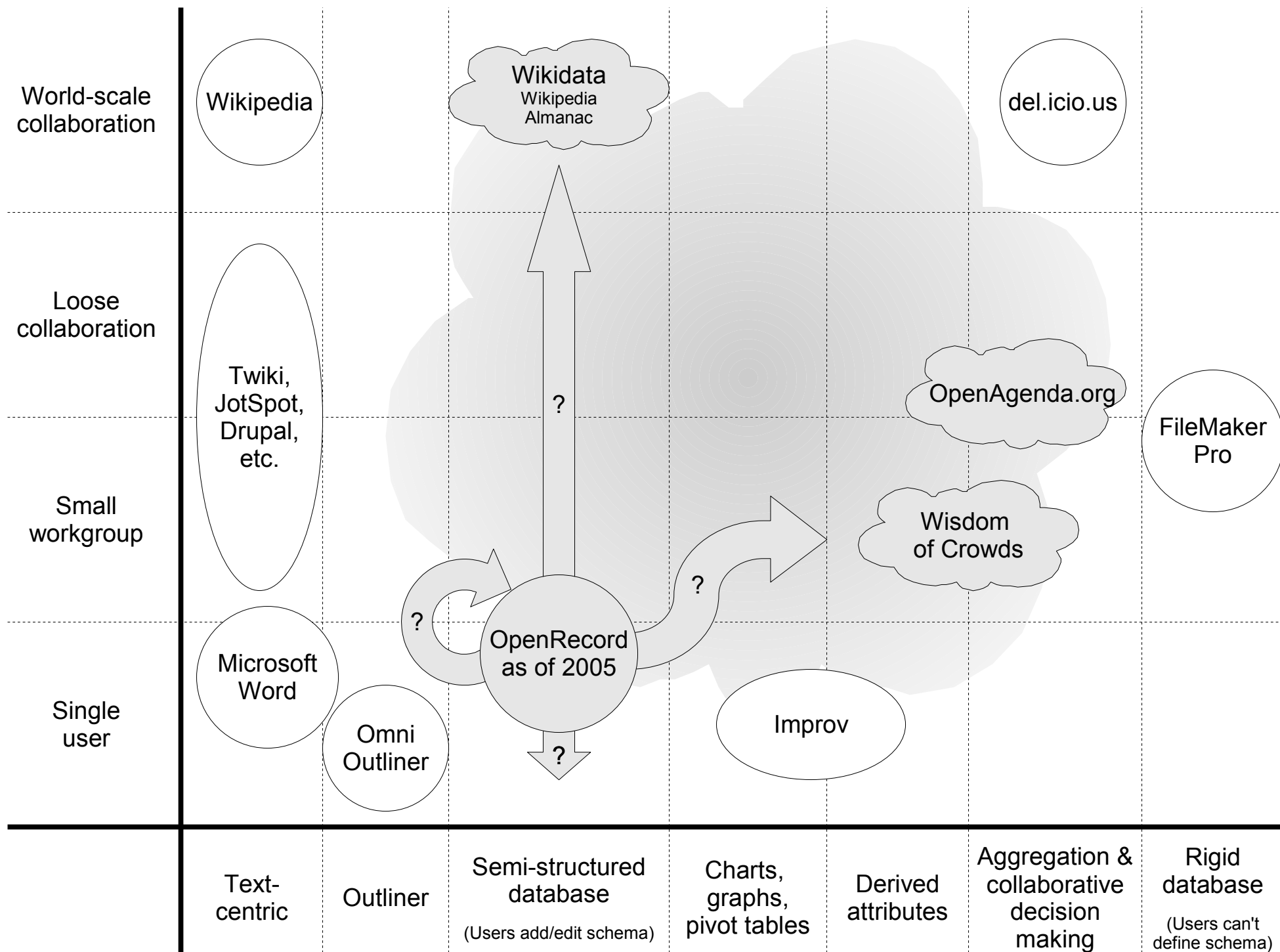
Motivated by:

- Wikipedia
- Chandler
- RDF
- del.icio.us
- Improv

Required for:

OpenAgenda.org,  
a Web-based  
collaborative  
philanthropy  
decision-support  
tool





# Potential Applications

## Museum Inventory

- small town museum
- 6 volunteers
- no paid staff
- no software budget
- no database experience

### Set up to keep track of:

- objects on exhibit
- rooms and displays
- donors

---

## Wikidata Almanac

- could be a Wikimedia project?
- “world-scale collaboration”

### Set up to keep track of:

- cities, countries, populations
- chemicals & compounds
- schools & universities
- corporations & products
- elected officials & political parties
- poverty, employment, wealth
- ships, planes, trains, cars
- lifespan, health, education
- births, deaths, migration
- etc.

## OpenAgenda.org

- collaborative philanthropy
- collect quantitative metrics
- do cost-benefit comparisons

### Set up to keep track of:

- poverty, disease, death
- available interventions
- non-profit organizations
- budgets and outcomes
- openness: survey results
- cost effectiveness metrics

---

## Tobacco Control

- Globalink is a network of tobacco control advocates
- thousands of people
- “loose collaboration”

### Set up to keep track of:

- violations of smoke-free laws
- violations of the 1998 Master Settlement Agreement (MSA)
- tobacco industry funded research and researchers
- organizations that accept funds from tobacco firms
- compliance with WHO FCTC (Framework Convention on Tobacco Control)

## Stamp Collecting

- single user
- thousands of items
- simple structured database

### Set up to keep track of:

- postage stamps
- coins, currency
- folders, drawers, and albums

---

## Book Club

- small workgroup
- books, reviews, meetings

## Investment Club

- small workgroup
- stocks, ratings, industries

## Family Genealogy

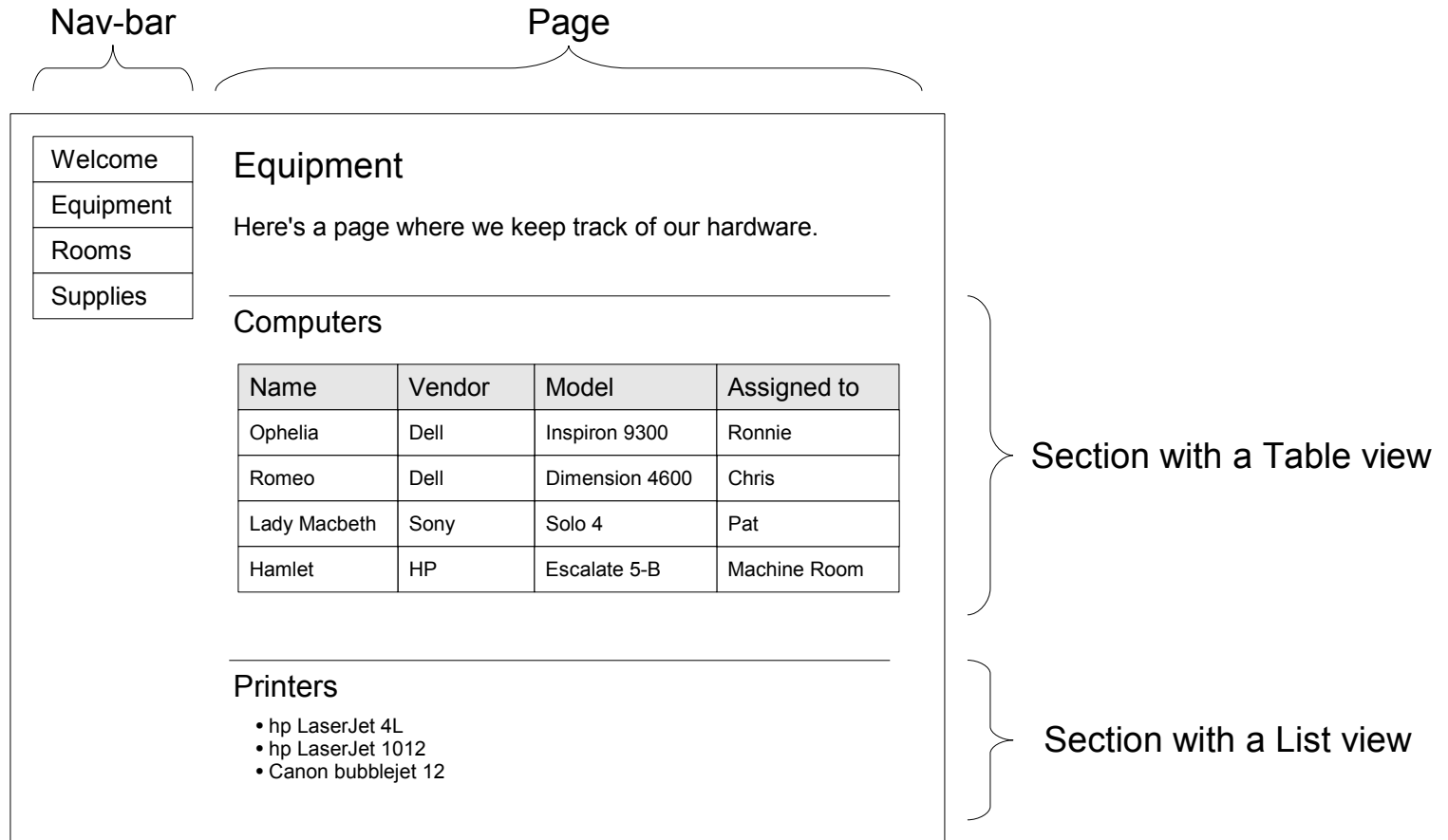
- extended family as workgroup
- people, places, cemeteries

etc.

# Content

Items  
Attributes  
Categories  
Tags  
Entries  
Types  
Connections  
etc.

# Presentation



## Layout

- a repository has a set of pages
- a page can have sections
- a section has a query
- users can edit queries using a query editor
- a query returns a list of content items
- sections use different types of views to display results
- users can pick what view to use to display the result list
- different sections can show different views of the same content items

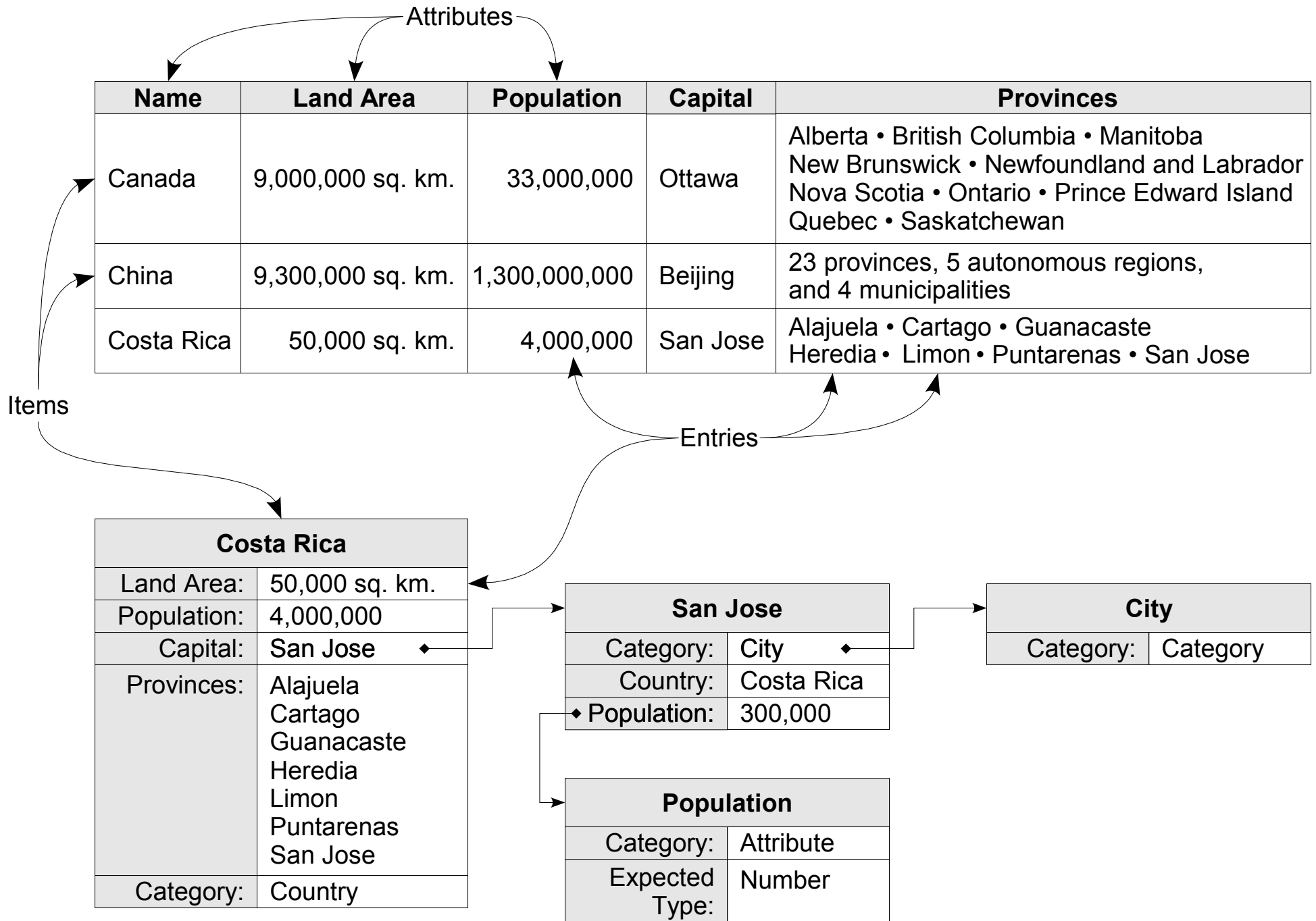
## Views today

- Table view
- List view
- Detail view
- Master-detail (soon)  
(like an e-mail reader)

## Views someday

- Outliner
- Bar chart
- Scatterplot
- Pivot table
- Gantt chart
- Rollback history
- Diff version
- etc.

# Data model



# Data model

## Categories

- items can be assigned to categories
- a category is a little bit like the Chandler notion of a Kind
- you can search for all items in a category
- an item can belong to more than one category (kind) – San Francisco can be both a City and a County
- an item does not need to belong to any category – “kind-less items”
- you can switch an item from one category to another
- a category is really just an item itself, so a category can have a description and other attributes
- any item can serve as a category

## Items

- an item can have entries for attributes
- an item can use any attributes, regardless of what categories it's in
- an item can have an attribute that no other item has, or that isn't typical of items in this category

## Attributes

- all attributes are really “ad-hoc” attributes
- we are “attribute-centric”, in the sense that all schema is associated with attributes
- two unrelated items can have the same attribute
- an attribute can have an “expected type”
- an attribute can have an “inverse attribute”
- an attribute is really just an item, so you can add ad-hoc attributes to the attribute
- any item can serve as an attribute

## Entries

- an item can have one or more entries for each attribute
- each entry holds one data value
- an entry can hold a literal value or an item reference
- an entry knows what type of value it holds
- an entry's type does not need to match the “expected type” of the attribute

## Types

- literals: Text, Number, Date, URL, etc.
- bi-directional item connections
- one-way item references
- dates can be “fuzzy” – users can enter dates like: 8/18/2005, August 1944, 1944, Today, or Tomorrow

## Tags

- items can have tags
- tags are like the tags in del.icio.us or flickr
- you can search for all items with some tag
- the tags are items themselves, so a tag can have a description as well as other attributes

## Names and Identifiers

- every item has its own URL
- any item can have a name
- names never need to be unique – any two items can have the same name
- an item can have more than one name
- names are never used as unique ids
- unique ids are never used as names
- unique ids never appear in the UI\*

\*(except in URLs)

## Forking and Merging

- a repository can be forked
- any two repositories can be merged, including two forks of a repository

## History

- we keep a complete transaction log
- a repository can be rolled back to any point in its history

## Attribution

- an item knows who created it, and when
- an entry knows who created it, and when

## No Kinds

- there is no real notion of kinds

## No Collections

- there is no real notion of collections
- however, a query result set is like a collection, and the list of entries for an attribute is like a collection

## No Cardinality

- there is no real notion of cardinality
- all attributes can be multi-valued

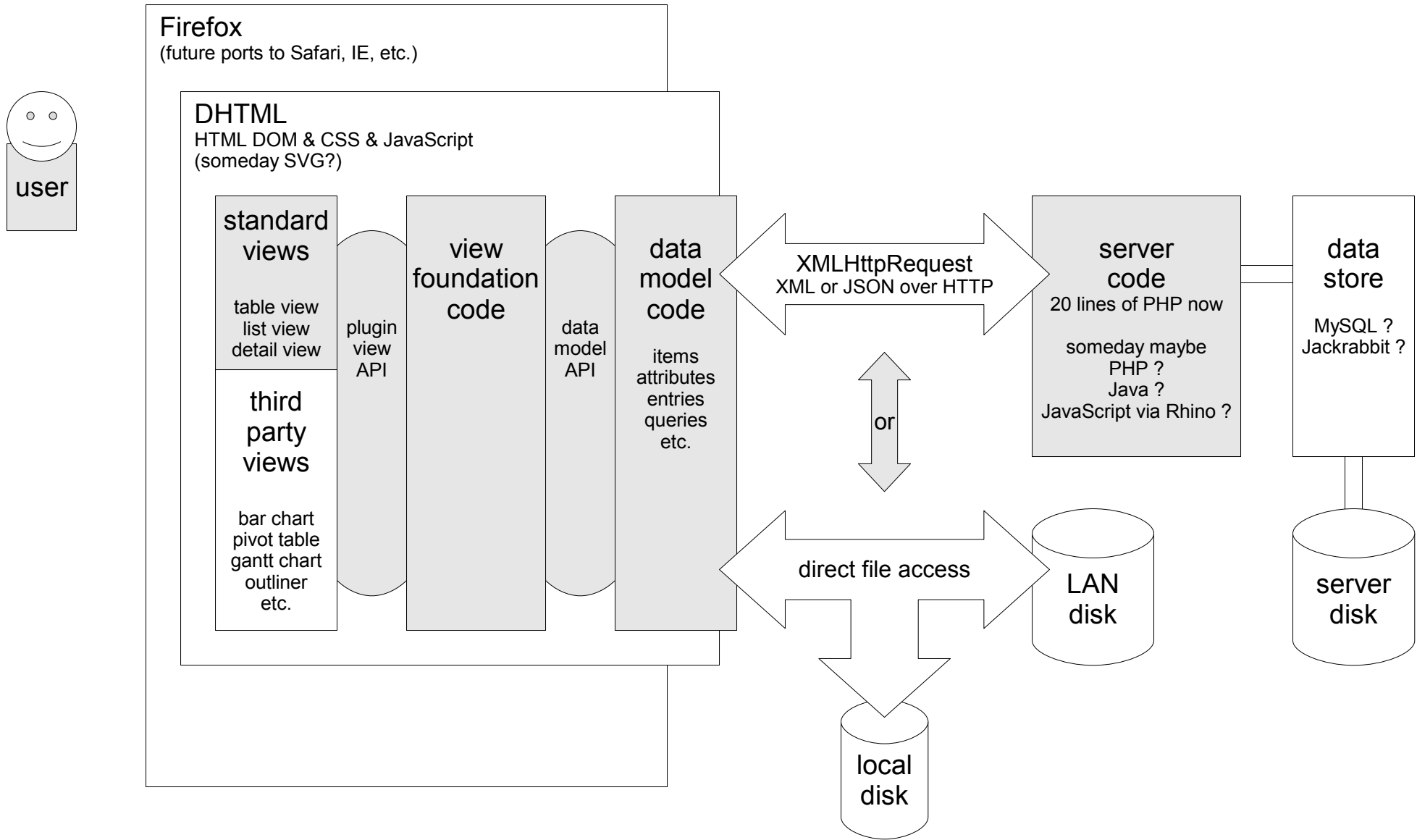
## No Permissions

- all content is readable by anyone
- anyone with an account can add content

## No Anonymity

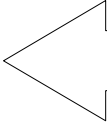
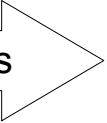

- all content can be attributed to someone

# Architecture





# Open source licenses

	 no restrictions	attribution requirement	 more restrictions
copyright provisions	creative commons public domain dedication	MIT license    BSD license	
patent provisions		apache license	GPL    etc.

## Contributor license agreement (CLA)

### Option 1:

A peer-to-peer CLA agreement between a group of contributors, “executed in counterparts”.

### Option 2:

Create a non-profit foundation. CLA is between the contributor and the foundation.

### Option 3:

Operate as an autonomous project of an established foundation. CLA is between the contributor and the foundation. Candidates might include:

- Tides Foundation ?
- Dojo Foundation ?
- OSAF ?