Web Information Management and Knowledge Bases

Serge Abiteboul
INRIA Saclay & ENS Cachan

ICWE, Wien, 2010

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

INRIA

centre de recherche SACLAY – ÎLE-DE-FRANCE

Webdam

Context: Web data management

- Scale (lots of users, servers, large volume of data)
- Relation → Tree (HTML, XML, Xpath…)
- Centralized → Distributed (Web services, BPEL…)
- Precise data → Incomplete, probabilistic (belief, trust)
- Precise schemas → Ontologies (RDF, OWL)

Moving from publish to sharing (Web 2.0)

Moving from text to data and semantics (Semantic Web)

And more (Web of objects, Web 4D…)

From Relational data management to Web data management

The success of the relational model was due to formal foundations

Web data management is even more complex

It is time to stop hacking

It is important to develop formal foundations?

- Logic of course: first-order, monadic second-order
- Tree automata
- Probabilities
- …

Context of the works presented here

Active XML 2002-2008

2008-2013, European Research Council project

All these works joint with many colleagues/students, in particular:

Tova Milo (Tel Aviv)	Victor Vianu (UCSD)
Luc Segoufin (INRIA)	Ioana Manolescu (INRIA)
Georg Gottlob (Oxford)	Alkis Polyzotis (UCSC)
Angela Bonifati (Cozenza)	Marie-Christine Rousset (Grenoble)
Omar Benjelloun (Google)	Bogdan Marinoiu (SAP)
Pierre Bourhis (INRIA)	Alban Galland (INRIA)
Marco Manna (Roma)	Nicoleta Preda (Franhoffer)
Zoe Abrams (Google)	Emmanuel Taropa (Google)
Bogdan Cautis (Telecom Paris)	Spyros Zoupanos (INRIA)

Organization

A holistic approach based on a distributed knowledge base

What data do you use?
Example: personal data management

Real data

- Pictures, movies, music, emails, ebooks, reports
- Main information from access viewpoint: metadata, e.g., format, name, time, provenance, etc.
- Web sites

Personal and social annotations

- Semantic tagging, e.g., of pictures in Picasa

Ontologies

- Essential for data integration: RDFS, OWL…

What data do you use? (continued)

Localization information

- Bookmark list, e.g., delicious or Mozilla Weave
- The systems that I control: laptop, iPhone, desktop at work, n-play box…
- The system where I have data: Facebook, Youtube, Gmail…
- The systems where my friends/contact put data
- What is where: Sigmod's pictures at Mohan's Facebook account

Access information & access rights

- Login/passwd, e.g. in Mozilla Weave
- E.g., rights of groups in social network
- Members of these groups

Services: Search engines, yellow pages, dictionaries…

And more…

Life is tough

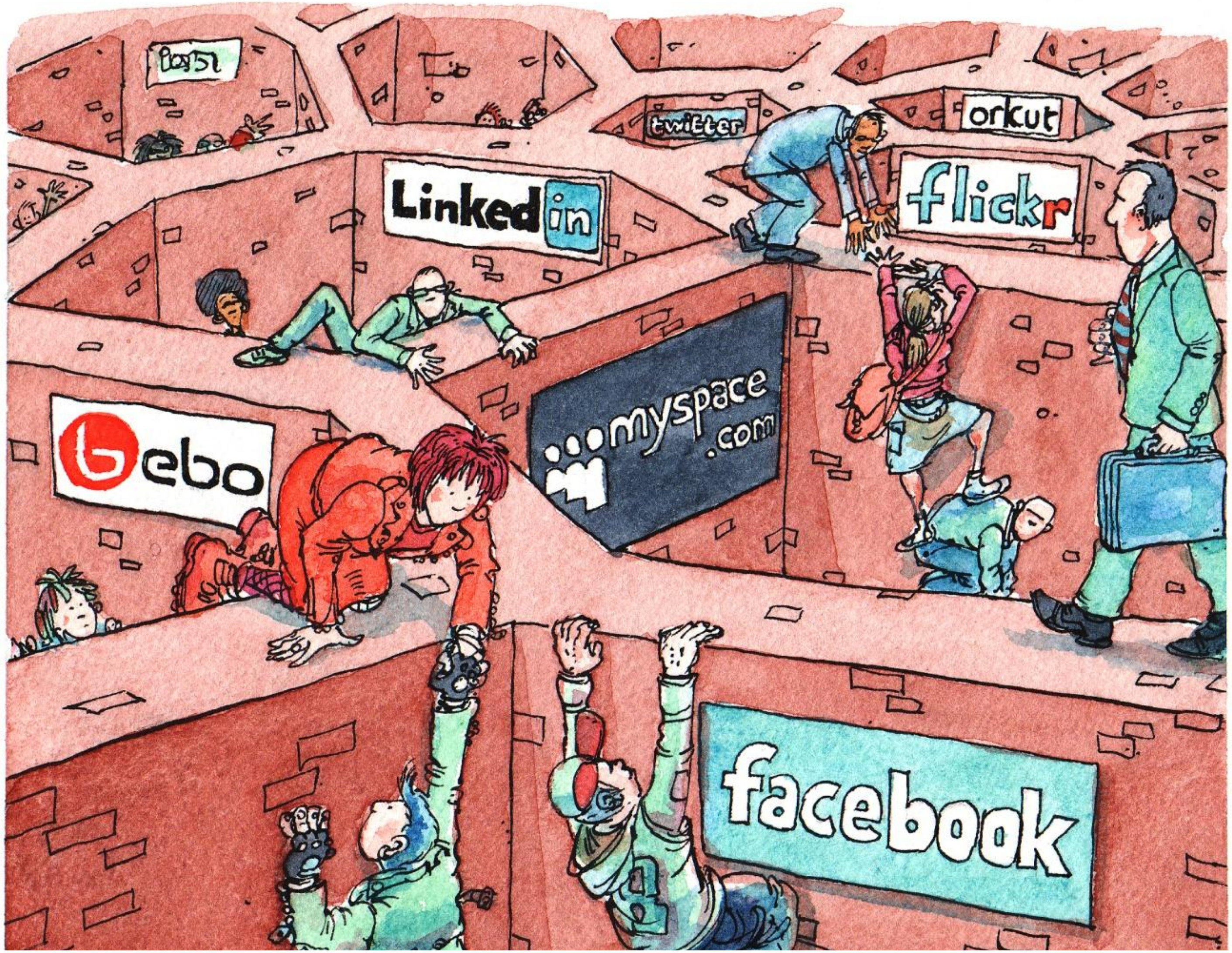
This data is spread across many systems that do not interoperate

- Query are hard: e.g., no global search
- Updates are hard: e.g., no global sync
- Some information is obsolete

Sometimes, you even forgot where

Your privacy is not even under your control

- Right of information: you should know when your data is copied/used
- Right of erasure: you should be able to delete some private data
- Right of objection: you should be able to refuse the disclosure to gvt of private data



Of course you are lost…
Any normal person would be in this jungle

Thesis: a holistic approach based on logic

Real data:	picture@Alice-iPhone(34434.jpg,date:...,from:..., …)
Annotations:	tag@delilicious.com(“wikipedia.org”, dictionary)
Localization:	where@Alice(pictures, Picasa/abiteboul)
	where@Alice(pictures, Alice-iPhone)
Access data:	access@Picasa/abiteboul(login:Alice, passwd:Alice)
Access rights:	right@Picasa/abiteboul(pictures,friends,read)
	group@picase/abiteboul(friends,bob)
Services:	search@google.com(“ICWE “,$X)
	addresse@pagesjaunes.fr(“John Doe”, Paris, $Y)

Etc.

Thesis detail

All this information forms a **distributed knowledge base** with

- Data
- Access control
- Keys
- Localization
- Time & provenance
- Services

Reasoning in this **distributed knowledge base** is used

- To answer queries
- To verify properties of the system such as enforcement of access control

Distributed logic base = distributed datalog

Why should you bother? Scenario

Alice query: get me recent pictures of Bob?

$X ← friends@Alice($Y), pictures@Y($Z), $Z.contains(Bob),
$Z.date<"01/01/2010"

What is going on:

- Find who are Alice's friends
- For one answer, say Sue, find where Sue keeps her pictures possibly using ontology mappings between Alice's schema and Sue's schema
- Check whether Alice has the right to see Sue's picture
- Convince whoever has this data that Alice has the right to get them …

Serious query processing/reasoning going on: data, localization, search, access rights, access keys, possibly data encryption/decryption

Distributed datalog revisited

The underlying model

Peer: Alice-iPhone, Picasa, facebook, AliceLaptop…

- Storage and processing capabilities
- Has a URI and can be sent query/update requests

Principal: Alice, AliceFriends, icweCommunity, databaseExperts

- Virtual so rely on peers for storage and processing
- Has an identity and can be authenticated (based on crypto protocol)

Peers and principals have relations and knowledge

- Alice states Bob is a friend = friends@Alice(Bob)
- album@Alice-iPhone, contacts@Alice-iPhone, calendar@Alice-iPhone...
- friends@Alice, where@Alice, access@Alice...
- friends@Alice($X) ← friends@bob($X), member@universityParis($X)

The underlying model

The principal Alice is virtual

- Where is her data? on some peers

External data in peers

- Knowledge about principals (storage for them), other peers (replication)
- facebook exports ‘Alice states Bob is a friend’
- Formally: use of reification
- exports@facebook(friend,Alice,Bob)

Query to Facebook

- $X ← exports@facebook(friend,Alice,$X)

Based on logical rules

Application of deductive datalog revisited: Access control and the Pastis system

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
SACLAY – ÎLE-DE-FRANCE

The Pastis system

Some knowledge stored on Alice's laptop

Base facts:	AlicePC exports "Georg is Professor at Oxford"
AC facts:	AlicePC exports "Bob canRead myPictures@Alice"
Localization	AlicePC exports "myPictures@Alice storedAt Sue"
Keys	AlicePC exports readKey@Bob

Accessing & updating information

Data

- Trees with references
- Collections (ala RSS feeds) represented as trees

Based on that one can locate and obtain information

Access rights

- Own – can also grant/revoke access rights
- Read
- Write
- Append/Remove from collections…
- Corresponding cryptographic keys

Enforcing access control & auditing

Time and provenance are also recorded

All statements are authenticated (by the author and the access right needed for the statement)

Data is possibly encrypted so that it may be stored on untrusted peers

What we do:

- We don't prevent you from misbehaving
- If you do, this shows
- As soon as you reach a honest peer, you can be caught

Reasoning

In the knowledge base

- To locate data and answer queries – datalog again not surprisingly
- To optimize queries

About strategies/systems

- To check whether peer strategies are sound (no leak) and complete (no denial of data/update)

Can be combined with beliefs and trust: e.g.,

Alice believes Paul stores her pictures

Datalog yes – But with lots of gadgets

Distribution: **Distributed datalog revisited**

Trees, service calls, intentional answers **Active XML**

Other aspects not discussed here

Time: Hellerstein's work; Dedalus

Negation: lots of works in the 90's Well-founded…

Non-safe variables in heads: Gottloeb's work; Datalog+-

- Needed to capture simple ontological reasoning

Trees and intentional data: Active XML

Active XML (see activeXML.net)

Based on Web standards:

XML + Web services + Xpath/Xquery

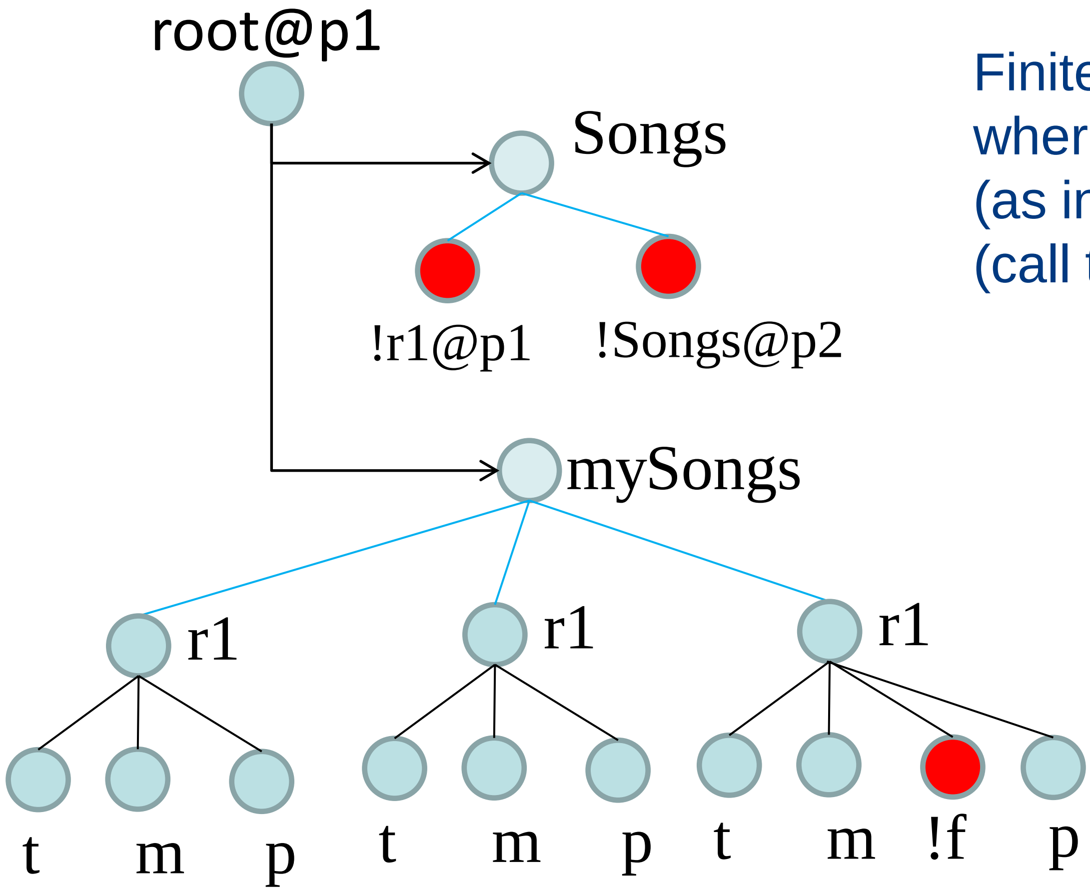
Simple idea

Exchange XML documents with embedded service calls

- Intentional data: get the data only when desired
- Dynamic data: If data sources change, the document changes
- Flexible data: adapt to the needs
- Function in push & pull mode; Sync and asynchronous

Embedding calls in data is an old idea in databases

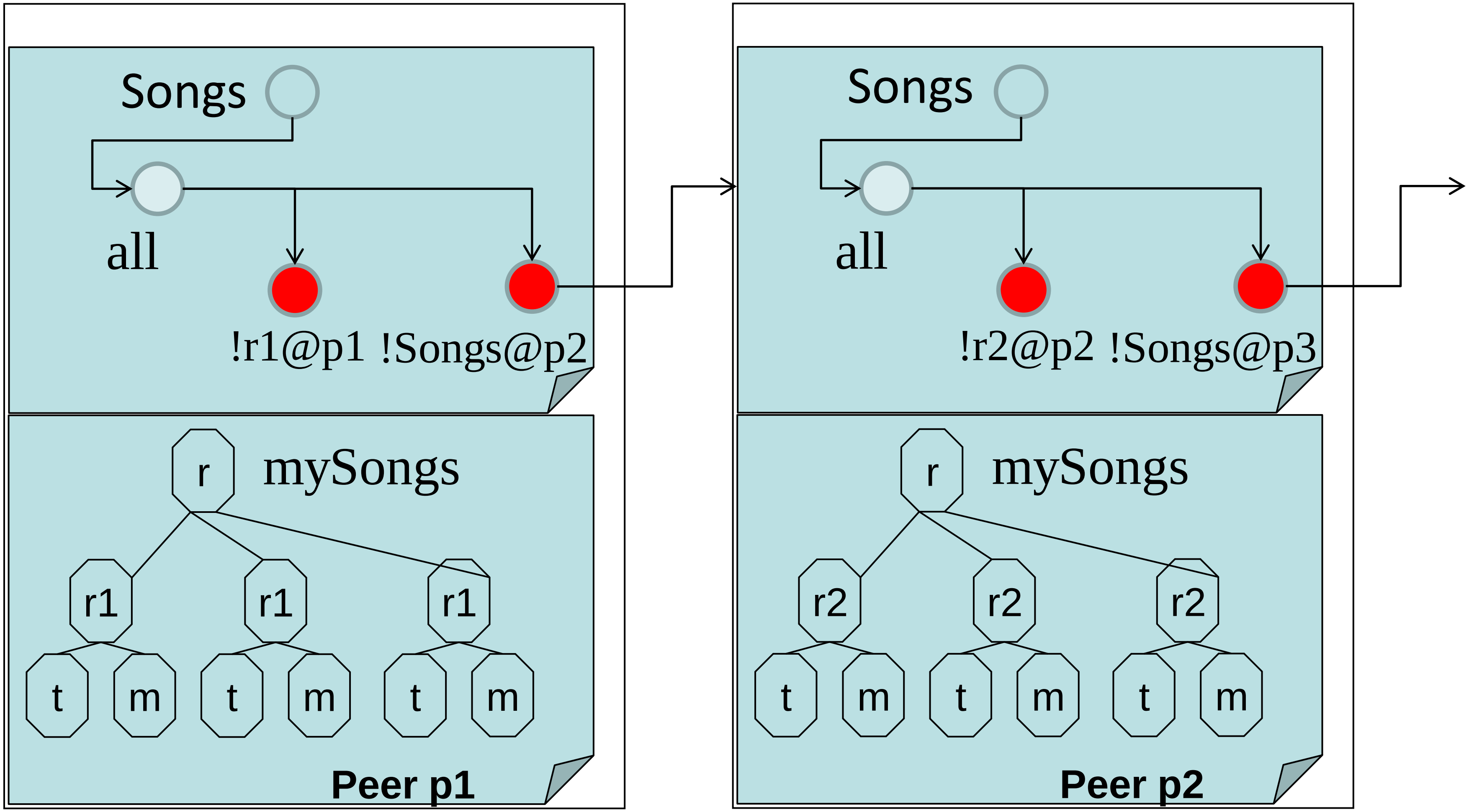
Active XML = Object database



XML & Web services

Finite labeled unordered trees where labels are tags, data (as in XML) or function calls (call to Web services)

ActiveXML:
XML documents with embedded service calls



This is distributed datalog over trees

Songs@p1(x,y) :- r1@p1(x,y)

Songs@p1(x,y) :- Songs@p2(x,y)

Songs@p2(x,y) :- r2@p2(x,y)

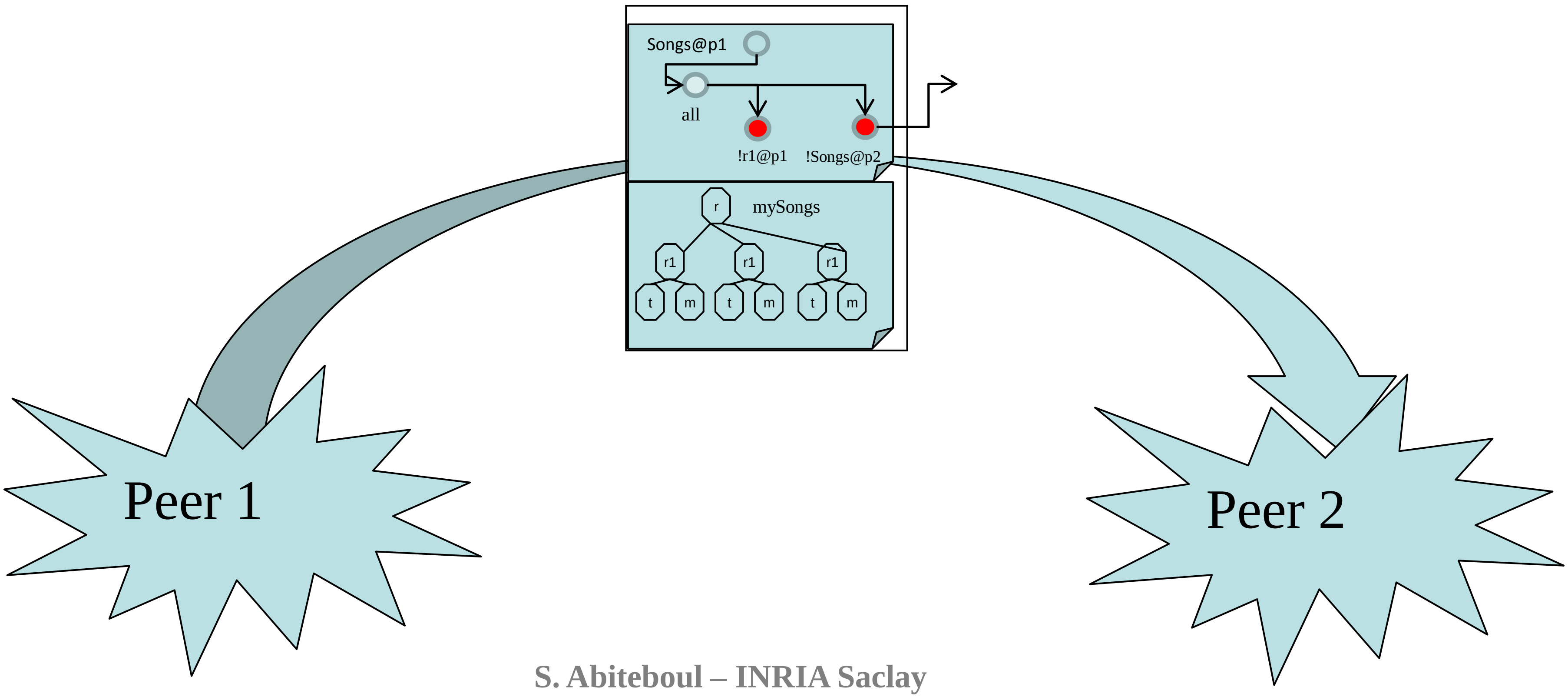
Songs@p2(x,y) :- Songs@p3(x,y)

Songs@p3(x,y) :- r3@p1(x,y)

Songs@p3(x,y) :- Songs@p1(x,y)

Songs@p1(“Carla Bruni”, x) :-

Moving data and logic around



The semantics of calls

When to activate the call?

- Explicit pull mode: active databases
- Implicit pull mode: deductive databases
- Push mode: query subscription

What to do with its result?

How long is the returned data valid?

What to send?

- *Phone number of the Prime Minister of France?*
- *Use whoswho.com then look in www.gouv.fr/phone*
- *Look for Fillon in www.gouv.fr/phone*
- *+33 1 56 00 00 07*

Active XML – cool idea – complex problems

Brings to a unique setting

distributed db,

deductive db,

active db,

stream data

warehousing & mediation

Is this unreasonable? Yes!

- And we have been working on it for several years
- And there are lots of problems left

Some works around AXML

The AXML system – open-source (on server, on smartphone)

The useful: Replication and query optimization

- How to evaluate a query efficiently by taking advantage of replication

The useful: Lazy query evaluation

- How to evaluate a query without calling all embedded services

The fun: Casting problem

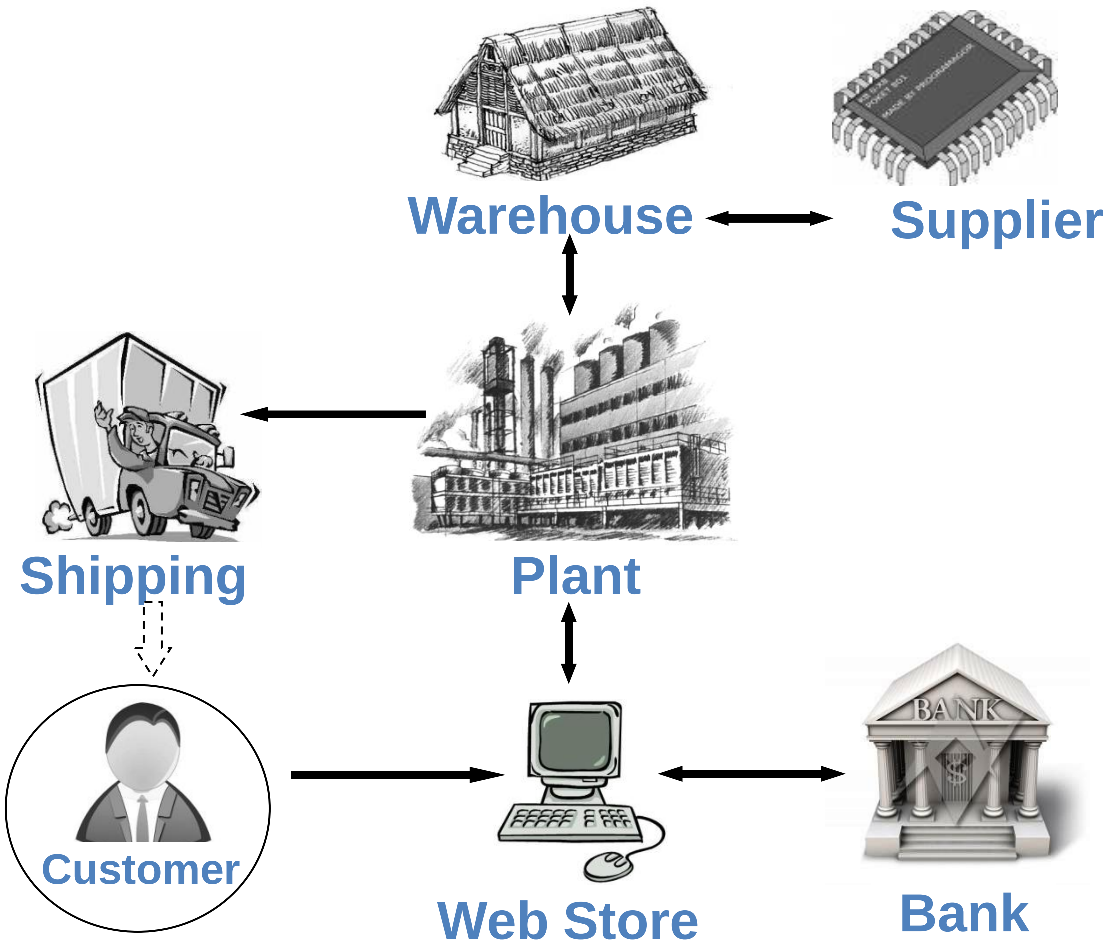
- Which functions to call to “match” a target type
- Active context-free games

The exotic: Diagnosis of communication systems

- The unfolding of the runs is described in AXML
- Datalog technology used for optimization

Verification: Guarded AXML

Example: Dell Supply Chain



Issues

More and more such Web systems

Challenges:

Verify the behavior of the system

Control the sequencing of the operations

A restricted model: guarded AXML

A datalog-style language so that we know what we are doing

Severe restrictions so that verification can be performed

Based on imposing constraints on call activation/return: guards

Constraints on data: DTD + tree pattern formulas

Focus: deciding whether a service S satisfies a Tree-LTL sentence

- Decidable for bounded services*:* no recursion
- Very high complexity – just a proof of feasibility
- Undecidable as soon as any of the syntactic restrictions are relaxed

Temporal formulas: Tree LTL

Boolean combinations of tree patterns & LTL operators

Syntax of Tree-LTL

φ :-pattern | φ and φ | φ or φ | not φ | **φ U φ | Xφ**

- pattern(X1,…,Xn) : all other variables are seen as existentially quantified
- **X: next U: until**
- Also **G: always? F: eventually. etc**

Tree-LTL sentence ∀φ

- All free variables are *quantified universally at the end*
- These are all the free variables from patterns

Example

Every webOrder is eventually completed (delivered or rejected)

$\forall$X [**G((T1(X) $\rightarrow$ F(T2(X) $\vee$ T3 (X)))]** where

T1(X): SYS [webOrder [Order-id [X]]]

T2(X): SYS [webOrder [Order-id [X] Delivered]]

T3(X) : SYS [webOrder [Order-id [X] Rejected]]

AXML Artifact = Data & Control

Concept introduced by IBM Research

[Nigam & Caswell 03, Hull & Su 07]

Data-centric workflows

- A process is described by a document (possibly moving in the enterprise)
- The behavior of an artifact is specified by some constraints on how this document should evolve

Vs. state-transition-based workflows

- Based on some form of state transition diagrams (BPEL, Petri,…)
- Mostly ignore data

webOrder id=7787780
Customer
    Name: John Doe
    Address: Sèvres
Product: *committed*
    Ref: PC 456
Factory: Milano
Parts: *waiting*
orderDate: 2009/07/24
Site: http:// d555.com
Payment: *done*
    Bank-account …
Delivery: *not-active*

AXML Artifacts move on the Web

In webStore

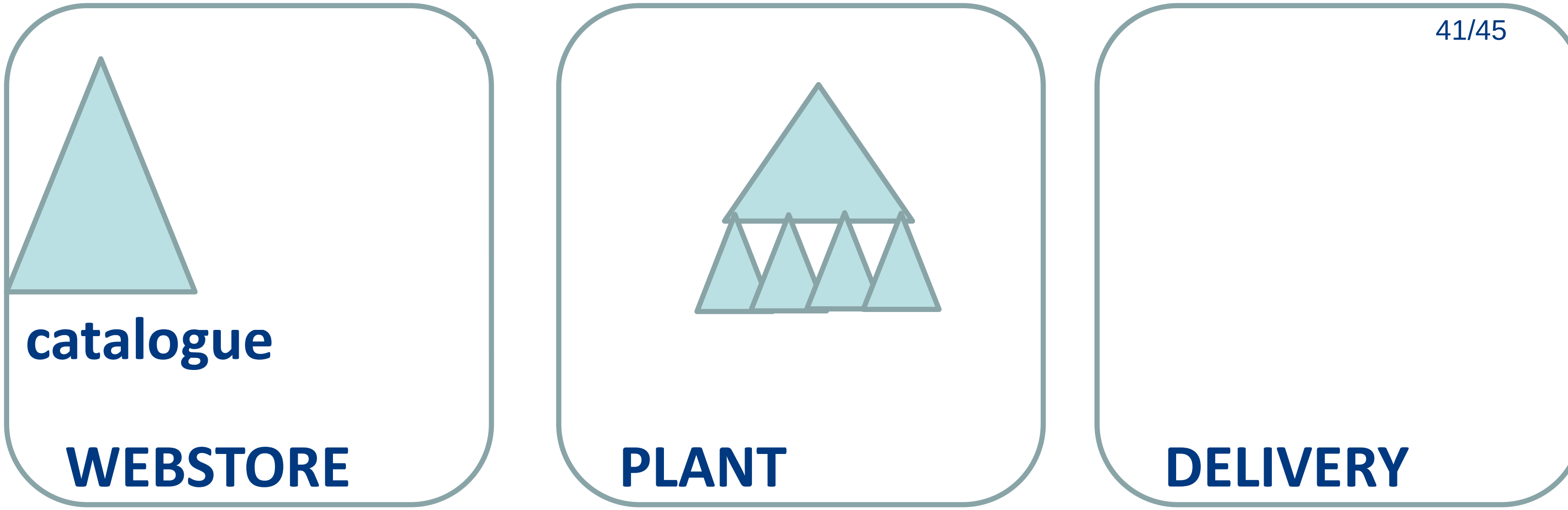
webOrder id=7787780
Customer
Name: John Doe
Address: Sèvres
Order selection: **on-going**
Ref: PC 456
Factory: *undecided*
Parts: *not-active*
orderDate: 2009/07/24
Site: http://d555.com
Payment: *pending*
Delivery: *not-active*

In plant

webOrder id=7787780
Customer
Name: John Doe
Address: Sèvres
Order selection : *committed*
Ref: PC 456
Factory: Milano
Parts: ***on-going***
orderDate: 2009/07/24
Site: http:// d555.com
Payment: *done*
Bank-account …
Delivery: *not-active*

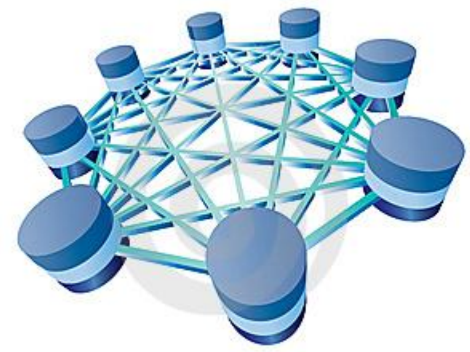
In delivery

webOrder id=7787780
Customer
Name: John Doe
Address: Sèvres
Order selection : *committed*
Ref: PC 456
Factory: Milano
Parts: *done*
orderDate: 2009/07/24
Site: http:// d555.com
Payment: *done*
Bank-account: CEIF-4457889
Delivery: ***on-going***
Address: Orsay



AXML
Artifact model

Sequencing of operations

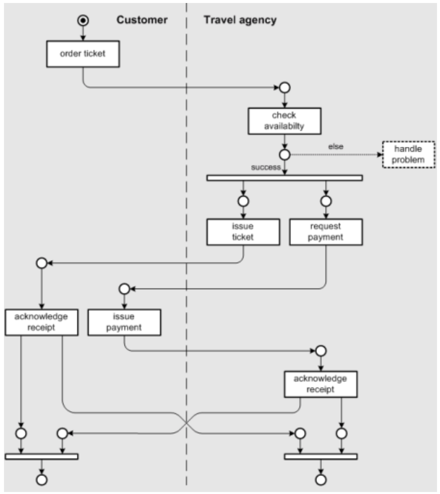


Different ways of expressing sequencing of tasks

- Guards: preconditions for function calls
- Transition-based diagrams
- Formulas in temporal logic

Study how they can simulate each other using some “scratch paper”

Data & workflow



Conclusion

Web data management

Lots of problems to investigate

Lots of challenges

Lots of fun

Major challenge for Industry: build systems that we can control, where we can notably control privacy

Major challenge for Academia: be able to teach properly a course on Web data management

Deductive databases inside
Object databases inside

Good ideas take always more time than we thought to win

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
SACLAY – ÎLE-DE-FRANCE