

Semantic Web

Software View vs. Math View

Guest lecture @ Dickinson College, 2016-10-25

Dr. Ora Lassila

Technology Architect & Senior Fellow
Pegasystems, Inc.

(former) Board Member & Fellow
W3C

Pega[®] 7

THE POWER
TO SIMPLIFY™



Who am I...?

Current:

- Technology Architect, Pegasystems, Inc.

Earlier:

- many positions at Nokia (1996-2013)
- visiting scientist, MIT CSAIL/LCS
- project manager, Carnegie Mellon Univ.
- research scientist, Helsinki Univ. of Tech.
- startup founder, etc.

Education:

- Ph.D in CS, Helsinki Univ. of Tech.

Notable (or Dubious) Achievements:

- co-author of the W3C RDF specification
- co-author of the seminal Semantic Web article (20k+ citations)
- co-author of Semantic Web Science Association's "10-year Award"
- designed and wrote software for NASA's "Deep Space 1" probe
- grand prize @ Usenix "Obfuscated C Code Contest"
- father of two ballerinas (one studies here at CPYB)

Game plan

1. Idea and dream
2. Technological underpinnings
3. Specific technologies & applications
4. Q&A

Idea and dream

WWW was built for humans, and human interpretation is needed to give information some meaning.

This problem is not limited to the Web. We want to automate things for the user, and not make users worry so much about the logistics of all that.

How could systems (services, applications, machines, ...) share data and make use of data created by others? How could they do it if their designers didn't anticipate this?

Our current fascination with “apps” only makes this situation more difficult.

Serendipity defines the Semantic Web

Serendipity in...

Interoperability: Is it possible to interoperate with systems and services we knew nothing about at design time?

Reuse: When information has accessible semantics, this is a lot easier...

Integration: Can information from various independent sources be combined?

To realize this, more demands are put on **data sharing**...

How to share data between systems

Make it an engineering problem:

- Pick two systems, design a way how they can exchange data

How to share data between systems

~~Make it an engineering problem:~~

- ~~• Pick two systems, design a way how they can exchange data~~

DOES NOT SCALE!

How to share data between systems

~~Make it an engineering problem:~~

- ~~• Pick two systems, design a way how they can exchange data~~

Make it a standardization problem:

- Decide in advance what to say and what it means
- Base future designs on this specification

DOES NOT SCALE!

How to share data between systems

~~Make it an engineering problem:~~

- ~~• Pick two systems, design a way how they can exchange data~~

~~Make it a standardization problem:~~

- ~~• Decide in advance what to say and what it means~~
- ~~• Base future designs on this specification~~

DOES NOT SCALE!

***LIMITING AND NOT
"FUTURE-PROOF"!***

How to share data between systems

~~Make it an engineering problem:~~

- ~~• Pick two systems, design a way how they can exchange data~~

~~Make it a standardization problem:~~

- ~~• Decide in advance what to say and what it means~~
- ~~• Base future designs on this specification~~

Use Semantic Web technologies:

- Make data semantics **accessible** and **declarative**
- Make data **self-describing**
- Standardize **how** to say things, not **what** to say or what it means (“delayed semantic commitment”)

DOES NOT SCALE!

***LIMITING AND NOT
“FUTURE-PROOF”!***

Now you will say: “I will just use JSON” (or XML...)

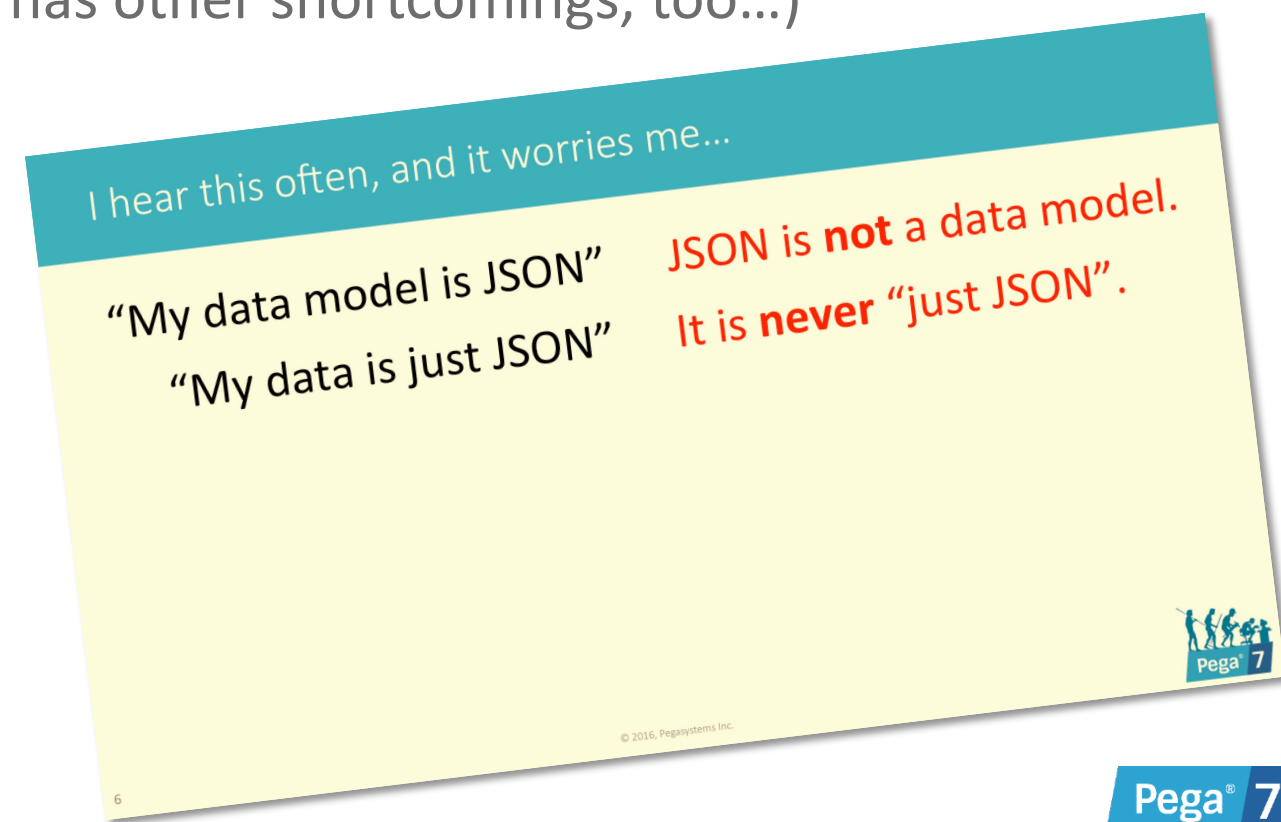
Wrong answer, you just flunked the class!

JSON is a serialization syntax for a (very) simple metamodel

- it establishes almost no semantics (and has other shortcomings, too...)

Any interpretation of JSON data is “layered over” the standard

- “black box”, procedural semantics
- i.e., this is not JSON semantics, this is **your** semantics



What establishes (data) semantics?

1. Relationship of data to (accessible & declarative) definitions of data types
2. Relationship of data to some other data
3. Some (procedural) software that “hard-wires” how to process some particular kind of data

All semantics is grounded in the above three

- note that #1 is recursive
- the less you have #3, the better
(and yet, today, most of semantics is captured via #3)

How do we achieve #1 and #2...?

Possible answer: Knowledge Representation

Representing information in a form that allows **reasoning**

- much of recent work on KR has focused on the use of mathematical logic as a means of description and representation (and reasoning thereof)

Expressiveness vs. practical implementations

- FOPC often considered a baseline for expressiveness, but limited systems of logic are better in terms of decidability & computability → Description Logics

Ontologies (as rich models of concepts and relations)

Historical background

- Leibniz, Hilbert, Church, Turing: decidability, “*Entscheidungsproblem*”
- Husserl: Ontology

Mathematical foundations

Logic

Sets

Graphs

Graphs... why graphs?

The universality of graphs (as a data structure) underlies Semantic Web data.

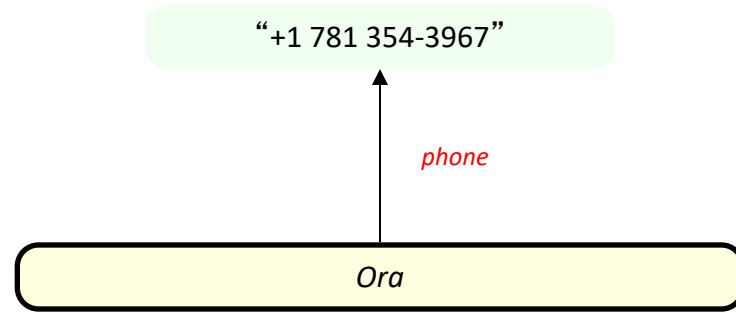
Software view:

- processing of Semantic Web representations can rely on what we already know about the manipulation of graphs
- it could be argued that (much of) computer science in general reduces to graph theory and algorithms on graphs (no, really)

Math view:

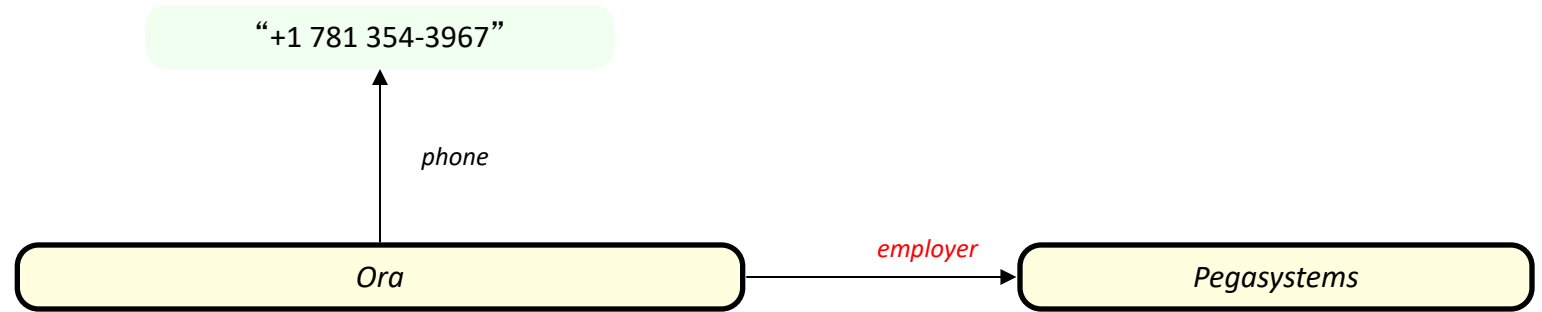
- origins of KR are in graph representations (e.g., semantic networks, frames)
- KR community has established formal, logic-based foundation for these graphs

Graphs: a natural way to represent the world



“Ora’s **phone** is +1 781 354-3967”

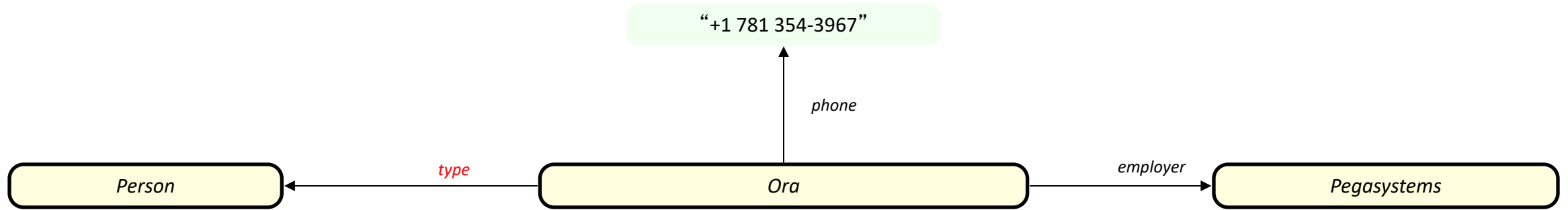
Graphs: a natural way to represent the world



“Ora’s **phone** is +1 781 354-3967”

“Ora’s **employer** is Pegasystems”

Graphs & logic



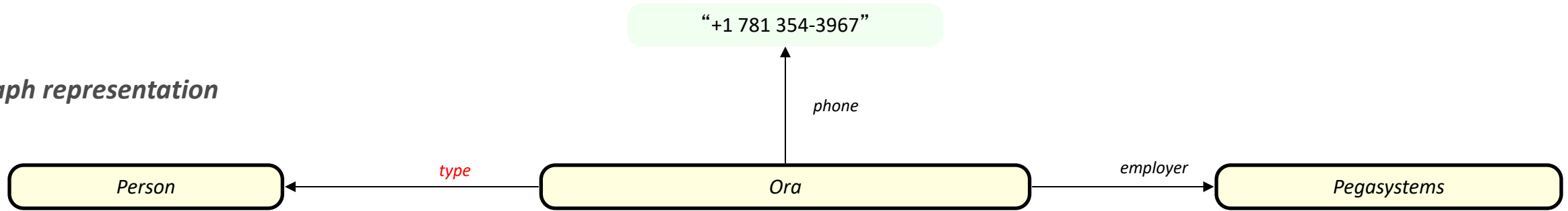
“Ora’s **phone** is +1 781 354-3967”

“Ora’s **employer** is Pegasystems”

“Ora is of **type** Person”

Graphs & logic

graph representation



assertions in "human language"

"Ora's **phone** is +1 781 354-3967"

"Ora's **employer** is Pegasystems"

"Ora is of **type** Person"

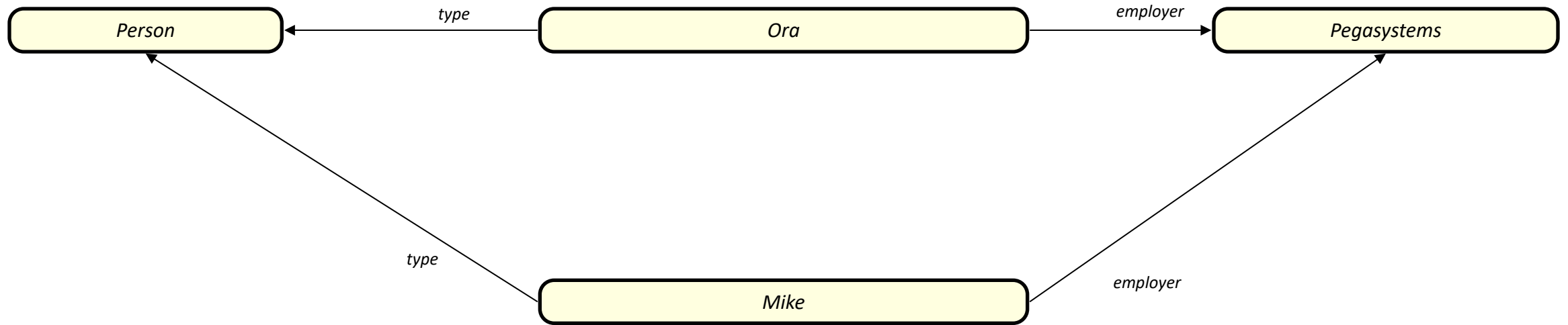
assertions using logic and set theory

phone(Ora, "+1 781 354-3967")

employer(Ora, Pegasystems)

Ora ∈ Person

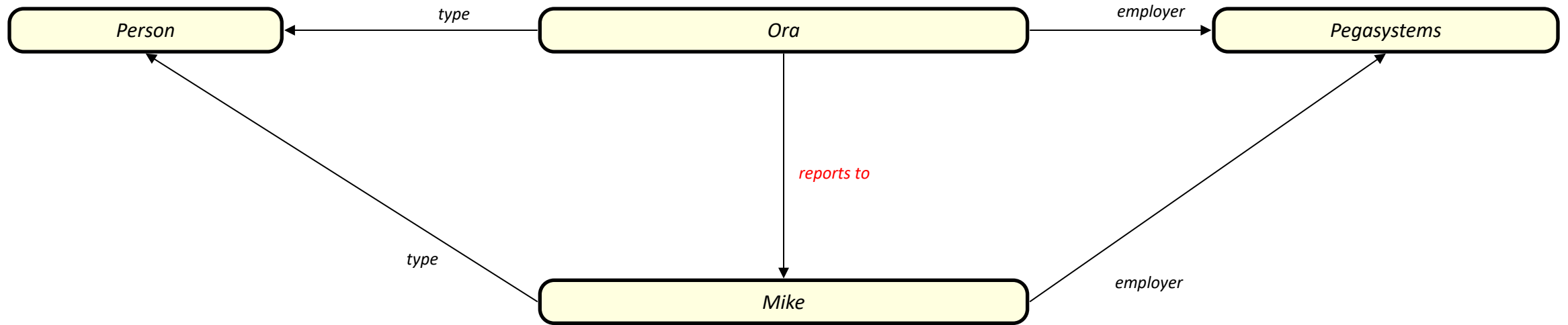
Let's add more people



“Mike works for Pegasystems”

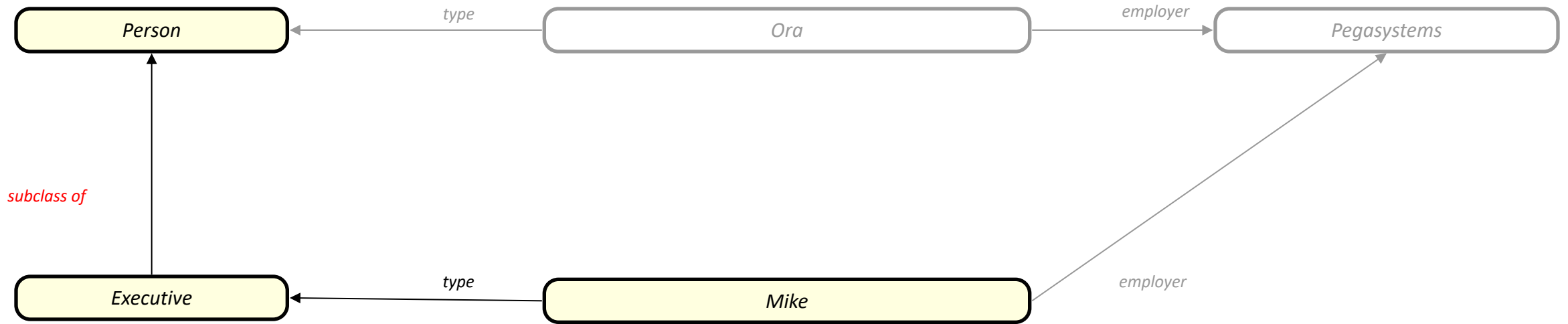
“Mike is of type Person”

Social and organizational relations are easy...



“Ora reports to Mike”

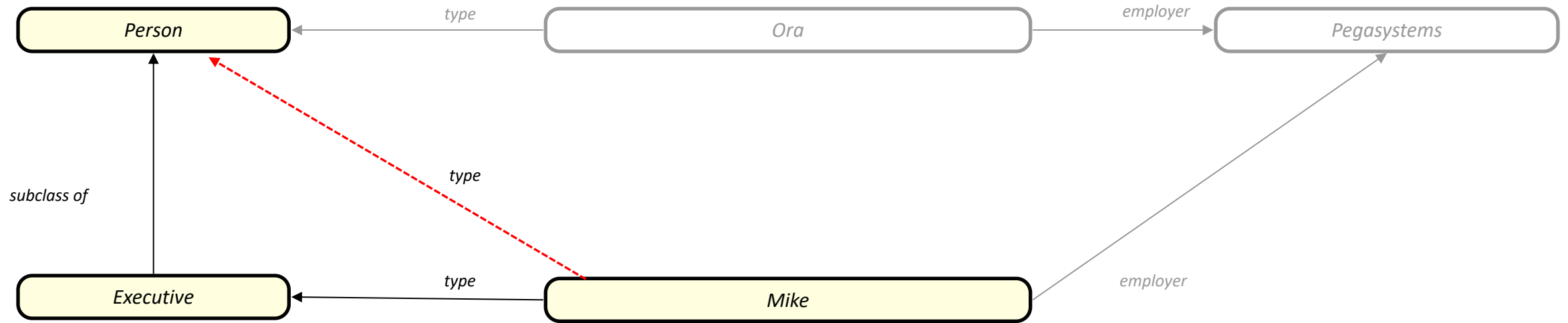
Polymorphism



“Executive is a **subclass of** Person”

$\text{Executive} \subset \text{Person}$

Polymorphism and reasoning



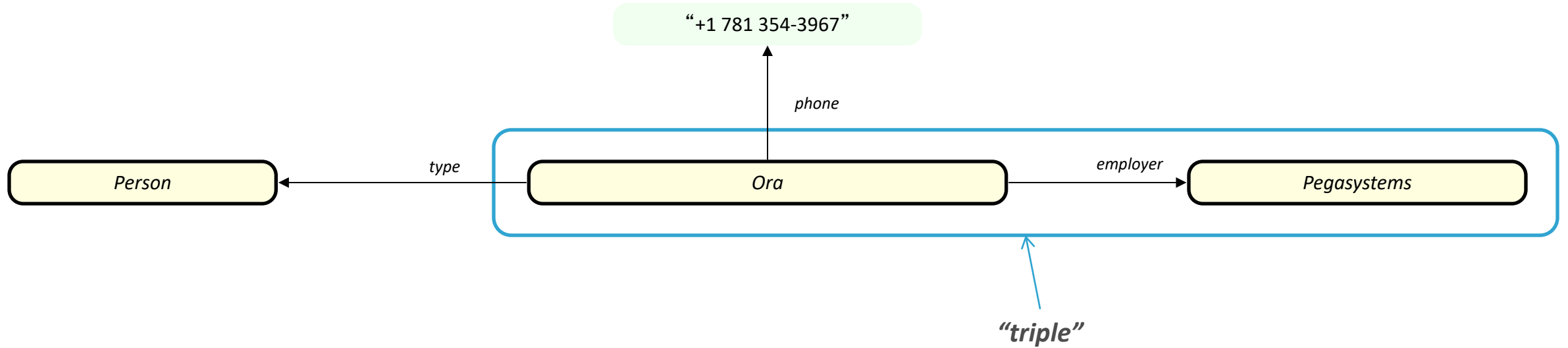
“Executive is a **subclass of** Person”

$\text{Executive} \subset \text{Person}$

We can **infer** that Mike is also of type Person

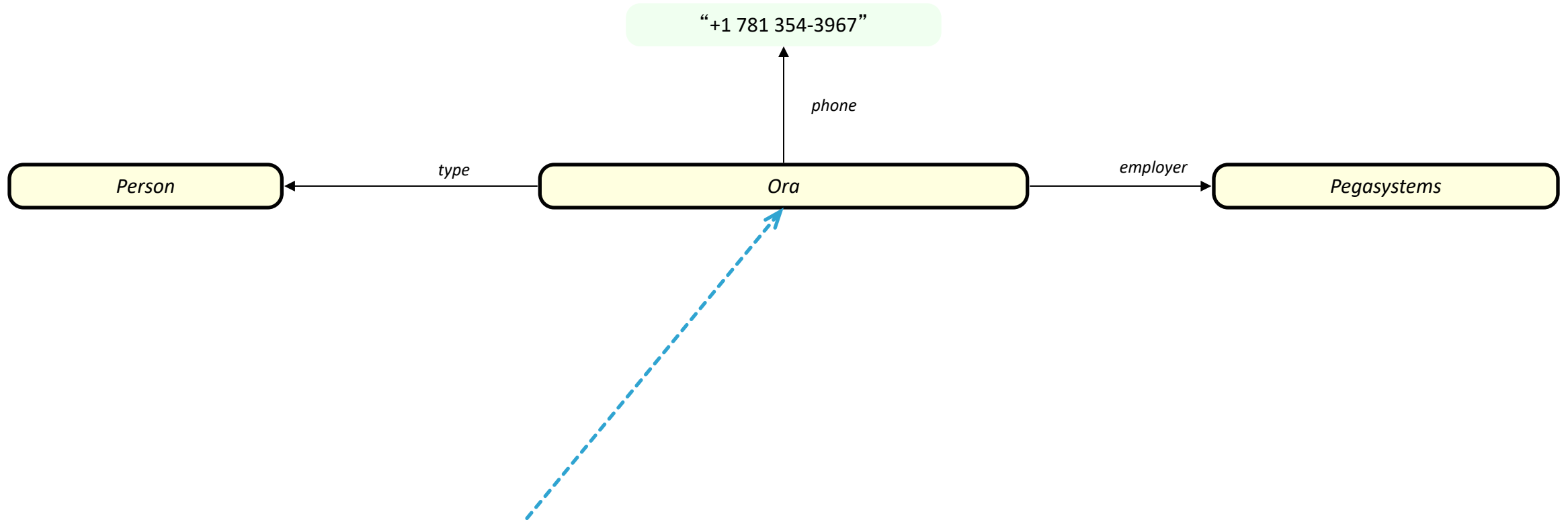
$\forall x: x \in A \ \& \ A \subset B \Rightarrow x \in B$

RDF as a representation for graphs



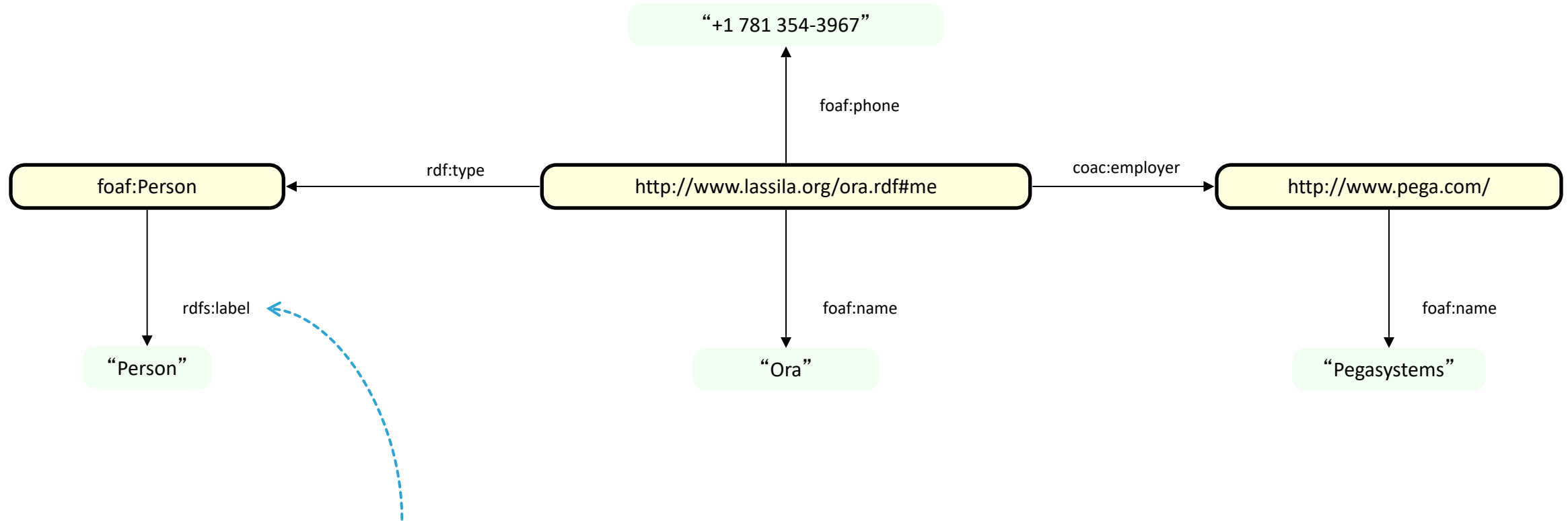
Much of the definition and processing of RDF thinks of graphs as collections of source/edge/sink tuples. In RDF parlance, the elements of these tuples are called “subject”, “predicate” and “object”.

RDF as a representation for graphs



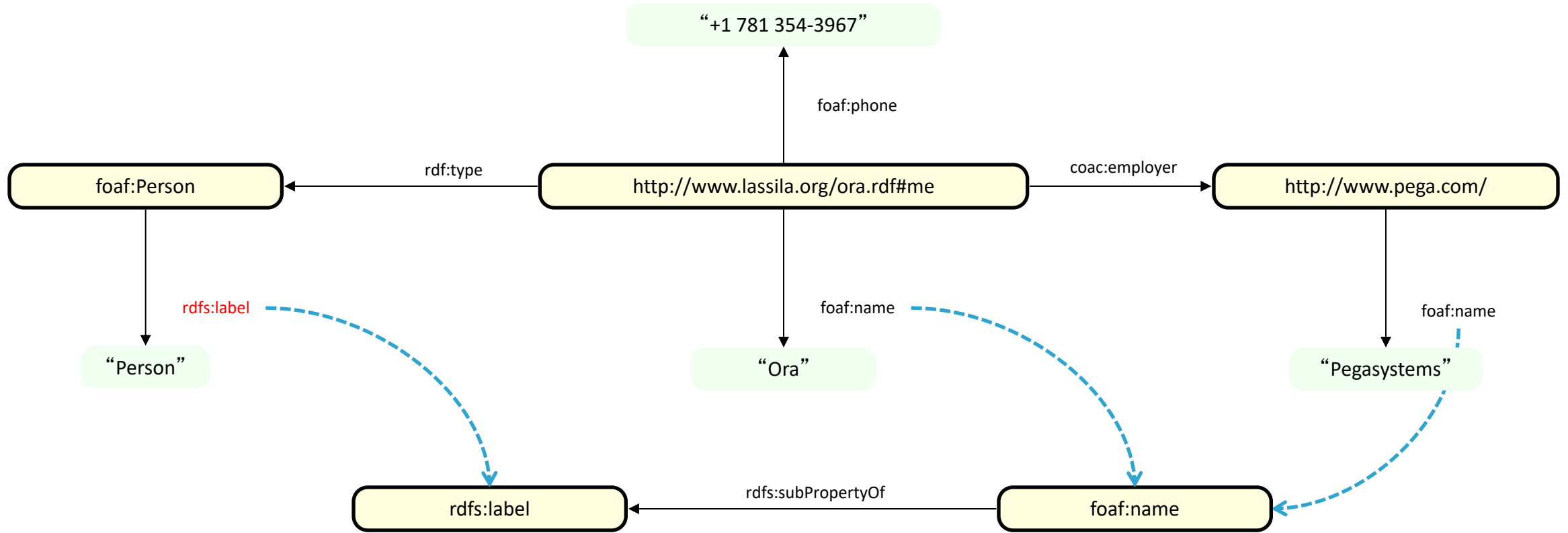
This is merely a “human-readable” name

Nodes and edges are named using URIs



Note: (as an example) “**rdfs:label**” is the XML QName for the URI “**http://www.w3.org/2000/01/rdf-schema#label**”

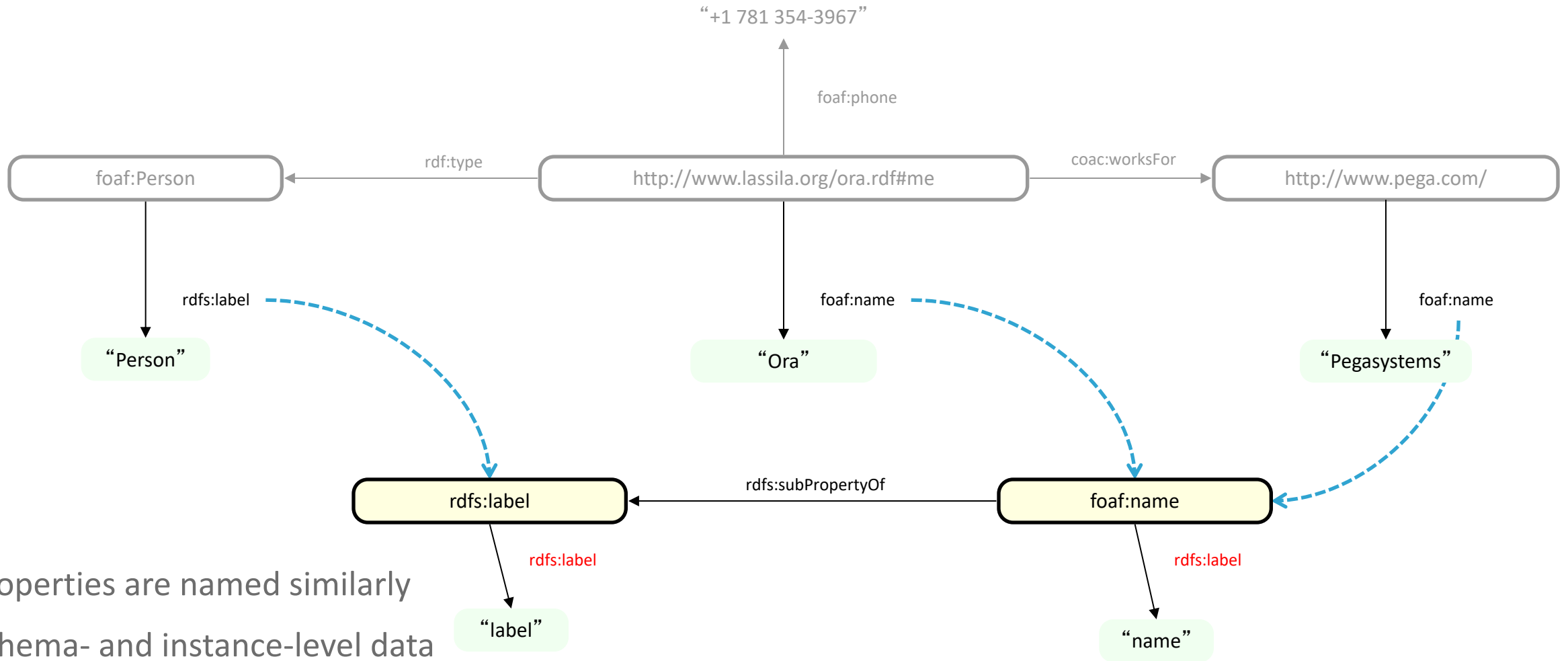
Properties (= edges) correspond to special nodes in the graph



Human-readable form of a node is found by looking for **rdfs:label** properties

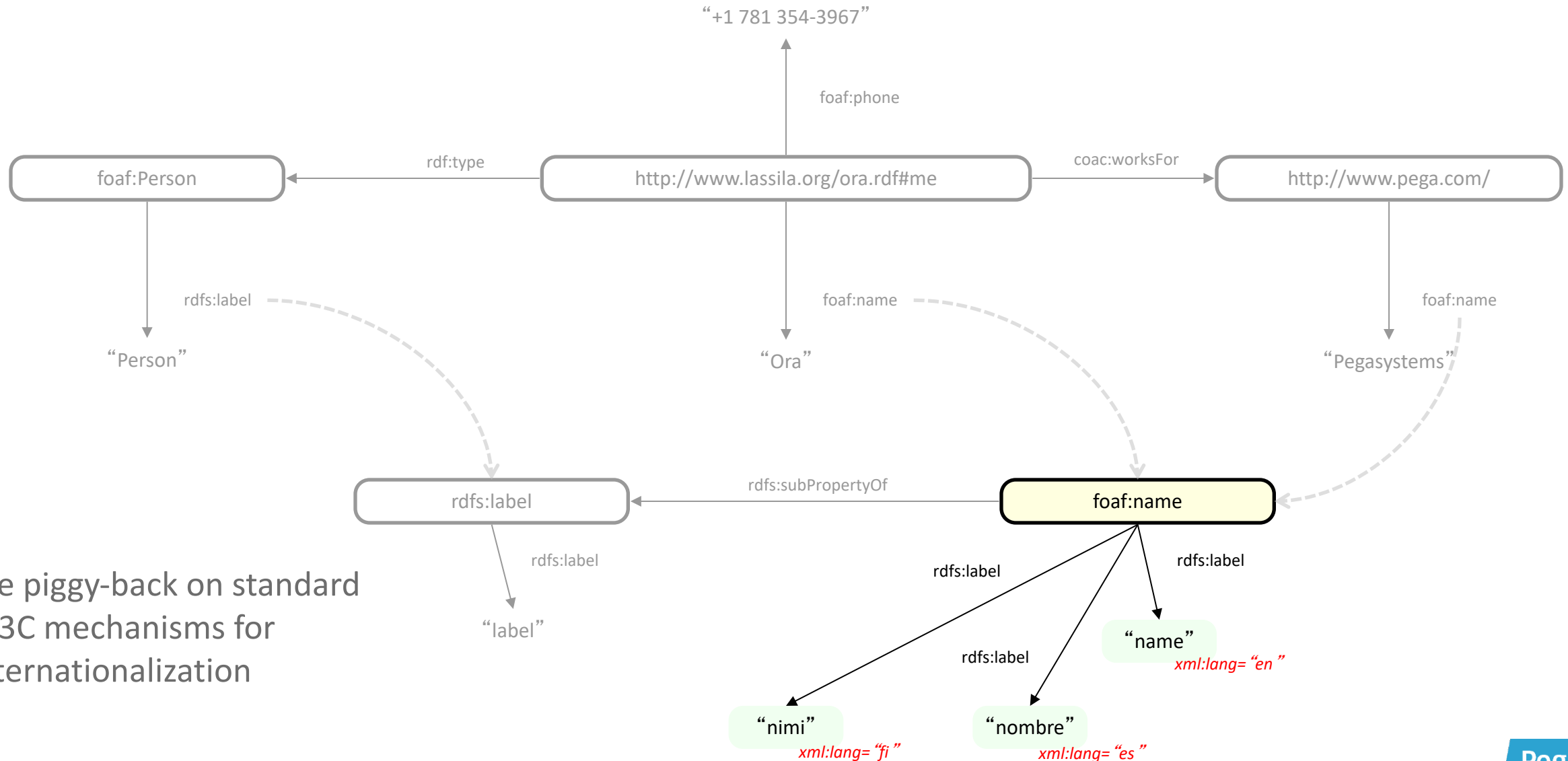
A reasoner will find all **sub-properties** of **rdfs:label** as well

All of this is recursive...



Properties are named similarly
Schema- and instance-level data
use the same representation

Multiple, alternative human-readable names are possible



We piggy-back on standard W3C mechanisms for internationalization

Basic Semantic Web technologies (from W3C)

RDF + RDFS

- simple ontological language
- data is a graph
- some inferential semantics
- specification gives model-theoretical semantics (also axiomatic semantics exist)
- XML-based syntax, JSON-based syntax, etc.

SPARQL

- query language for RDF data
- treats the graph as a collection of relations (tuples)
- has a formally defined query algebra
- supports federated queries across multiple “SPARQL endpoints”

Basic Semantic Web technologies (from W3C)

OWL

- family of more expressive ontological languages
- different systems of logic are used (FOPC, various forms of description logics)
- extends the “ontological vocabulary” of RDF

RIF

- vocabulary and basic semantics for exchanging rules between rule-based systems
- logic-based rules and “production rules”

Typical approach is to use RDFS with some features of OWL thrown in...

Semantic Web software

To develop an application that makes use of Semantic Web technologies, you typically need to be able to

- store and manage data → graph databases, “triple stores”
- extract data from other documents and databases → parsers, mappers
- reason over data → reasoners, inference engines

Both proprietary and open source options exist for the above

- W3C has a list: <https://www.w3.org/2001/sw/wiki/Tools>

In my opinion, adopting these technologies and tools is still difficult

- a particular challenge is to match UI capabilities with the richness of representation in the underlying data

Applications of the Semantic Web

Semantic Web technologies are particularly well suited to problems where

- data needs to be shared and reused
- complexity of data models is high
- data is heterogeneous (e.g., many different models)

We can see successful applications in fields such as

- pharmaceutical and life science research (complex data that needs to be shared)
- e-government (many heterogeneous data sets that people want to explore)
- digital libraries, museum collections (integration thereof)

“Linked Data” = Semantic Web with the inferential component removed

Conclusions, last words...

Current way of designing, building and delivering information technology to end users is **broken**

- information is **isolated**, information space is **fragmented**

Semantic Web technologies can be used to address some of the problems

- however, covering “a lot of ground” is difficult

We should **focus on data**, understanding that various means to process it come and go

- make it possible to **share** data, others will come up with new ways of using your data

Homework: what about **business models** for all this?

Difficult message? Are there dark clouds?

Any specific problem (typically) has a specific solution that does not require Semantic Web technologies

Q: Why then is the Semantic Web so attractive?

A: For future-proofing

**Semantic Web can be a solution to those problems
and situations that we are yet to define.**

Questions (and maybe even answers...)

More information:

- <https://www.w3.org/standards/semanticweb/>
- <https://www.w3.org/2013/data/>
- <http://dl.kr.org/>

To contact me:

- ora.lassila@pega.com
- Twitter: [@gotsemantics](https://twitter.com/gotsemantics)
- <http://www.lassila.org/>