The *SoLaSoTe* ontology for software languages & technologies

Ralf Lämmel, Martin Leinberger, and Andrei Varanovich
Software Languages Team
University of Koblenz-Landau, Germany
http://softlang.wikidot.com

Ontologies in software engineering — some data points

Ontologies in software engineering

```
@article{GasevicGTW10,
  author    = {Dragan Gasevic and
               Giancarlo Guizzardi and
               Kuldar Taveter and
               Gerd Wagner},
  title     = {Vocabularies, ontologies, and rules for enterprise and business
               process modeling and management},
  journal   = {Inf. Syst.},
  volume    = {35},
  number    = {4},
  year      = {2010},
  pages     = {375-378},
}
```

Ontologies in software engineering

```
@inproceedings{SouzaFV13,
  author    = {Erica F. Souza and
               Ricardo de Almeida Falbo and
               N. L. Vijaykumar},
  title     = {Ontologies in Software Testing: A Systematic Literature
               Review},
  booktitle = {ONTOBRAS},
  publisher = {CEUR-WS.org},
  series    = {CEUR Workshop Proceedings},
  volume    = {1041},
  year      = {2013},
  pages     = {71-82},
}
```

Ontologies in software engineering

```
@inproceedings{CarvalhoAG14,
  author    = {Victorio Albani de Carvalho and
               Jo{\~a}o Paulo A. Almeida and
               Giancarlo Guizzardi},
  title     = {Using Reference Domain Ontologies to Define the Real-World
               Semantics of Domain-Specific Languages},
  booktitle = {CAiSE},
  year      = {2014},
  pages     = {488-502},
  publisher = {Springer},
  series    = {LNCS},
  volume    = {8484},
}
```

Ontologies in software engineering

```
@inproceedings{BarcellosF13,
  author    = {Monalessa Perini Barcellos and
               Ricardo de Almeida Falbo},
  title     = {A software measurement task ontology},
  booktitle = {SAC},
  publisher = {ACM},
  year      = {2013},
  pages     = {311-318},
}
```

Ontologies in software engineering

```
@article{WongthongthamCDS09,
  author    = {Pornpit Wongthongtham and
               Elizabeth Chang and
               Tharam S. Dillon and
               Ian Sommerville},
  title     = {Development of a Software Engineering Ontology for Multisite
               Software Development},
  journal   = {IEEE Trans. Knowl. Data Eng.},
  volume    = {21},
  number    = {8},
  year      = {2009},
  pages     = {1205-1217},
}
```

Ontologies in software engineering

```
@inproceedings{DobsonLS05,
  author    = {Glen Dobson and
               Russell Lock and
               Ian Sommerville},
  title     = {QoSOnt: a QoS Ontology for Service-Centric Systems},
  booktitle = {EUROMICRO-SEAA},
  year      = {2005},
  pages     = {80-87},
  publisher = {IEEE Computer Society},
}
```

Ontologies in software engineering

```
@article{Henderson-SellersGML14,
  author    = {Brian Henderson-Sellers and
               Cesar Gonzalez-Perez and
               Tom McBride and
               Graham Low},
  title     = {An ontology for ISO software engineering standards: 1) Creating
               the infrastructure},
  journal   = {Computer Standards {\&} Interfaces},
  volume    = {36},
  number    = {3},
  year      = {2014},
  pages     = {563-576},
}
```

Ontologies in software engineering

```
@inproceedings{WongthongthamCDS05,
  author    = {Pornpit Wongthongtham and
               Elizabeth Chang and
               Tharam S. Dillon and
               Ian Sommerville},
  title     = {Software Engineering Ontologies and Their Implementation},
  booktitle = {IASTED Conf. on Software Engineering},
  publisher = {IASTED/ACTA Press},
  year      = {2005},
  pages     = {208-213},
}
```

Ontologies in software engineering

```
@incollection{RuizH06,
  author = "Francisco Ruiz, Jos\'e R. Hilera",
  title = "Ontologies for Software Engineering and Software Technology",
  year = "2006",
  pages = "49-102",
  booktitle = "Using Ontologies in Software Engineering and Technology",
  publisher = "Springer"
}
```

Ontologies in software engineering

```
@inproceedings{Ahmed08,
  author    = {Emdad Ahmed},
  title     = {Use of Ontologies in Software Engineering},
  booktitle = {SEDE},
  year      = {2008},
  publisher = {ISCA},
  pages     = {145-150},
  bibsource = {DBLP, http://dblp.uni-trier.de}
}
```

The *SoLaSoTe* ontology for software languages and technologies

SoLaSoTe's cause: **Knowledge representation**

- **Classification** of languages and technologies as well as related concepts.
- **Dependencies** between languages and technologies.
- **Concept-based characterization** of languages and technologies.
- **Links to existing knowledge resources** for languages and technologies.
- **Traceability** for language and technology usage **in shared software systems**.

http://101companies.org/wiki/Contribution:simplejdbc

Metadata

- this *developedBy* Contributor:DerDackel
- this *implements* Feature:Cut
- this *implements* Feature:Depth
- this *implements* Feature:Total
- this *uses* Language:Java
- this *uses* Language:SQL
- this *uses* Technology:Eclipse
- this *uses* Technology:H2
- this *uses* Technology:JDBC

What kind of ontology?

- Domain ontology
- Task ontology
- Application ontology
- Generic ontology

SoLaSoTe's perceived benefits

- **Unambiguous terminology** in the "domain" of languages and technologies.
- Identification of **commonalities and differences** of entities in ditto domain.
- Systematic **demonstration** of languages and technologies.
- **Integration** of otherwise scattered knowledge resources.

Querying SoLaSoTe to „infer“ knowledge

Paradigm-specific concepts Given a small set of programming paradigms, find the concepts that appear to be (more or less) uniquely associated with each paradigm—by means of collecting concepts being mentioned in the documentation of contributions, which are using programming languages of the different paradigms.

Simple baseline implementation Find the contribution that uses a given language and exercises a given concept such that there is no other contribution with less features, languages, technologies, and concepts involved.

Knowledge holder shortage Identify languages and technologies that are used infrequently by contributions without a proportional frequency of contributors who appear to be knowledgeable for these languages and technologies.

The *SoLaSoTe* schema

Top-level classification of entities

– Entity	*Everything in the scope of the ontology*
• Language	*Software languages such as Java or XML*
• Technology	*Software technologies such as JUnit or Eclipse*
• Concept	*Software concepts such as Visitor or Unit testing*
• Feature	*Features of 101's imaginary system*
• Contribution	*Implementations of 101's imaginary system*
• Contributor	*Contributors of code and documentation*
• Theme	*Containers of related contributions*
• Vocabulary	*Containers of domain-specific terms*
• Resource	*External resources such as standards and specifications*

There are a few „less important“ types.

Semantic properties grouped by subject entity

Entity		
instanceOf	Entity	*An instance/type relationship*
isA	Entity	*A specialization relationship*
partOf	Entity	*A whole-part relationship*
dependsOn	Entity	*Dependence relationship*
mentions	Entity	*Nonspecific reference in documentation*
sameAs	URL	*Equivalence relative to external resource*
similarTo	URL	*Similarity relative to external resource*
linksTo	URL	*Nonspecific reference to external resource*
documentedBy	Contributor	Authorship of documentation
memberOf	Vocabulary	*Membership in vocabulary of terms*
Contribution		
uses	Language	*Language usage*
uses	Technology	*Technology usage*
uses	Concept	*Concept usage*
implements	Feature	*Feature implementation*
developedBy	Contributor	*Developer of contribution*
reviewedBy	Contributor	*Reviewer of contribution*
memberOf	Theme	*Membership in theme of contributions*
basedOn	Contribution	*Indication of reuse*
varies	Contribution	*Indication of variation*
moreComplexThan	Contribution	*Indication of complexity*
Resource		
describes	Language	Language definitions, et al.
describes	Technology	API specifications, et al.
describes	Concept	Textbook, white papers, et al.
Technology		
uses	Language	*Language usage*
uses	Technology	*Technology usage*
uses	Concept	*Concept usage*
implements	Language	*Parsers, compilers, interpreters, et al.*
implements	Resource	*Compliance with a standard, et al.*
supports	Concept	*Support of a protocol, et al.*

Semantic properties grouped by subject entity

Entity		
instanceOf	Entity	*An instance/type relationship*
isA	Entity	*A specialization relationship*
partOf	Entity	*A whole-part relationship*
dependsOn	Entity	*Dependence relationship*
mentions	Entity	*Nonspecific reference in documentation*
sameAs	URL	*Equivalence relative to external resource*
similarTo	URL	*Similarity relative to external resource*
linksTo	URL	*Nonspecific reference to external resource*
documentedBy	Contributor	Authorship of documentation
memberOf	Vocabulary	*Membership in vocabulary of terms*
Contribution		
uses	Language	*Language usage*
uses	Technology	*Technology usage*
uses	Concept	*Concept usage*
implements	Feature	*Feature implementation*
developedBy	Contributor	*Developer of contribution*
reviewedBy	Contributor	*Reviewer of contribution*
memberOf	Theme	*Membership in theme of contributions*
basedOn	Contribution	*Indication of reuse*
varies	Contribution	*Indication of variation*
moreComplexThan	Contribution	*Indication of complexity*

documentedBy	Contributor	Authorship of documentation
memberOf	Vocabulary	*Membership in vocabulary of terms*
Contribution		
uses	Language	*Language usage*
uses	Technology	*Technology usage*
uses	Concept	*Concept usage*
implements	Feature	*Feature implementation*
developedBy	Contributor	*Developer of contribution*
reviewedBy	Contributor	*Reviewer of contribution*
memberOf	Theme	*Membership in theme of contributions*
basedOn	Contribution	*Indication of reuse*
varies	Contribution	*Indication of variation*
moreComplexThan	Contribution	*Indication of complexity*
Resource		
describes	Language	Language definitions, et al.
describes	Technology	API specifications, et al.
describes	Concept	Textbook, white papers, et al.
Technology		
uses	Language	*Language usage*
uses	Technology	*Technology usage*
uses	Concept	*Concept usage*
implements	Language	*Parsers, compilers, interpreters, et al.*
implements	Resource	*Compliance with a standard, et al.*
supports	Concept	*Support of a protocol, et al.*

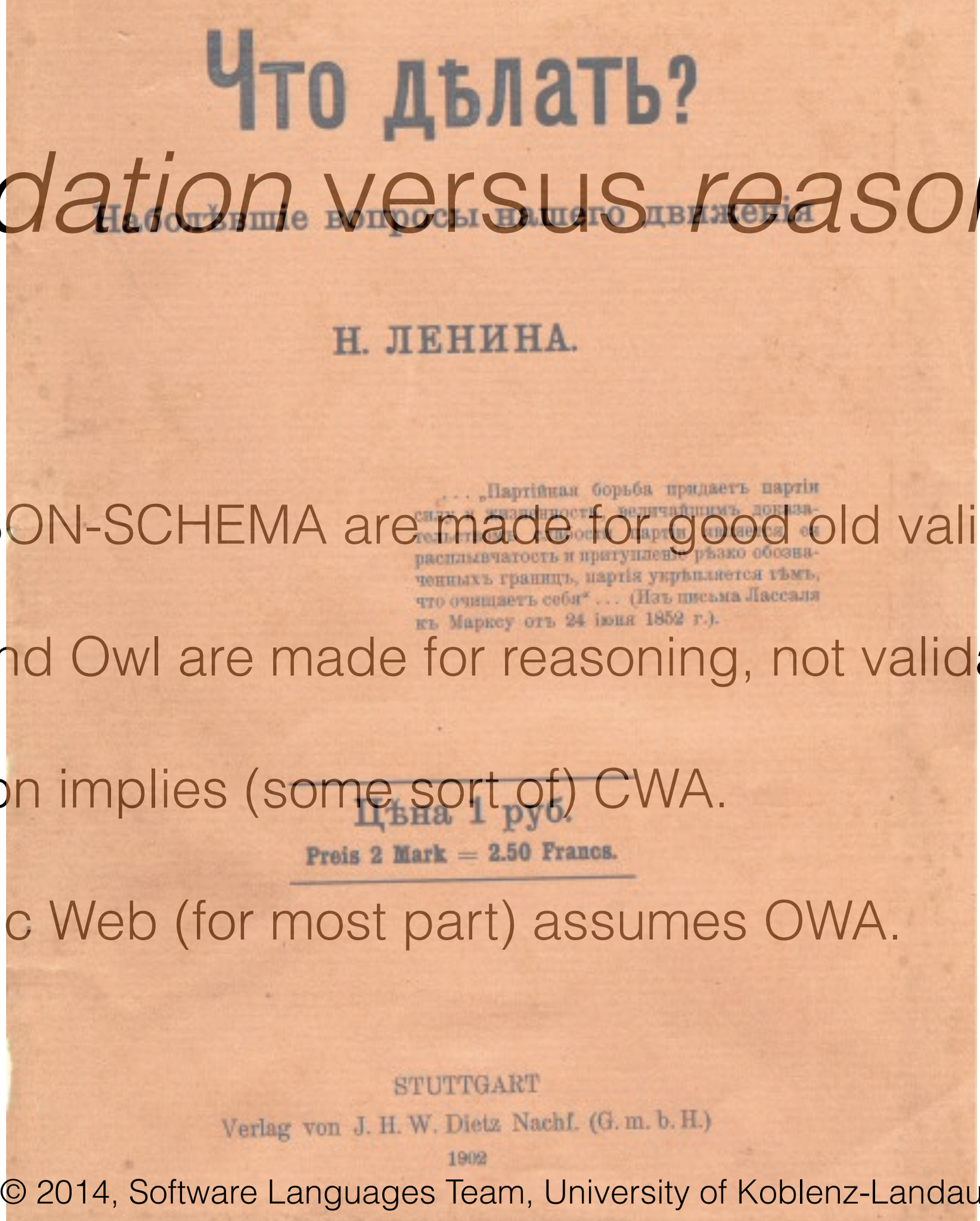
Technicalities of SoLaSoTe

```
<rdf:RDF ...>

<rdf:Description rdf:about="http://...#Entity">
<rdf:type rdf:resource="http://...#Class"/>
</rdf:Description>

<rdf:Description rdf:about="http://...#Language">
<rdfs:subClassOf rdf:resource="http://...#Entity"/>
</rdf:Description>

<rdf:Description rdf:about="http://...#Technology">
<rdfs:subClassOf rdf:resource="http://...#Entity"/>
</rdf:Description>

<rdf:Description rdf:about="http://...#Concept">
<rdfs:subClassOf rdf:resource="http://...#Entity"/>
</rdf:Description>

<owl:AllDisjointClasses>
<owl:members rdf:parseType="Collection">
<owl:class rdf:about="http://...#Language"/>
<owl:class rdf:about="http://...#Technology"/>
<owl:class rdf:about="http://...#Concept"/>
</owl:members>
</owl:AllDisjointClasses>

</rdf:RDF>
```

„Do you use OWL?“

Rarely

Validation versus reasoning



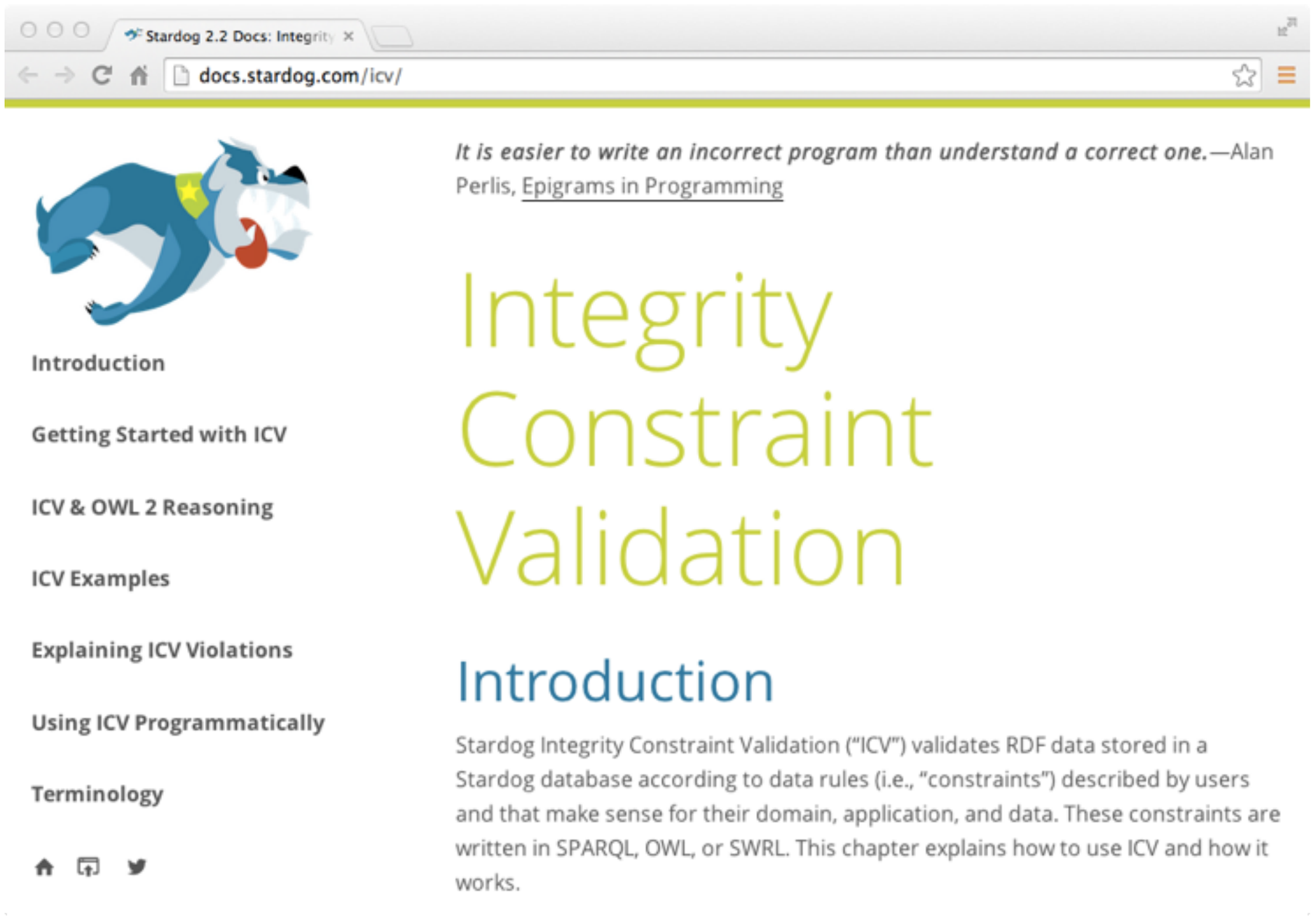
- XSD, JSON-SCHEMA are made for good old validation.
- RDFS and Owl are made for reasoning, not validation.
- Validation implies (some sort of) CWA.
- Semantic Web (for most part) assumes OWA.

Our validation process

- **Extract RDF triples from the semantic wiki:**
  - Entity becomes a root class.
  - Language, Technology, etc. become subclasses of Entity.
  - The isA properties give rise to rdfs:subClassOf properties.
  - The instanceOf properties give rise to rdf:type properties.
  - All other semantic properties are adopted, as is.
- **Analyze the integrity of the RDF triples:**
  - All resources have an rdf:type property.
  - The subjects and objects of properties agree with the schema.
  - No properties other than those of the schema are used.
  - An instance is never specialized (as in OWL DL).

BTW, such techniques are used elsewhere.

http://docs.stardog.com/icv/



```
{
    "@id": "onto:Technology",
    "@type": "onto:Entity",
    "properties": [
        {
            "property": "onto:uses",
            "range": "onto:Technology",
            "minCardinality": "0"
        },
        {
            "property": "onto:supports",
            "range": "onto:Protocol",
            "minCardinality": "0"
        },
        {
            "property": "onto:implements",
            "range": "onto:Specification",
            "minCardinality": "0"
        }
    ]
}
```

DSL for constraints

SoLaSoTe in action

OpenRDF Workbench – Qu ×

[redacted]

Workbench

OpenRDF

Sesame server

Repositories
- New repository
- Delete repository

Explore
- Summary
- Namespaces
- Contexts
- Types
- Explore
- Query
- Export

Modify
- SPARQL Update
- Add
- Remove
- Clear

System
- Information

Current Selections:
Sesame server: http://triples.101companies.org/openrdf-sesame [change]
Repository: Testing 2 (Testing_2) [change]

Query Repository

Query Language: SPARQL

Query:

```
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl:<http://www.w3.org/2002/07/owl#>
PREFIX onto:<http://101companies.org/ontology#>
PREFIX res:<http://101companies.org/resources#>

SELECT ?language WHERE {
  ?language rdf:type onto:Language .
}
```

Limit results: 100

☑ Include inferred statements

Execute

SPARQL endpoint of SoLaSoTe

The End.