Welcome to INFO216:
Advanced Modelling

Theme, spring 2016:
Modelling and Programming the Web of Data

Andreas L. Opdahl
<Andreas.Opdahl@uib.no>

About me

- Background:
  - siv.ing (1988), dr.ing (1992) from NTH/NTNU
  - Univ. of Bergen since the early 1990-ies
  - part-time programmer for industry
  - consulting (enterprise and information modelling)
- Central research interest:
  - modelling of information systems and enterprises
  - several Forskningsråd projects and networks
- Semantic technologies:
  - semantics of modelling languages
  - Interop Network of Excellence (EU)
  - start-up on social semantic tagging (Lexitags)

New project: BDEM

- Leveraging *Big Data for Emergency Management*
  - how can semantic technologies play a part?
  - developing new Master courses

SAN DIEGO STATE UNIVERSITY

- Pending proposals:
  - knowledge graphs for newsroom systems
  - semantic workflows for science

Lecture 1

- Themes:
  - *what is the Web of Data?*
    - and what are semantic technologies?
    - ...picking up the thread from INFO116
  - *introduction to INFO216*
    - organisation of the course
    - practical things
  - *overview of Jena*
    - application programming interface (API) for Java
    - important classes and methods
  - *the programming project*

Readings

- Sources:
  - Allemang & Hendler (2011): Semantic Web for the Working Ontologist chapters 1-2
  - Apache Jena introductions
  - Apache Jena tutorials
  - Apache Jena javadoc
- Does http://wiki.uib.no/info216 work for you?
  - the detailed readings are there!

The Web of Data

Transition

- From a *Web of Documents*
  - today, the “plain old web” (PoW)
  - document-centric
  - document-to-document links
  - for humans
- to a *Web of Data*
  - the future “semantic web”, “Web 3.0”, “Linked Data”, “Web of Knowledge”
  - document- *and data-centric*
  - doc-to-doc *and data-to-data links*
  - for humans *and machines*
- *AAA = Anyone can say Anything about Any topic*

Challenge

- There's an enormous amount of data on the web
  - ...but the data are mostly not linked
    (think of a world wide web without document links!)
  - availability, accessibility does not go all the way
  - *what if we had standard ways of representing data so that linkable data could always be automatically linked?*
  - *enormous potential to solve, simplify, speed up... many critical information handling problems*
- This is the purpose of *semantic technologies*
- This is the vision behind the *Web of Data*

Tim Berners-Lee: <http://www.youtube.com/watch?v=HeUrEh-nqtU>

Developments

- Not a single coordinated effort:
  - ...a *broad variety of developments*:
    - e.g., Semantic Web, Web of Data, contextual web, microformats, Microdata, Linked Open Data cloud, company-internal semantic data, knowledge graphs, Google's knowledge graph / vault and rich snippets, Facebook's graph API, social tagging, lifting social media, Wikidata, JSON-LD...
  - ...with some (rather) *common themes*:
    - e.g., semantically tagged data, standard vocabularies, standard exchange formats, RDF graphs, openness, community-based
  - ...using many of the *same technologies*

So what are semantic (-ally tagged) data?

- *Metadata* are data about other data (actually *information*)
  - e.g., data about the format and language of a web page
- *Semantic metadata* are data about the meaning of other data
  - e.g., data about the meaning of each table and column in a relational database
- *Semantic data* (or *semantically tagged data*) are data supported by semantic metadata
  - or: semantic data are data supported by metadata about their meaning
  - e.g., the above relational database along with the data about its meaning

Can we really represent meaning?

- Only in part!
  - meaning a *complex concept* with
  - *several levels*: semantics, pragmatics, social...

- *Vocabularies* can capture certain aspects of meaning:
  - standard URIs for *types of resources*
  - standard URIs for *properties*
  - standard types for *literals*
  - *rules* about how they combine
- Other *open semantic datasets* define:
  - standard URIs for *individual resources*

How to represent semantic data?

- Can in principle be represented on many formats
- The Web of Data relies heavily on the *Resource Description Framework (RDF)*
  - a “normal form” for semantic data
  - also called “knowledge graphs”
  - used both for the data and their metadata
  - either *native/reified*, *embedded*, *or virtual*
- More expressive vocabularies are available using
  - *RDF Schema (RDFS)*
  - *Web Ontology Language (OWL)*
  - ...both can be said to build on RDF

Resource Description Framework

- Represents data as triples (“statements”):
  - *(subject, predicate, object)*
  - *subject:*
    - represents what the statement is about
    - the URI of a semantic resource
  - *predicate:*
    - represents a property of the subject resource
    - the URI of a semantic property
  - *object:*
    - represents the value of a property for a subject
    - either: the URI of a semantic resource
    - or: a literal (number, string, boolean...)

Semantic graphs and data sets

- *Graph (or Model)*:
  - a collection of *triples/statements* (possibly none)
  - *"knowledge graphs"*
- *Data set*:
  - a collection of graphs (at least one)
  - one of the graphs is *default/unnamed*
  - the others are *named*
  - from triples/statements:
    - *(subject, predicate, object)*
  - to quadruples *(quads):*
    - *(graph, subject, predicate, object)*

Organisation of the course

Curriculum

- Mandatory:
  - textbook:
    Allemang & Hendler: Semantic Web for the Working Ontologist, 2nd ed. 2011
  - lectures
  - materials in the wiki (wiki.uib.no/info216)
    - introductions, tutorials, javadoc
    - standards documents
    - academic papers
- Cursory:
  - further materials in the wiki (wiki.uib.no/info216)

Lectures and labs

- Around 14 lectures:
  - usually Thursdays 1015-1200
  - mostly theory
    - some workshop-style parts
      - bring laptops in case!
  - *lectures are part of the curriculum ("pensum")*
  - no lectures on:
    - March 2nd, March 9th, April 13th
- Around 14 lab days, *starting next week*:
  - lab leader: Sigve Solvaag
  - 4 lab days used for project presentations/discussions:
    - February 7th, March 7th, April 4th, May 2nd
  - the rest are practical assignments
  - 80% mandatory, including *all the presentation days*

Theory lectures (semi-tentative)

1. Introduction
2. RDF
3. RDFS
4. Architecture
5. Services
6. SPARQL
7. Visualisation*
8. RDFS Plus
9. Vocabularies
10. Linked Open Data
11. Resources
12. OWL
13. OWL DL
14. Ontology development*
15. Interoperability*

You learn programming (mostly) through the lab exercises and project!

Lab exercises (tentative)

1. Eclipse and Jena
2. RDF programming
3. RDFS programming
4. *Project presentations*
5. Storing graphs & TDB
6. Web APIs & JSON-LD
7. SPARQL with Fuseki
8. *Project presentations*
9. SPARQL in Jena
10. Client-side presentation
11. SPARQL Update
12. *Project presentations*
13. Protege-OWL
14. OWL programming
15. *Project presentations*

You will learn Jena programming (mostly) through the lab exercises and project!

Programming project

- Mandatory programming project by May 22th
  - groups:
    - 3 people recommended, 1-2 ok, *never 4 or more*
  - counts 40% of course grade
    - follow-up meetings
    - mandatory presentations
      - February 7th, March 7th, April 4th, May 2nd
      - May 22nd
    - graded on deliveries *and process*
      - originality, effort, code, interfaces...
  - *we will outline project types*

Programming project

- The programming project shall develop a semantic/linked data application. Development and run-time platform is free choice, as is programming language.
- The project should be carried out in groups of three and not more. Working individually or in pairs is possible, but not optimal. Groups of more than three will not be accepted.
- The application will be presented in the seminar groups, and each group member will describe their contribution to the finished product. The assignment must be done in the teaching semester.
- ***...more about that later!***

Evaluation

- Two-part evaluation:
  - individual, written 3-hour exam (60%)
  - group assignment/programming project (40%)
- Exam requirements:
  - submitted programming project
  - participation in 80% of labs

Programming RDF with Jena

(and RDFS, OWL, SPARQL...)

Jena

- Java framework for the Web of Data
  - for developing apps, services and servers
  - provides a collection of Java libraries and tools (API = Application Programming Interface)
  - stores information as RDF triples in directed graphs
    - in memory, files and databases
  - allows adding, removing, manipulating, and moving that information
- Originally developed by HP
  - later taken over by Apache

http://jena.apache.org/about_jena/

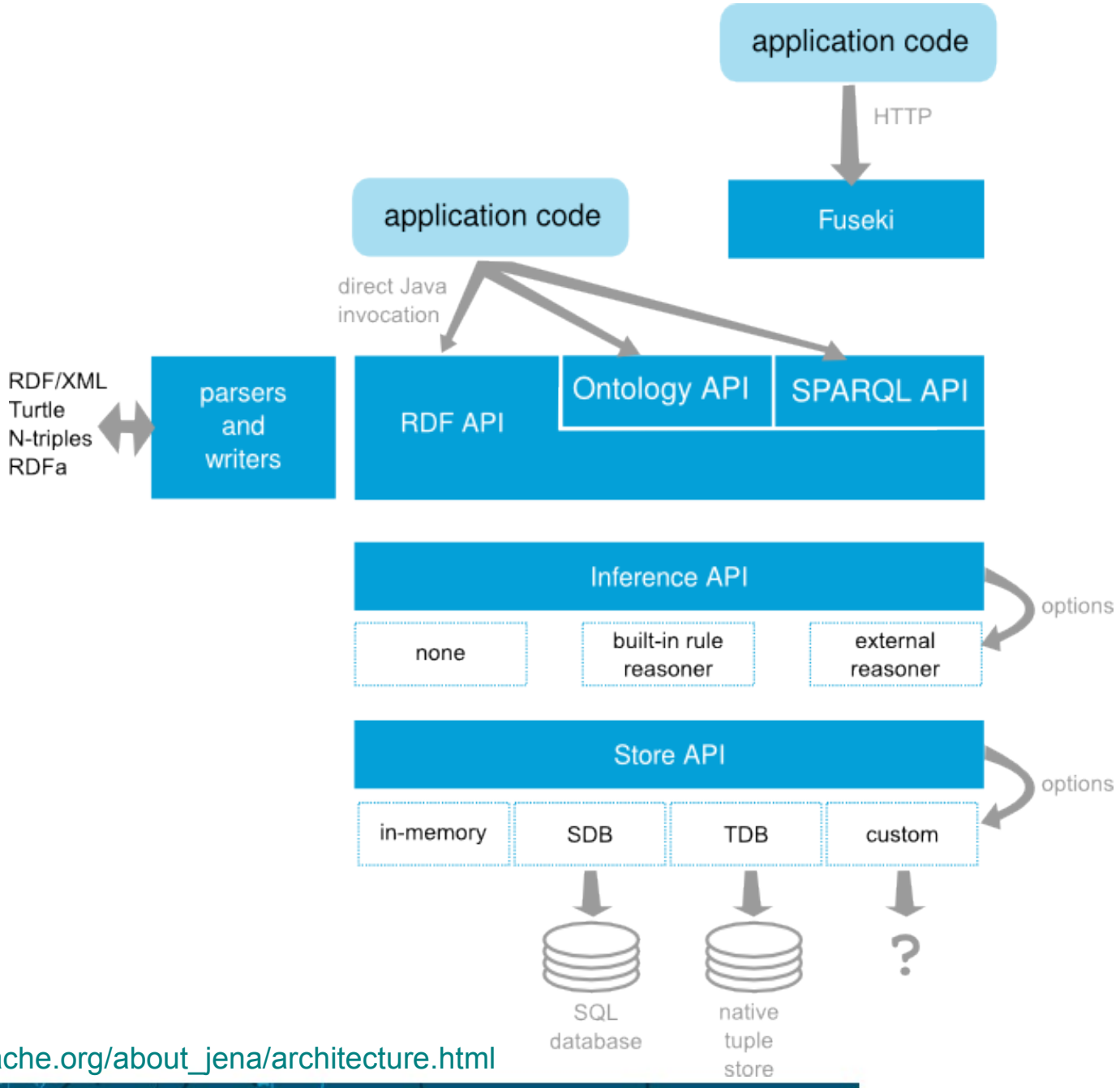
Components

- The Jena Framework includes:
  - an API for reading, processing and writing RDF data in XML, N-triples, Turtle, JSON-LD, and other formats;
  - a triple store to allow large numbers of RDF triples to be efficiently stored on disk;
  - a query engine compliant with the latest SPARQL specification;
  - a server to allow RDF data to be published to other applications using a variety of protocols, including SPARQL;
  - an ontology API for handling OWL and RDFS ontologies;
  - a rule-based inference engine for reasoning with RDF and OWL data sources.

RDF API

Jena's RDF API:

- for accessing graphs, triples and their components
  - adding, removing, finding triples and graphs
  - parsing and writing RDF files
- *Resource* represents an RDF resource
- *Literal* for data values
- *Property* represents an RDF property
- *Statement* represents an RDF triple
- *Model* represents an RDF graph
- *Graph* represents an RDF graph much more plainly
  - makes it simpler to implement, e.g., storage



http://jena.apache.org/about_jena/architecture.html

Store API

- Jena Store API:
  - by default, Jena stores *Models* in-memory
    - ...as wrapped *Graphs*
  - TDB stores the *Graphs* in persistent, indexed files
    - it is an example of a *triple store*
    - a DBMS for RDF graphs and data sets
  - *adapters* can be written to connect *Graphs* to other types of external stores

Reasoner API

- Jena's reasoner API:
  - uses semantic rules defined by RDFS and OWL to infer triples that are not explicitly stated in the graph
  - makes these entailed triples available in the store just as if they had been added explicitly
  - uses built-in rule engines for RDFS and OWL
  - engines for application custom rules
  - interfaces to external reasoners
    - e.g., description logic (DL) engines through the DIG (DL Interface Group) protocol

Ontology and SPARQL API

- Jena's Ontology API:
  - adding, removing, finding classes, individuals and properties
  - parsing and writing OWL files
- Jena's SPARQL API:
  - conforms to all of the published standards
  - supports both SPARQL Query and Update
  - queries/updates as strings or using constructors

Vocabulary APIs

- Jena's Vocabulary APIs:
  - define classes for well-known vocabularies
  - classes have static fields for the types, classes, properties etc. that are defined by the vocabulary
  - the schemagen tool generates new vocabularies from RDF/OWL descriptions

Tools

- Fuseki:
  - data publishing server
  - present and update RDF models over the web using SPARQL and HTTP
- Other tools, e.g.:
  - ARQ: command-line SPARQL motor
  - TDB: indexed file storage system
  - assemblers: setting up Jena Models from RDF specs
  - schemagen: generating Java vocabulary classes from OWL or RDFS vocabularies
  - eyeball: checking RDF (and OWL) models

Important packages

- org.apache.jena.rdf.model: The RDF API
- org.apache.jena.datatypes: Dealing with typed literals
- org.apache.jena.riot: Parsing and writing RDF.
- org.apache.jena.tdb: The Store API
- org.apache.jena.ontology: The ontology API
- org.apache.jena.rdf.listeners: Listening for changes
- org.apache.jena.reasoner: The reasoner API
- org.apache.jena.shared: Utility classes
- org.apache.jena.vocabulary: Predefined vocabularies.
- org.apache.jena.xmloutput: Writing RDF/XML

Programming project

Past projects

- Example projects:
  - *make your own muncipalities*
  - *map of party financing*
  - *reasoning over toll roads*
  - *social assessment network*
  - *LinkedMDB-portal*
  - *tracking IT infrastructure*
  - *music concert assistant*
  - *quiz generator*
  - *live semantic flight data*
  - *semantic security service*

Success factors

- Show that you can program with semantic technologies
  - *at least* RDF, *preferred* RDFS, SPARQL, ...
  - ...JSON-LD is an *emerging alternative*
- Use existing data sets (open semantic resources)
- Use existing vocabularies (and perhaps extend them)
- *Simple* presentation interface / dashboard
- Make the program run :-)
- *Shortcuts can be ok* (some manual steps, artificial data)
- Progress throughout the semester
- Final presentation (May 2nd), submission (May 22th)

Try to have an original idea

Example: combination projects

- Take two or more (semantic?) data sets
- Read them
- If necessary: lift them (i.e.: add semantic tags)
- Combine the data sets semantically
- Use them to derive new data/answer new queries
  - impossible to answer before
  - harder to answer before
- Mantainability:
  - what happens when the data sets change?
- *Dynamic data sets are more interesting that static ones!*

Exmaple: lifting projects

- Take a data set or a Web API (web service)
- Read it / access it over the net
- Lift it (i.e.: add semantic tags)
  - using existing vocabularies as far as possible
- Show and implement use cases
  - that were impossible before
  - that were harder before
  - that were less flexible before
- *Focus on maintainability – making it easy run over time!*

Other projects are very possible!

- Combination and lifting projects are the most common
- Other types are very possible, e.g.:
  - semantic crawlers and spiders
  - presentation / visualisation of graphs
- *You are free to propose (almost) anything!*

- How big should my project be?
  - usually not a problem
  - always possible to narrow the scope
  - usually possible to expand the scope
  - a bit easier to start “too big” than “too small”

Expectations to first meeting (February 7th)

- Alone or in groups of 2-3
  - not plenary the first time
  - first talk to Sigve (if you want), then with me
- Which data sets will you use?
- Which vocabularies will you use?
- What will you use them for?
  - something that cannot be done today
  - something that is harder to do today
  - something that is harder to do flexibly today
- You may bring several alternatives
  - but make sure you have a clear favourite