Semantic Web, SW Services, Grid, Cloud

Martin Kuba, ÚVT MU
makub@ics.muni.cz

Outline

- Semantic Web
  - expressing semantics
  - RDF, OWL, ontologies
  - Semantic Web Services
- Grid
- Cloud
- Containers
- Infrastructure as Code

Semantic Web

- idea introduced by Tim Berners Lee (inventor of WWW) in 2001
- *“The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation”*
- web instead of platform for distributed presentations would be platform for distributed knowledge

Semantic Continuum

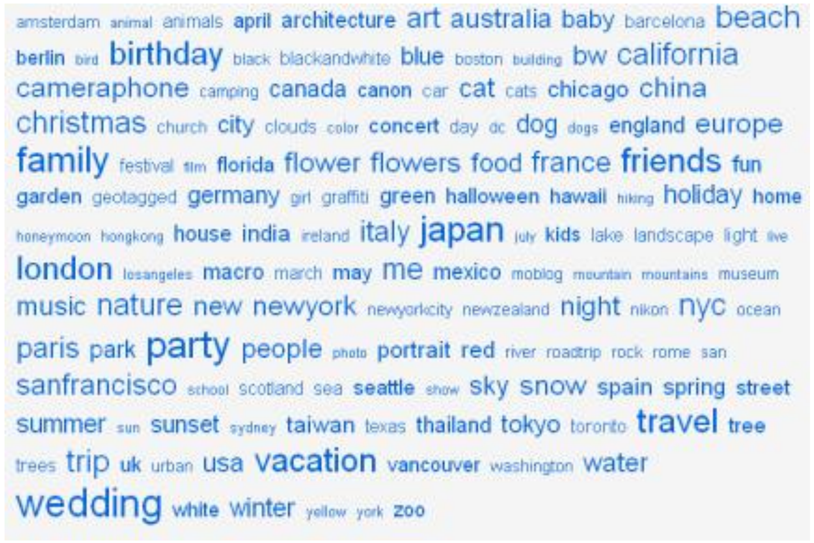
- semantics = meaning
- semantic in SW means *machine-processable*
- semantic continuum (Uschold 2003)
  a. implicit semantics in the minds of humans
  b. explicit informal semantics (text description in natural language, e.g. HTML specification)
  c. formal semantics for humans (in formal language processed by humans)
  d. formal semantics for machine processing
- goal is to create robotic decision-making devices
- metadata - data about data

Expressing Semantics

- folksonomies
- microdata
- RDF triples and RDF Schema vocabularies
- OWL-DL ontologies for automated reasoning

Folksonomies

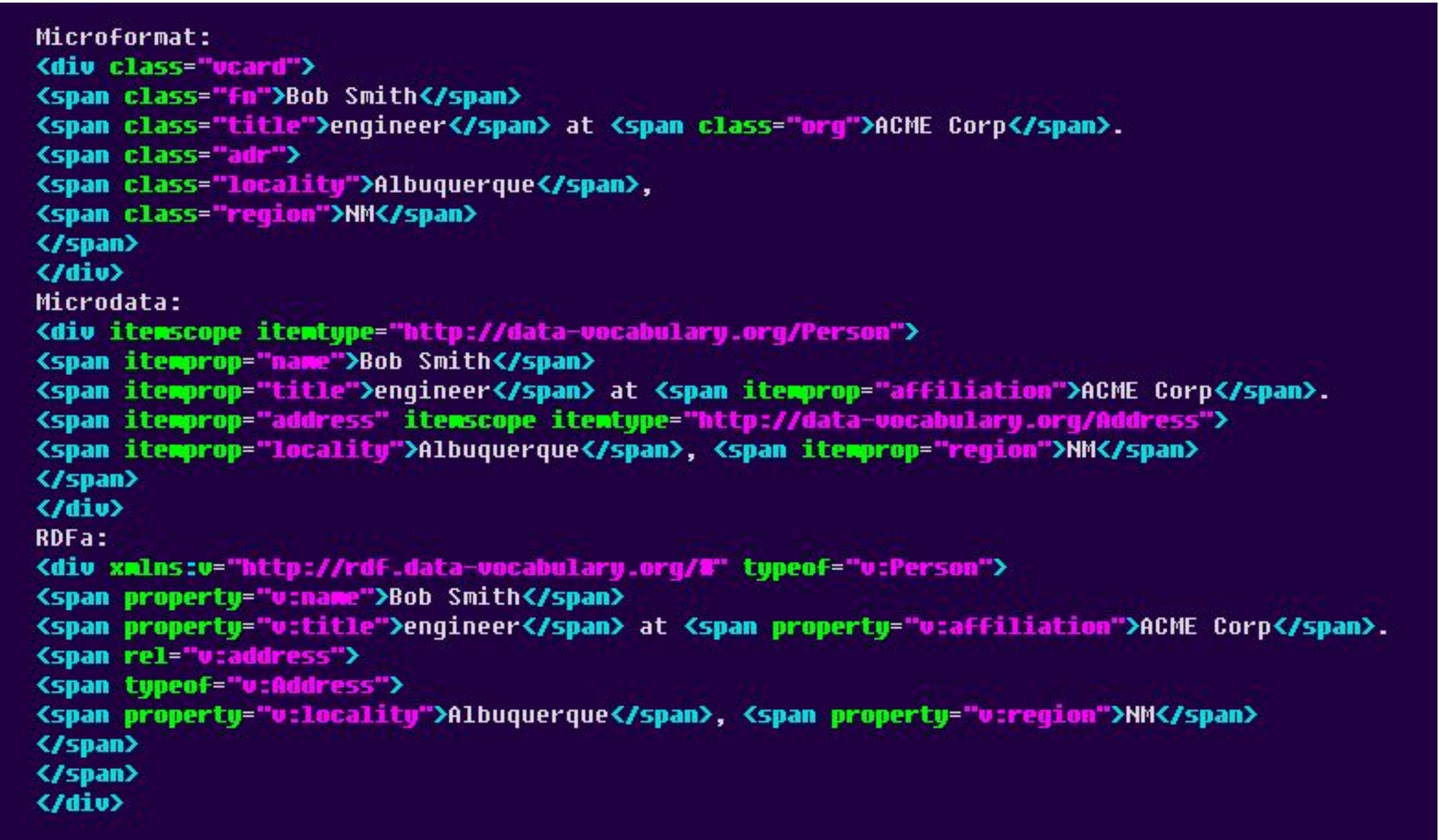
- keyword metadata as **tags**
- e.g. an image of a dog may be tagged with tags *dog*, *collie* or *pet*
- (+) low entry barrier, no user training
- (-) no synonym control, flat structure
- tag clouds


amsterdam animal animals april architecture art australia baby barcelona beach
berlin bird birthday black blackandwhite blue boston building bw california
cameraphone camping canada canon car cat cats chicago china
christmas church city clouds color concert day dc dog dogs england europe
family festival film florida flower flowers food france friends fun
garden geotagged germany girl graffiti green halloween hawaii hiking holiday home
honeymoon hongkong house india ireland italy japan july kids lake landscape light live
london losangeles macro march may me mexico moblog mountain mountains museum
music nature new newyork newyorkcity newzealand night nikon nyc ocean
paris park party people photo portrait red river roadtrip rock rome san
sanfrancisco school scotland sea seattle show sky snow spain spring street
summer sun sunset sydney taiwan texas thailand tokyo toronto travel tree
trees trip uk urban usa vacation vancouver washington water
wedding white winter yellow york zoo

Microdata

- competing Microdata, Microformats, RDFa
- nesting semantics within existing content on web pages
- RDFa only inside XML, not in HTML5
- Microdata provides JavaScript API
- Microdata use namespace-qualified vocabularies predefined at data-vocabulary.org or schema.org
- supported by Google search engine
- opposite vision than in 2000:
  - XML with CSS or XSLT - semantic markup with presentational metadata
  - HTML5 with Microdata - presentational markup with semantic metadata

Comparison of Microdata and others

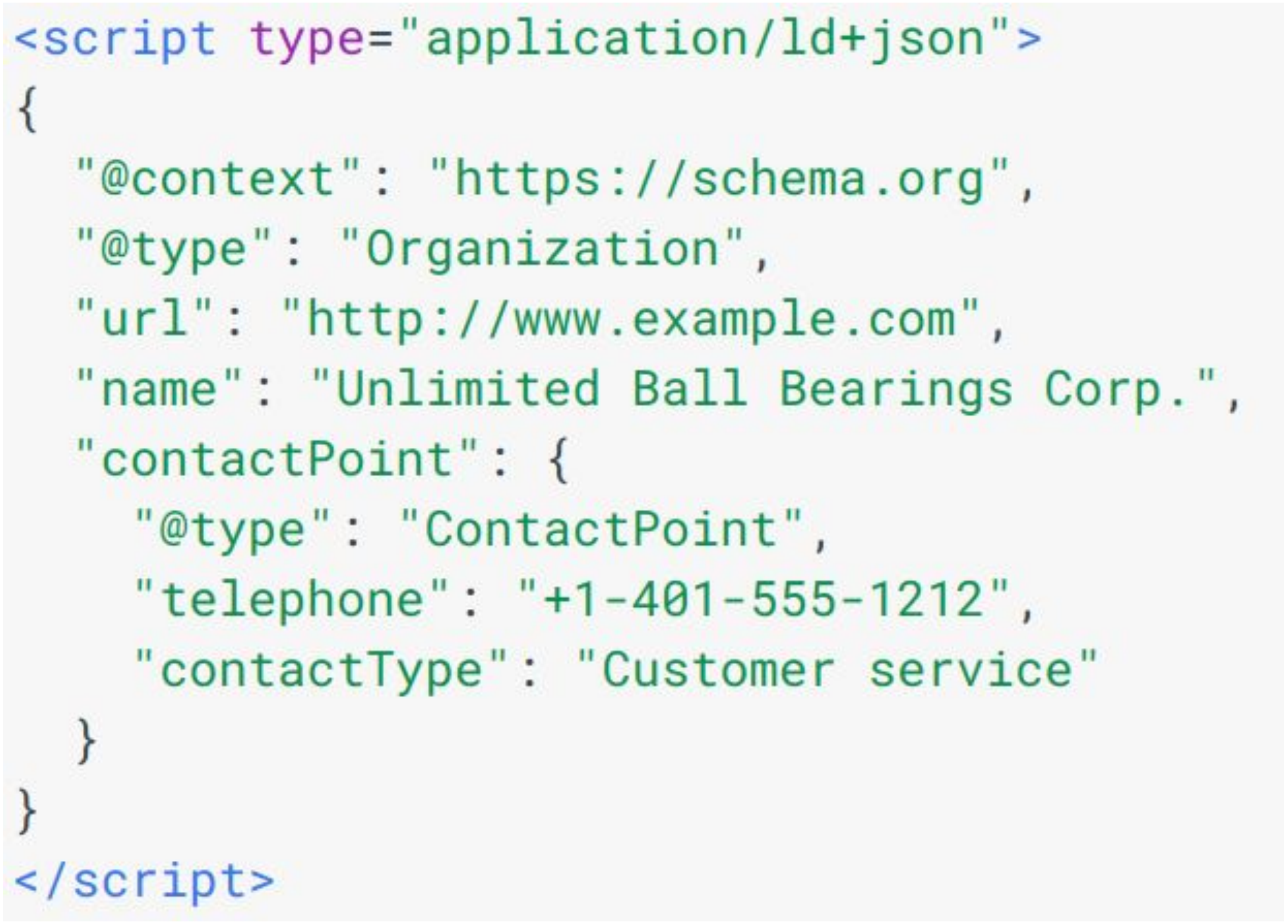


```
Microformat:
<div class="vcard">
<span class="fn">Bob Smith</span>
<span class="title">engineer</span> at <span class="org">ACME Corp</span>.
<span class="adr">
<span class="locality">Albuquerque</span>,
<span class="region">NM</span>
</span>
</div>
Microdata:
<div itemscope itemtype="http://data-vocabulary.org/Person">
<span itemprop="name">Bob Smith</span>
<span itemprop="title">engineer</span> at <span itemprop="affiliation">ACME Corp</span>.
<span itemprop="address" itemscope itemtype="http://data-vocabulary.org/Address">
<span itemprop="locality">Albuquerque</span>, <span itemprop="region">NM</span>
</span>
</div>
RDFa:
<div xmlns:v="http://rdf.data-vocabulary.org/#" typeof="v:Person">
<span property="v:name">Bob Smith</span>
<span property="v:title">engineer</span> at <span property="v:affiliation">ACME Corp</span>.
<span rel="v:address">
<span typeof="v:Address">
<span property="v:locality">Albuquerque</span>, <span property="v:region">NM</span>
</span>
</span>
</div>
```

JSON-LD

- Microdata are deprecated since 2016
- Microdata DOM API is deprecated since 2018 in Mozilla, removed since Firefox 49
- JSON-LD (JavaScript Object Notation for Linked Data) replaces them
- W3C Recommendation since 2014
- Google parses JSON-LD and uses them in searches

Example of JSON-LD



```
<script type="application/ld+json">
{
  "@context": "https://schema.org",
  "@type": "Organization",
  "url": "http://www.example.com",
  "name": "Unlimited Ball Bearings Corp.",
  "contactPoint": {
    "@type": "ContactPoint",
    "telephone": "+1-401-555-1212",
    "contactType": "Customer service"
  }
}
</script>
```

RDF - Resource Description Framework

- statements about web resources
- triples *subject-predicate-object*
- subject and predicate are URIs
- object can be a URI or a data value
- **reification** - an RDF statement is assigned a URI and treated as a resource
- producers and consumers of RDF statements must agree on the semantics of the resource identifiers, conveyed by some controlled vocabulary

RDF Schema

- tool for defining controlled vocabularies
- defines
  - classes of things
  - properties (binary predicates)
  - subsumption relationships (subclasses, subproperties)
  - `rdf:type` - resource is an instance of a class
- **SPARQL** (SPARQL Protocol and RDF Query Language) is an SQL-like language for querying RDF graphs
- *entailment rules* allow to entail e.g. that when a resource is in a particular class, then it is also in all its superclasses

RDF Schema example

- RDFS can define two classes:
    - Person
    - Student as subclass of Person
- a RDF statement may state that a resource representing John Doe is of rdf:type Student
- by entailment, John Doe is also a Person

OWL

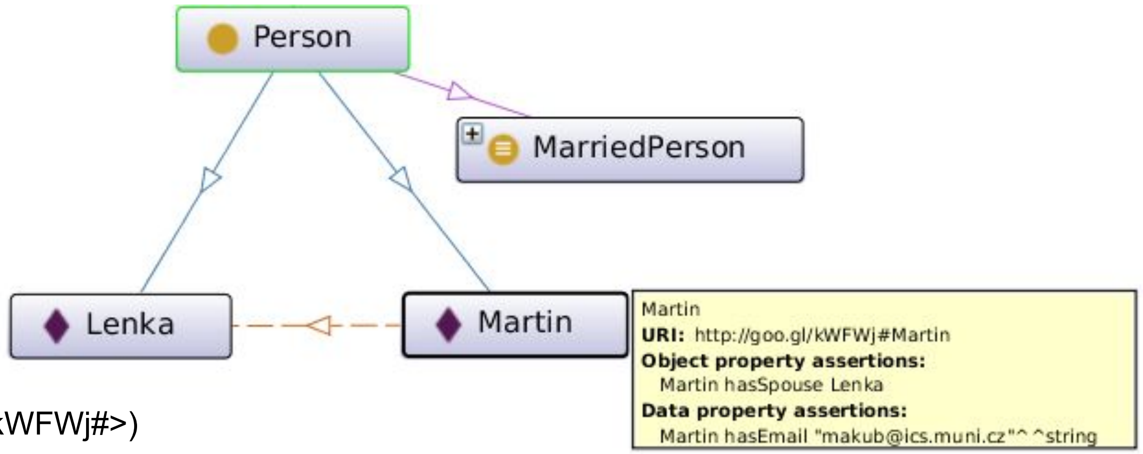
- Web Ontology Language defined by W3C
- **ontology** is a term from artificial intelligence
- ontology is “*an explicit (written) formal conceptualization*”, used for capturing knowledge about some domain of interest
- “**conceptualization** is an abstract simplified view of some selected part of the world, containing the objects, concepts, and other entities that are presumed of interest for some particular purpose and the relationships between them”
- OWL 1 released in 2004, OWL 2 in 2009
- two different (incompatible) semantics
  - RDF based - OWL Full
  - DL (Description Logics) based - OWL DL

Types of logic

- Propositional logic
  - formulae made of atomic propositions with values **true** or **false**, and logical connectives like **negation** ($\neg A$), **and** ($A \land B$), **or** ($A \lor B$) and **implication** ($A \rightarrow B$)
  - sound, complete and decidable in finite time
- Predicate logic
  - adds predicates, quantifiers, terms
  - formulae look like $\forall x \exists y (P(x) \rightarrow Q(f(y)))$
- First order predicate logic
  - quantifiers can range only over elements of sets
  - sound and complete, but **not decidable**
- Description logics
  - logics designed to be as expressive as possible while retaining decidability

OWL DL

- Description Logics is a decidable fragment of First Order Predicate Logic (FOL) plus decidable extensions
- **reasoners** - software able to entail complete inferable knowledge in finite time
- OWL DL uses:
  - classes
  - individuals
  - properties (binary relations)
    - object properties (between two objects)
    - data properties (between object and data literal)
- can use SWRL (Sem. Web Rule Language)



Martin
URI: http://goo.gl/kWFWj#Martin
Object property assertions:
Martin hasSpouse Lenka
Data property assertions:
Martin hasEmail "makub@ics.muni.cz"^^string

```
Prefix(:=<http://goo.gl/kWFWj#>)
Prefix(xsd:=<http://www.w3.org/2001/XMLSchema#>
Ontology(<http://goo.gl/kWFWj>
Declaration( Class( :Person ) )
Declaration( Class( :MarriedPerson ))
Declaration( NamedIndividual( :Martin ) )
Declaration( NamedIndividual(:Lenka ) )
Declaration( ObjectProperty(:hasSpouse ) )
Declaration( DataProperty(:hasEmail ) )
SymmetricObjectProperty( :hasSpouse )
FunctionalObjectProperty( :hasSpouse )
ClassAssertion( :Person :Lenka )
ClassAssertion( :Person :Martin )
DifferentIndividuals( :Martin :Lenka )
ObjectPropertyAssertion( :hasSpouse :Martin :Lenka )
DataPropertyAssertion( :hasEmail :Martin "makub@ics.muni.cz"^^xsd:string )
SubClassOf( :MarriedPerson :Person )
EquivalentClasses( :MarriedPerson ObjectSomeValuesFrom( :hasSpouse :Person ))
```

OWL DL Tools

- ontology editor with GUI - Protege
  - http://protege.stanford.edu/
- reasoners
  - Pellet
  - HermiT
  - FACT++
  - Stardog
- Java API for OWL - OWL API
  - http://owlapi.sourceforge.net/

Limits of OWL DL

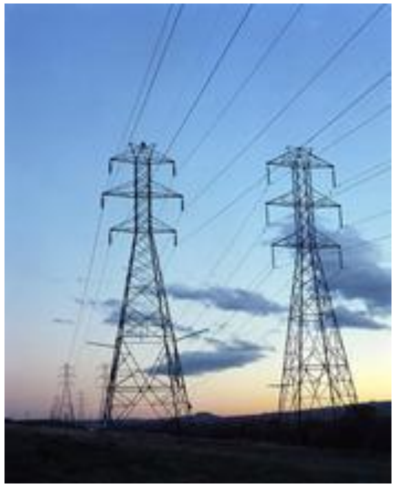
- based on FOL $\forall x \exists y (P(x) \rightarrow Q(f(y)))$
- cannot express
  - fuzzy expressions - “It **often** rains in autumn.”
  - non-monotonicity - “Birds fly, penguin is a bird, but penguin does not fly.”
  - propositional attitudes - “Eve **thinks** that 2 is not a prime number.”
  - modal logic
    - possibility and necessity - “It is **possible** that it will rain today.”
    - epistemic modalities - “Eve **knows** that 2 is a prime number.”
    - temporal logic - “I am **always** hungry.”
    - deontic logic - “You **must** do this.”
- Transparent Intensional Logic (TIL)
  - can express anything that can be said
  - has no calculus or reasoning algorithms

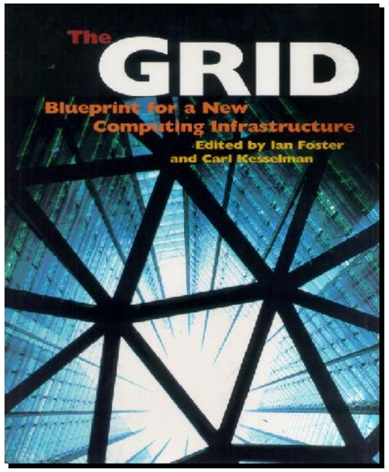
Semantic Web Services

- research efforts OWL-S, WSDL-S, WSMO
- semantics can enhance discovery
  - on the semantic continuum move it from b) to d)
  - e.g. search for "getHardDriveQuote" can find also "getQuoteForHardDrive" (synonym) and "getSCSIDriveQuote" (subsumed term)
- web service semantics
  - **Data semantics** - it defines meaning of the data, i.e. inputs and outputs of operations
  - **Functional semantics** - it defines meaning of the operations, i.e. how they transform input to output
  - **QoS semantics** - it provides meaning for quality aspects, like price, availability, level of trust etc. Service selection may be based on such characteristics.
  - **Execution semantics** - it provides details like preconditions and effects of service invocation, conversation patters of service invocation etc

Grid

- term introduced in 1998 by Carl Kesselman and Ian Foster in book **"The Grid: Blueprint for a New Computing Infrastructure"**
- analogy to electrical power grid
- *"A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities."*
- in 2001 in article **"The Anatomy of the Grid"** added Virtual Organizations




The GRID
Blueprint for a New Computing Infrastructure
Edited by Ian Foster and Carl Kesselman

What is The Grid ?

- coordinates resources that are not subject to centralized control
- using standard, open, general-purpose protocols and interfaces
- to deliver nontrivial qualities of service.

Grid Usage

- high performance computing (HPC)
  - research of medical drugs
  - gravitation waves research
  - earthquake prediction
  - electronic chip engineering
  - ...
- large data
  - Large Hadron Collider in CERN
- expensive scientific instruments
  - large microscope in Japan
- remote cooperation
  - teleconferences, remote surgery, ...

Grid Middleware

- not a single middleware
  - in U.S.A. Globus
  - in Europe gLite
  - in Germany UNICORE
- services
  - information services (Globus: MDS, gLite: BDII)
  - gridFTP - striped transfer, third party transfer
  - resource allocation (Globus: GRAM, gLite: WMS)
  - virtual organization membership (VOMS)
- Computing Element
  - grid gate, batch system, cluster of worker nodes
- Storage Element
  - disk servers, disk arrays, tape storage

Grid Security

- based on X509 certificates and PKI (Public Key Infrastructure)
- list of selected CAs (Certification Authorities) maintained by IGTF (International Grid Trust Federation)
- allows so-called **proxy certificates**
  - short-lived (24 hours, 1 week)
  - a certificate signed by a user certificate or proxy cert.
  - can be delegated to a running job
- VOMS (Virtual Organisation Membership Service) issues attribute certificates specifying user privileges on resources

European Grid History

- in 2001-2003 project DataGrid
  - for processing massive data produced by Large Hadron Collider in CERN
- in 2004-2010 projects EGEE I, II, III
- in 2010-2014 EGI (European Grid Infrastructure) built in project InSPIRE
- 2015-2017 project EGI-Engage
- EGI in November 2016
  - 730000 CPUs
  - 285 PB disk storage, 280 PB archive storage
- EGI consists of NGIs (National Grid Infrastructures)
- Czech NGI is MetaCentrum, operated by CESNET, collects 18260 CPUs, 4.7 PB disk storage, 15.6 PB tape storage

Cloud Computing

- use of computing resources (hardware and software) that are delivered as a service over a network
- in 1960 utility computing
- in 2006 Amazon released AWS (Amazon Web Service)
    - EC2 (Elastic Compute Cloud)
    - S3 (Simple Storage Service)

Cloud definition

- **cloud computing** is a general term for anything that involves delivering hosted services over the Internet
- definition by NIST (National Institute of Standards and Technology, U.S. Department of Commerce): *Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.*
- five essential characteristics: on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service

Cloud Service Models

- Software as a Service (SaaS)
- Platform as a Service (PaaS)
- Infrastructure as a Service (IaaS)

SaaS - Software as a Service

- best known to computer users, the only one they directly use, provides **device independence**
- examples:
  - web mail – GMail, Hotmail
  - social networking and messaging – Facebook, Google+, Twitter
  - on-line office suites – Google Docs, Microsoft Office 365
  - file services – Dropbox, Google Drive, Microsoft OneDrive, ownCloud
  - image libraries – Picasa, Flickr
  - video libraries – YouTube, Vimeo
  - communication tools – Adobe Connect, WebEx
  - business software – Salesforce, NetSuite

PaaS - Platform as a Service

- platform is a software environment used to develop and run applications
- not visible to end users, targeted to application developers and maintainers delivering their SaaS applications
- examples:
  - Google App Engine (provides PHP, Python, Java, Go)
  - Amazon Elastic Beanstalk (provides Ruby, PHP, Python, .NET, Java, JavaScript)
  - Heroku (provides Ruby, PHP, Python, Java, JavaScript, Perl)
  - Microsoft Azure Websites (provides PHP, Python, .NET, JavaScript)
  - Red Hat OpenShift (provides Ruby, Python, PHP, JavaScript, Perl, Java, Haskell, .NET)

IaaS - Infrastructure as a Service

- provides a virtual data center
- IaaS provider provides **virtual machines** (VMs) with complete operating systems
- many VMs can be hosted on a single physical machine running **hypervisor** software (Xen, KVM, VMWare)
- resources hired from an IaaS cloud can be used directly (e.g. on-demand movie rendering) or as a layer under a PaaS or SaaS cloud

IaaS Providers Provide

- **disk images** with pre-installed popular operating systems (various versions of Linux, MS-Windows)
- **networking services** - virtual local area networks, virtual private networks, IP addresses, firewalls,
load balancers, domain name service (DNS)
- **storage services** - virtual block storage, file storage, object storage, relational database storage,no-SQL storage, tape archive storage, content delivery network (CDN)

IaaS Examples

- providers:
  - Amazon Elastic Compute Cloud
  - Google Compute Engine
  - Microsoft Azure
  - Rackspace Cloud Servers
- software:
  - OpenNebula
  - OpenStack
  - Eucalyptus
  - VMware vCloud Suite

Cloud Service Models Summary

- **Software-as-a-Service** model provides on-demand access to software, either as downloadable code executed on client computers, or through remote API calls to code executed on servers
- **Platform-as-a-Service** model provides on-demand software environment for deploying applications. The environment includes concrete programming languages, their specific libraries, and additional services like SQL and no-SQL storage. PaaS cloud is usually used as a layer under SaaS cloud services.
- **Infrastructure-as-a-Service** model provides on-demand resources from a virtual data center. The resources can be used directly or as a layer under PaaS or SaaS cloud services.

Masaryk University / CERIT-SC Cloud

- private IaaS cloud
- based on OpenNebula with KVM hypervisors
- disk images with Debian Linux, CentOS, SciLinux, Ubuntu and MS-Windows
- archive in the form of HSM (Hierarchical Storage Management) with layers
  - RAID disk arrays
  - massive arrays of idle disks (MAIDs)
  - magnetic tapes
- in March 2019 it provides 5000 CPUs

Docker containers

- real Virtual Machines have some overhead
- Docker is a tool for deployment of software in so-called containers
- a container is an isolated environment with complete system libraries, running inside a hosting OS
- a container is in principle a chroot directory with cgroups and namespaces, with exportable directories and TCP ports for linking to other containers
- a container can contain i.e. Ubuntu 18.04 with Apache, but can run on any Linux, MacOS X or Windows host

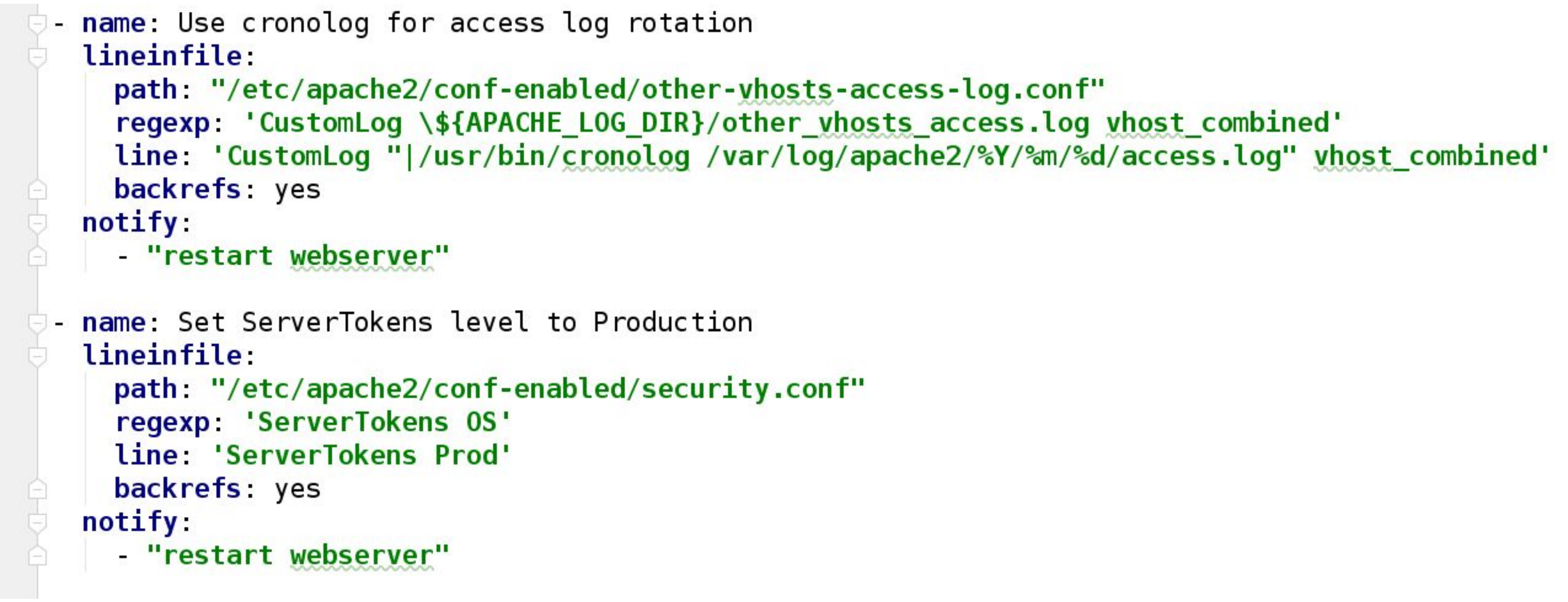
Docker containers (2)

- Docker containers are like .deb or .rpm packages, but OS-independent
- versioned repository of containers on DockerHub
- official containers for famous software (e.g. Postgres, Apache, Ubuntu, Centos, …)
- anybody can create a new container by modifying another container
- containers can be linked using TCP ports
  (e.g. Apache+PHP -> PostgreSQL)
- persistent data can be stored outside of containers using exported directories (called *volumes*)

Infrastructure as Code (IaC)

- managing computer data centers through machine-readable definition files
- no interactive configuration tools
- necessary when managing hundreds of machines
- definitions may be in a version control system (e.g. git)
- tools like Puppet, Ansible, SaltStack
- declarative approach defines the desired state of machines
- idempotent actions - can be run repeatedly with the same result as when run just once

Ansible example



```yaml
- name: Use cronolog for access log rotation
  lineinfile:
    path: "/etc/apache2/conf-enabled/other-vhosts-access-log.conf"
    regexp: 'CustomLog \${APACHE_LOG_DIR}/other_vhosts_access.log vhost_combined'
    line: 'CustomLog "|/usr/bin/cronolog /var/log/apache2/%Y/%m/%d/access.log" vhost_combined'
    backrefs: yes
  notify:
    - "restart webserver"

- name: Set ServerTokens level to Production
  lineinfile:
    path: "/etc/apache2/conf-enabled/security.conf"
    regexp: 'ServerTokens OS'
    line: 'ServerTokens Prod'
    backrefs: yes
  notify:
    - "restart webserver"
```

That’s it.

Thank you for your attention