

# Web 2.0

## Lecture 5: Data Structures – Atom and AtomPub

**doc. Ing. Tomáš Vitvar, Ph.D.**

tomas@vitvar.com • @TomasVitvar • <http://vitvar.com>



Czech Technical University in Prague

Faculty of Information Technologies • Software and Web Engineering • <http://vitvar.com/courses/w20>



EVROPSKÁ  
UNIE

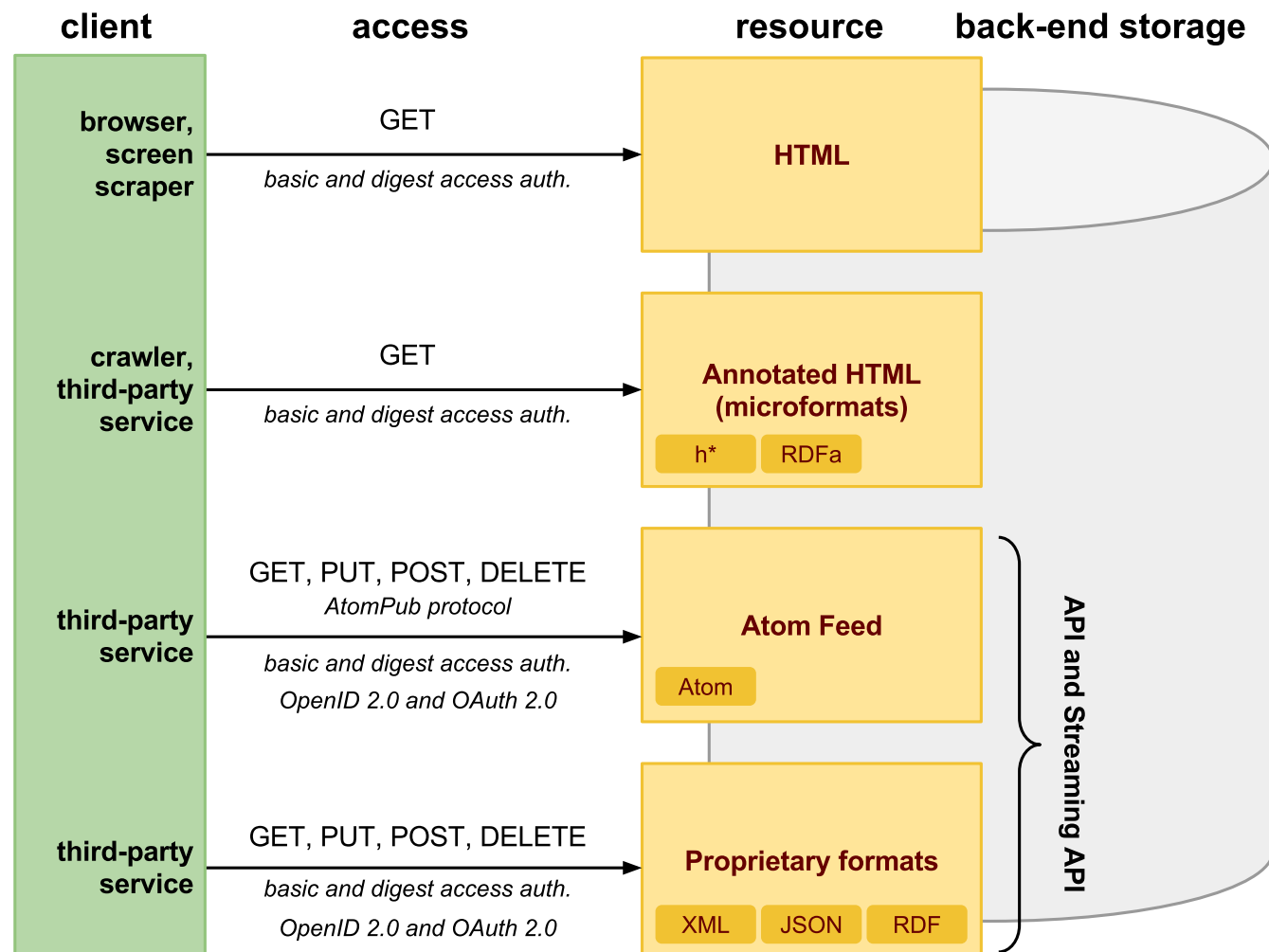
Evropský sociální fond  
Praha & EU: Investujeme do vaší budoucnosti

Modified: Tue Mar 21 2017, 16:07:13  
Humla v0.3

# Overview

- Overview of Formats and Protocols
- Atom Syndication Format
- AtomPub Protocol

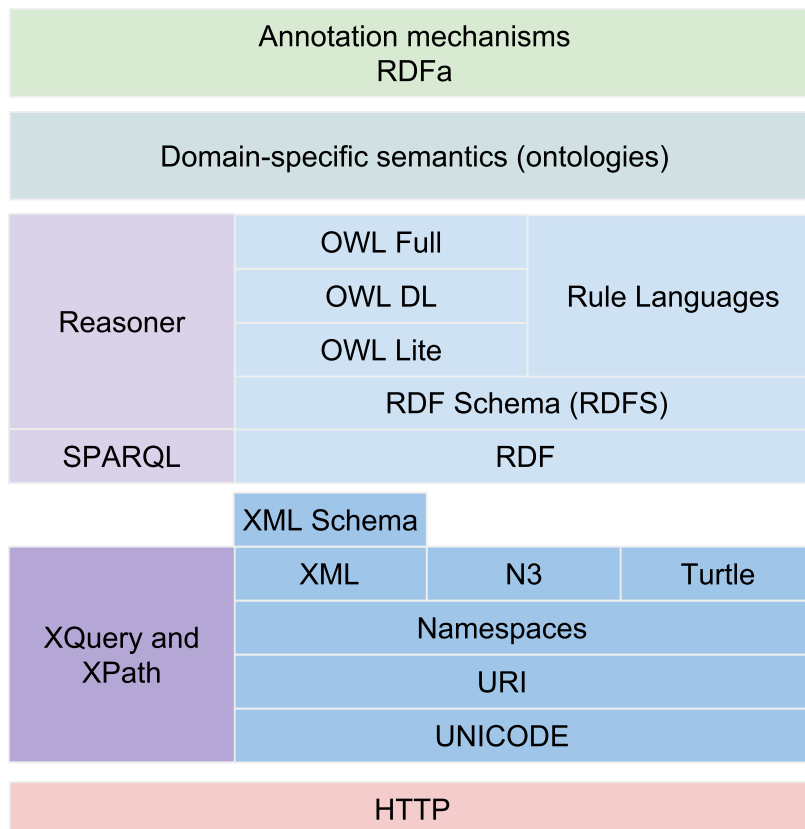
# Data on the Web



# Data Syntax, Structure and Semantics

## Semantic Web Layered Cake

syntax and formal semantics



## Web Data Formats

syntax and semantics (structure)



# Atom Standard

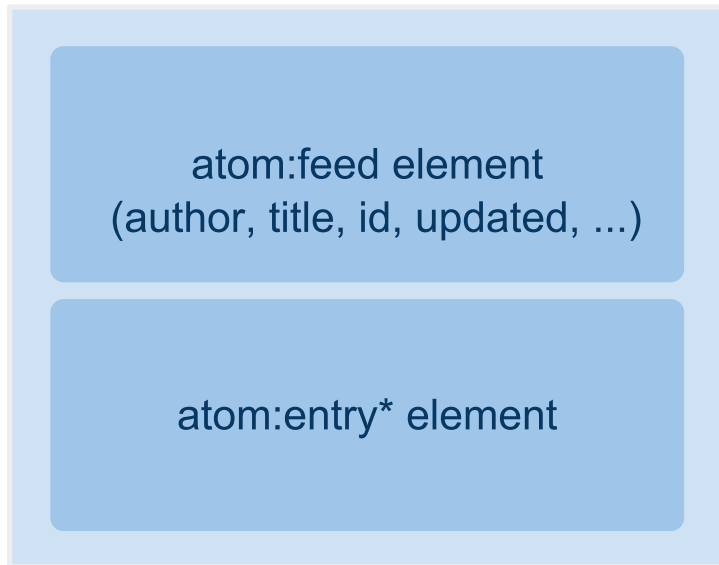
- A need for a standard syndication format
  - *machine-processable Web site content*
  - *Alternative to RSS*
    - *RSS spec does not say how to encode content, strings only ASCII-encoded, not clearly defined meaning of RSS elements, etc.*
    - *See RSS Flaws* [↗](#)
- IETF Atom Publishing Format and Protocol WG
  - *RFC 4287: Atom Syndication Format* [↗](#)
  - *RFC 5023: Atom Publishing Protocol* [↗](#)
- Adoption
  - *Google: Google Data Protocol (GData)*
  - *Microsoft: Open Data Protocol (OData)*

# Overview

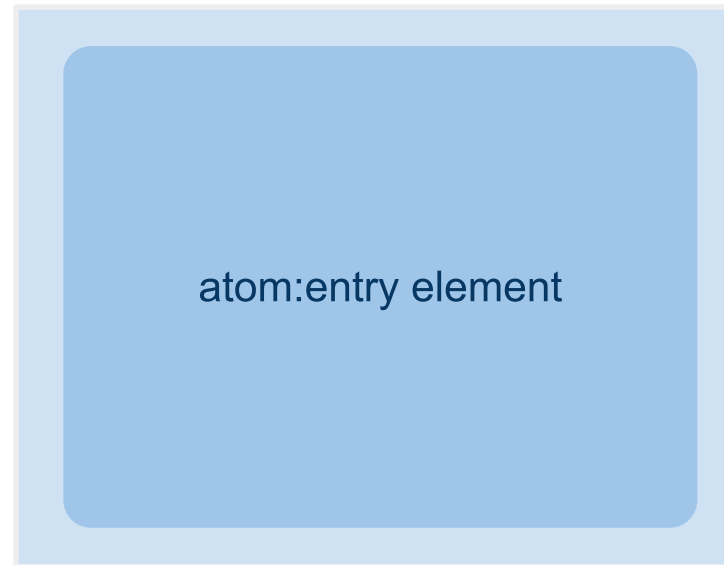
- Overview of Formats and Protocols
- Atom Syndication Format
- AtomPub Protocol

# Atom Syndication Format

## Atom Feed Document



## Atom Entry Document



- Two types of atom documents
  - *Atom Feed Document*
    - represents an atom feed, its metadata and some or all entries associated with it.
  - *Atom Entry Document*
    - represents exactly one entry, outside of context of atom feed

# Atom Syndication Format

- Atom Feed Document Example

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <feed xmlns="http://www.w3.org/2005/Atom">
3
4      <title>Example Feed</title>
5      <link href="http://example.org/" />
6      <updated>2003-12-13T18:30:02Z</updated>
7      <author>
8          <name>John Doe</name>
9      </author>
10     <id>urn:uuid:60a76c80-d399-11d9-b93C-0003939e0af6</id>
11
12     <entry>
13         <title>Example feed title</title>
14         <link href="http://example.org/2003/12/13/atom03" />
15         <id>urn:uuid:1225c695-cfb8-4ebb-aaaa-80da344efa6a</id>
16         <updated>2003-12-13T18:30:02Z</updated>
17         <summary>Some text</summary>
18     </entry>
19 </feed>
```



# Atom Elements – Atom Feed

- Specification
  - *defined as XML information set, serialized as XML 1.0*
  - *must be well-formed, no DTD/Schema → no requirements to be valid.*
- **atom:feed** element
  - (**\***): *zero or more occurrences – repeating fields*
  - (**?**): *zero or one occurrence – non-repeating fields*
  - ( **)**: *exactly one occurrence – non-repeating fields*

```
1  atomFeed =
2      element atom:feed {
3          atomCommonAttributes,
4          (atomAuthor*
5            & atomCategory*
6            & atomContributor*
7            & atomGenerator?
8            & atomIcon?
9            & atomId
10           & atomLink*
11           & atomLogo?
12           & atomRights?
13           & atomSubtitle?
14           & atomTitle
15           & atomUpdated
16           & extensionElement*),
17      atomEntry*
18  }
```

# Atom Elements – Atom Entry

- **atom:entry** element

*(\*)*: zero or more occurrences – repeating fields

*(?)*: zero or one occurrence – non-repeating fields

*( )*: exactly one occurrence – non-repeating fields

```
1  atomEntry =  
2      element atom:entry {  
3          atomCommonAttributes,  
4          (atomAuthor*  
5              & atomCategory*  
6              & atomContent?  
7              & atomContributor*  
8              & atomId  
9              & atomLink*  
10             & atomPublished?  
11             & atomRights?  
12             & atomSource?  
13             & atomSummary?  
14             & atomTitle  
15             & atomUpdated  
16             & extensionElement*)  
17      }
```

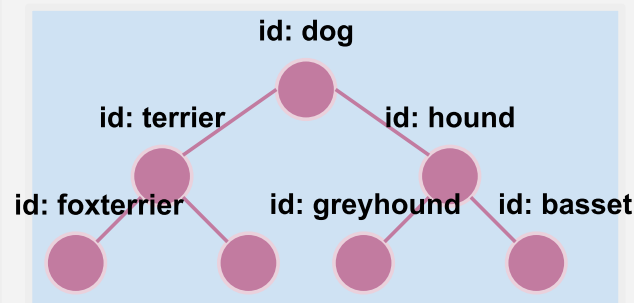
# Pointers to other information

- URI identifier
  - *unique identification of things*
  - *feed/entry id*
  - **author** *and*
  - **contributor** (*person uri*)
  - **generator** (*uri*)
  - **category** *schema (uri), term (uri)*

*example:*


```
1 | <category scheme="http://example.org/dogs"  
2 |   term="http://example.org/dogs#basset"  
3 |   label="Basset"/>
```

Example category schema  
URI: <http://example.org/dogs>



- Unambiguous identification of things using URIs
  - *Helps interoperability, can take advantage of wikipedia concepts*
  - *still not very common, will improve with linked data*

# Atom Links

- Links to other Atom documents
  - *Atom defines simple link structure*
  - **type** defines content type
  - **rel** defines relation to this resource
    - *self, alternate, related, enclosure, via*
    - *standardized by IANA*
- Adoption by RESTful services
  - *Core for HATEOAS*
  - *Adopted in Link header, see Web Linking* 
  - *More details in Lecture 4 – HATEOAS.*

# Encoding Textual Content

- Plain text

```
1 | <title type="text">
2 |   Less: &lt;
3 | </text>
```

– *simple text, must not contain child elements*

- HTML

```
1 | <title type="html">
2 |   Less: &lt;em> &amp;lt; &lt;/em>
3 | </text>
```

– *html text, must not contain child elements*

– *any markup must be escaped,*

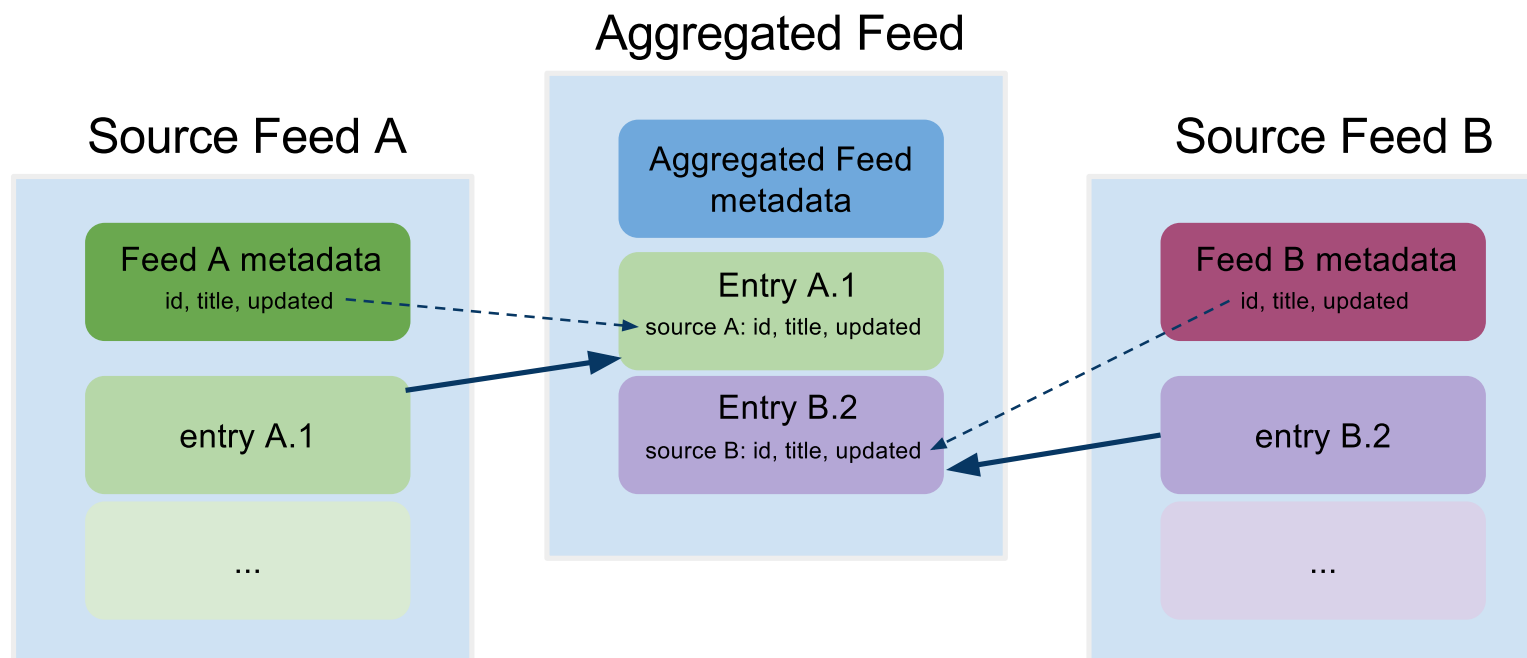
– *should be possible to display it as HTML inside <div> element*

- XHTML

```
1 | <title type="xhtml" xmlns:x="http://www.w3.org/1999/xhtml">
2 |   <x:div>Less: <x:em> &amp;lt; </x:em></x:div>
3 | </text>
```

– *the value is a single xhtml <div> element, not part of the content*

# Aggregation



- *Atom feed may include entries from another atom feed*
  - *these entries do not originally belong to this feed*
- **source** element should contain at least:
  - *required atom feed's metadata **id**, **title** and **updated***
- *retains information about an entry's source feed*

# Data and Time

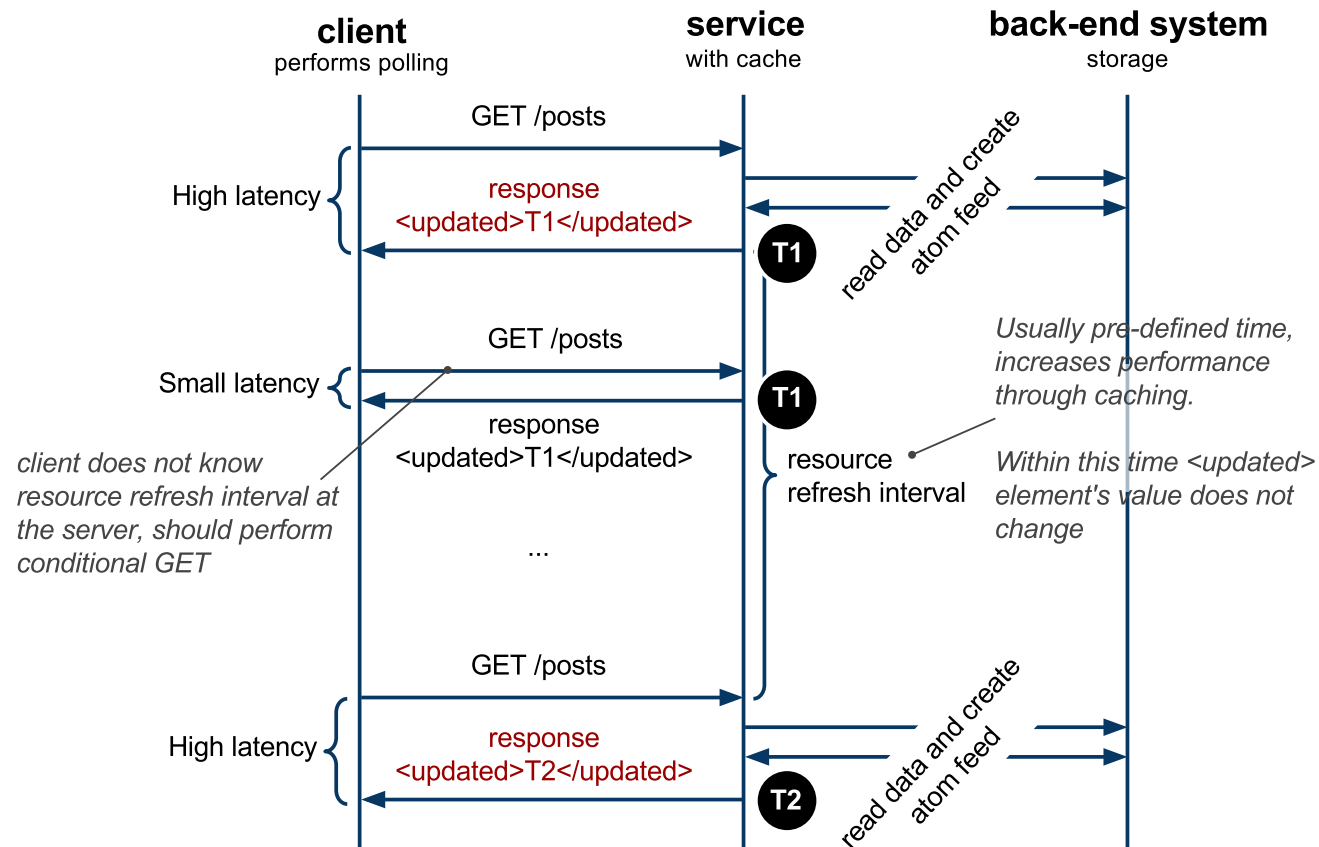
- Notion of time
  - *Atom document is a snapshot of resource in some time*
  - **updated** (feed, entry) – *last update of the resource*
  - **published** (entry) – *initial creation of the first availability of the resource*
- Data format
  - *Examples:*
    - 1 `<updated>2003-12-13</updated>`
    - 2 `<updated>2003-12-13T18:30:02.25Z</updated>`
    - 3 `<updated>2003-12-13T18:30:02.25+01:00</updated>`

**T** – *time delimiter*

**Z** – *identifies UTC time (~GMT)*

**(+|-)hh:mm** – *defines local time and a shift in hours and minutes from the UTC time*

# Polling



- **updated** is the last updated time of the resource at the server
- resource refresh interval is pre-defined by the service



# Extensions

- Possible to combine various vocabularies
  - *through namespaces `xmlns` attribute, extensions of `link.rel` attribute*
- Example: GData (PicasaWeb, Docs, ...)
  - *combines vocabularies such as Geo location*

```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <feed xmlns='http://www.w3.org/2005/Atom' xmlns:gml='http://www.opengis.net/gml'
3   xmlns:gphoto='http://schemas.google.com/photos/2007'
4   xmlns:georss='http://www.georss.org/georss'>
5   <id>http://picasaweb.google.com/.../albumid/5262593967320034641</id>
6   <updated>2010-02-25T20:47:53.295Z</updated>
7   <category
8     scheme='http://schemas.google.com/g/2005#kind'
9     term='http://schemas.google.com/photos/2007#album' />
10  <title type='text'>Památkově chráněný dům v Loukově</title>
11  <link rel='http://schemas.google.com/g/2005#feed' type='application/atom+xml'
12    href='http://picasaweb.google.com/.../albumid/5262593967320034641?hl=en_US' />
13  <link rel='http://schemas.google.com/photos/2007#slideshow'
14    type='application/x-shockwave-flash'
15    href='https://picasaweb.google.com/s/c/bin/slideshow.swf?...' />
16  <georss:where>
17    <gml:Point>
18      <gml:pos>50.5576865 15.0356436</gml:pos>
19    </gml:Point>
20  </georss:where>
21  <gphoto:allowPrints>true</gphoto:allowPrints>
22  ...
23 </feed>
```

# Overview

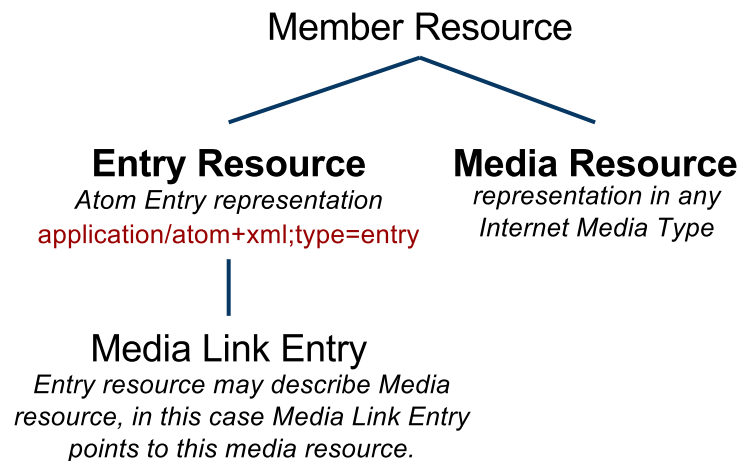
- Overview of Formats and Protocols
- Atom Syndication Format
- AtomPub Protocol
  - *Extensions*

# AtomPub Protocol

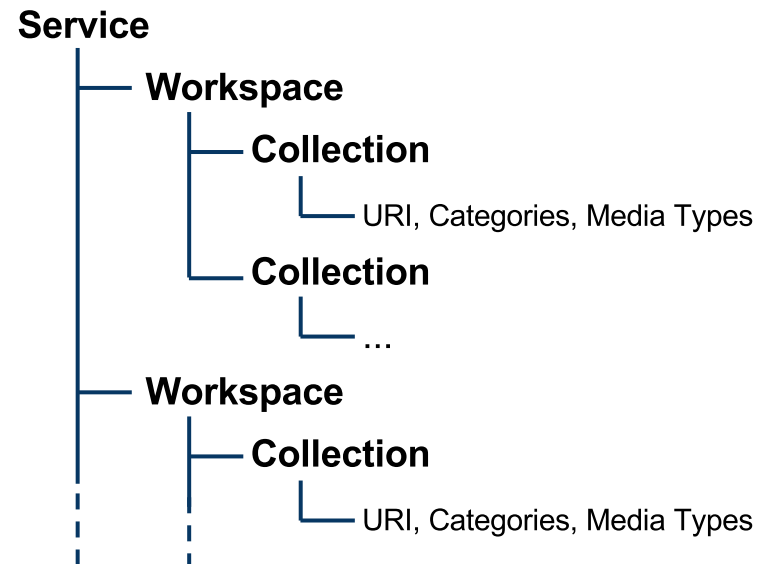
- Standard protocol for manipulation of resources
  - *Defines a service description by following constructs*
    - **service** – *a set of workspaces*
    - **workspace** – *a set of collections*
    - **collection** – *a set of resources*
  - *Defines protocol for editing, that is: creating (POST), updating (PUT), reading (GET), deleting (DELETE)*
- Relation to Atom Syndication Format
  - *Atom Feed and Atom Entry as resource representations*
- Basis for many, such as:
  - *Google Data Protocol (GData)*
  - *Microsoft Open Protocol (OData)*

# AtomPub Elements

## Types of Member Resources



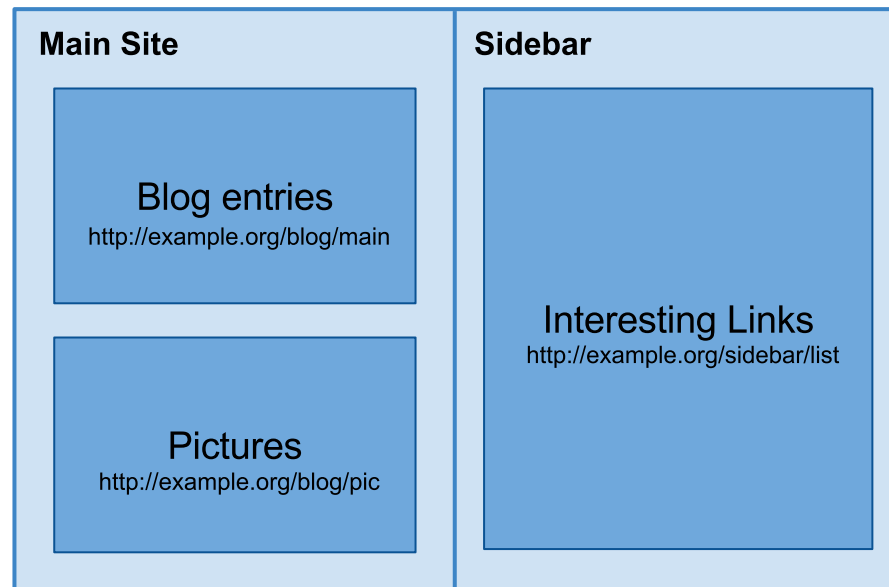
## AtomPub Service Description



- Collection properties and definition of constraints
  - **URI** – *id of the collection (Atom Feed)*
  - **categories** – *list of allowed categories in the collection*
  - **accept** – *list of Internet media types allowed in the collection*
  - **URI points to an Atom Feed resource!**

# Example Blogging Site Description

## Conceptual structure of a blogging site



- Workspaces
  - *Main Site, Sidebar*
- Collections
  - *Blog entries, pictures, interesting links*

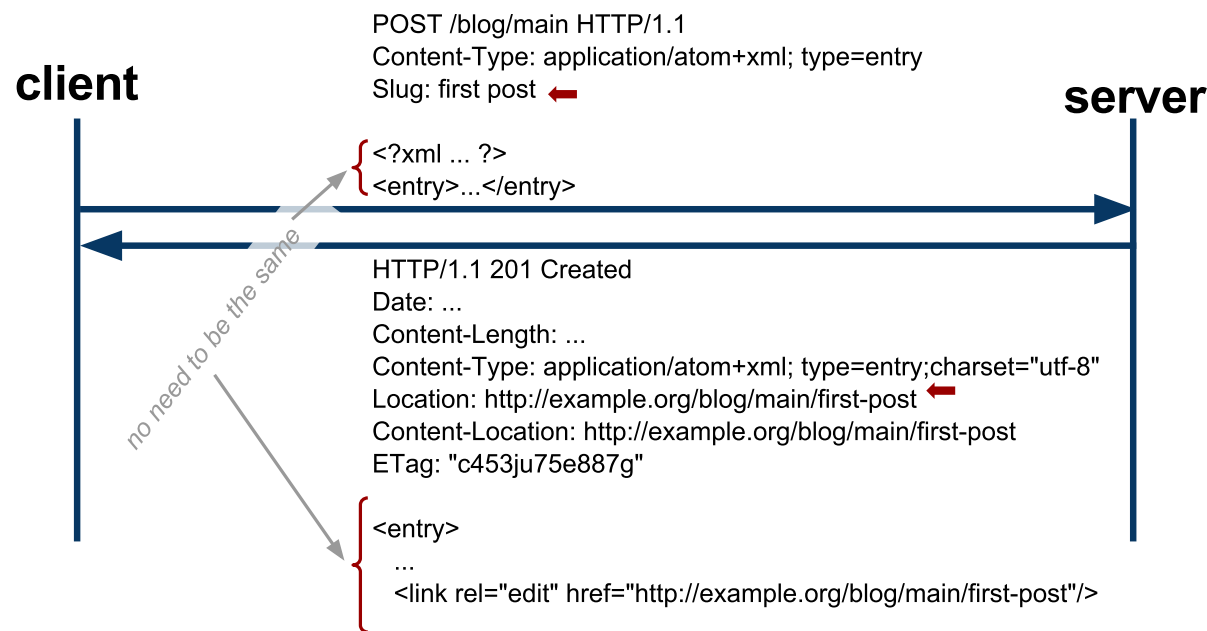
# Example Blogging Site Description

```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <service xmlns="http://www.w3.org/2007/app"
3   xmlns:atom="http://www.w3.org/2005/Atom">
4   <workspace>
5     <atom:title>Main Site</atom:title>
6     <collection href="http://example.org/blog/main">
7       <atom:title>Blog Entries</atom:title>
8       <categories
9         href="http://example.com/cats" />
10    </collection>
11    <collection href="http://example.org/blog/pic" >
12      <atom:title>Pictures</atom:title>
13      <accept>image/png</accept>
14      <accept>image/gif</accept>
15    </collection>
16  </workspace>
17  <workspace>
18    <atom:title>Sidebar</atom:title>
19    <collection href="http://example.org/blog/sidebar" >
20      <atom:title>Interesting Links</atom:title>
21      <accept>application/atom+xml;type=entry</accept>
22      <categories fixed="yes">
23        <atom:category
24          scheme="http://example.org/cats"
25          term="http://example.org/cats#joke" />
26        <atom:category
27          scheme="http://example.org/cats"
28          term="http://example.org/cats#serious" />
29      </categories>
30    </collection>
31  </workspace>
32 </service>
```

# Protocol Operations

- Operations to manipulate resources
  - *Retrieving a service document (is obvious, GET)*
  - *Listing collection members (filtering and projections)*
  - *Creating a resource (entry and media)*
  - *Editing a resource (is obvious, PUT and DELETE)*
- AtomPub **does not define:**
  - *Any manipulation with*
    - *service documents, workspaces and collections*
  - *How service documents are discovered*
- AtomPub may be used w/o service descriptions
  - *They're good for discovering constraints on the service*
  - *They're not a requirement*
  - *For example, GData does not have them*

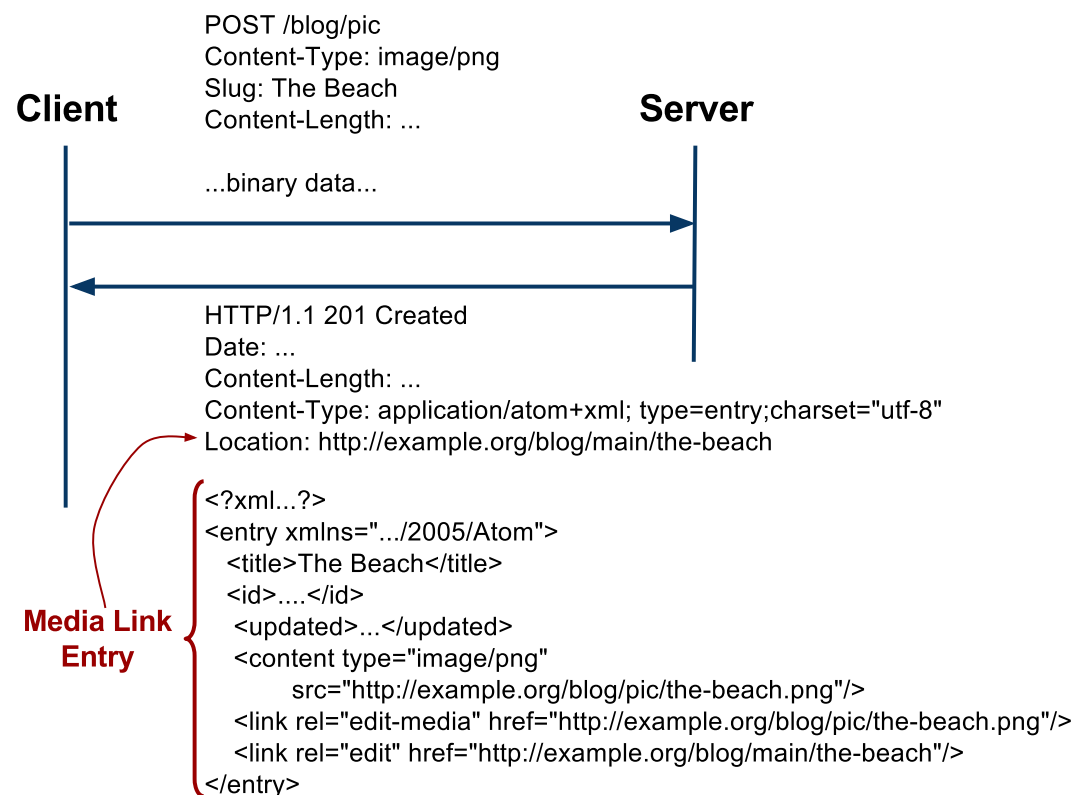
# Creating Entry Resource



- *Server checks constraints of the collection*
- *Server may modify member representation*
  - such as changes **id**, adds **updated** element
- *if **Content-Location** is not equal to **Location** the request and response representation are not the same!*
- **ETag** should be used for
  - *conditional GET and PUT (see lecture 4 – scalability)*



# Creating Media Resource



- *Server checks the constraints of the collection*
  - may return **415 Unsupported Media Type** if not accepted
- *Media Link Entry is an Entry resource that describes metadata about media resource (such as a picture)*

# Listing Collection

- Must provide representation in Atom Feed
- Contains list of Atom **Entry** elements
  - *must have **link** with attribute **edit***
  - *must have **edited**, order of entries by this date*
    - *is not the same as **Last-Modified** header*
- Entries in collection are not full representations
  - *clients should retrieve them using GET on entry URI*
- To limit amount of entries
  - *links with semantics for navigation through the whole list*

```
1 <feed xmlns="http://www.w3.org/2005/Atom">
2   <link rel="first" href="http://example.org/blog/main/" />
3   <link rel="previous" href="http://example.org/blog/main/3" />
4   <link rel="self" href="http://example.org/blog/main/4" />
5   <link rel="next" href="http://example.org/blog/main/5" />
6   <link rel="last" href="http://example.org/blog/main/10" />
7 </feed>
```

# Overview

- Overview of Formats and Protocols
- Atom Syndication Format
- AtomPub Protocol
  - *Extensions*

# Extensions

- OpenSearch
  - *Specification: OpenSearch* [🔗](#)
  - *Search service description and search results*
- Google Data Protocol
  - *Filtering, partial response and partial update*
  - *Entity tag attribute for **<feed>** and **<entry>** elements*
  - *HTTP methods overriding*

# OpenSearch

- Open Search Specification
  - *Open Search Description Document (OSDD)*
    - *description of a search service*
  - *OpenSearch Response Document*
    - *Standard description of search results by search services*
    - *extension of syndication formats, RSS and Atom*
- Adoption
  - *Browsers such as IE, Google Chrome – search engines you can use to search various sites.*
  - *APIs such as Bing API, Google Docs, etc. – description of search results.*

# OpenSearch Description Document

- Example:

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1.1/">
3    <ShortName>Web Search</ShortName>
4    <Description>Use Example.com to search the Web.</Description>
5    <Tags>example web</Tags>
6    <Contact>admin@example.com</Contact>
7    <Url type="application/atom+xml"
8        template="http://example.com/?q={searchTerms}&pw={startPage?}&format=atom"/>
9    <Url type="application/rss+xml"
10       template="http://example.com/?q={searchTerms}&pw={startPage?}&format=rss"/>
11    <Url type="text/html"
12       template="http://example.com/?q={searchTerms}&pw={startPage?}"/>
13    <Image height="64" width="64" type="image/png">
14      http://example.com/websearch.png
15    </Image>
16    <Query role="example" searchTerms="cat" />
17    <Developer>Example.com Development Team</Developer>
18    <AdultContent>false</AdultContent>
19    <Language>en-us</Language>
20    <OutputEncoding>UTF-8</OutputEncoding>
21    <InputEncoding>UTF-8</InputEncoding>
22  </OpenSearchDescription>
```

— **searchTerms** *is a free text*

# OpenSearch Response Document

- Example:
  - *Result in Atom format of a search query*

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <feed xmlns="http://www.w3.org/2005/Atom"
3      xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/">
4      <title>Example.com Search: New York history
5      <updated>2003-12-13T18:30:02Z</updated>
6      <author>
7          <name>Example.com, Inc.</name>
8      </author>
9      <id>urn:uuid:60a76c80-d399-11d9-b93C-0003939e0af6</id>
10     <opensearch:totalResults>4230000</opensearch:totalResults>
11     <opensearch:startIndex>21</opensearch:startIndex>
12     <opensearch:itemsPerPage>10</opensearch:itemsPerPage>
13     <opensearch:Query role="request" searchTerms="New York History" />
14     ...
15     <link rel="search" type="application/opensearchdescription+xml"
16         href="http://example.com/opensearchdescription.xml"/>
17     <entry>
18         <title>New York History</title>
19         ...
20     </entry>
21 </feed>
22
```

# GData Protocol: Advanced Search Query

- OpenSearch does not specify syntax for search query
  - *It can be anything, free text*
  - *GData Protocol further allows for filtering and projection*
- Filtering
  - *Fine-grained conditions based on values of various elements*
    - *such as **author**, **category**, **max-results**, min and max of **published** and **updated** elements.*
  - 1 | `http://www.example.com/feeds/jo?q=Darcy&updated-min=2005-04-19T15:30:00Z`
  - 2 | `http://www.example.com/feeds?category=Fritz%7CLaurie` // URL encoded OR
  - 3 | `http://www.example.com/feeds?category=Fritz,CLaurie` // AND
- Partial Response (~Projection)
  - *Which elements of an entry should appear in the search result*
  - *A language based on XPath syntax (subset of a valid XPath expression)*
    - 1 | `http://example.org/blog/main?fields=link,entry(@gd:etag,updated,link[@rel='edit'])`



# GData Protocol: Partial Update

- **PATCH** HTTP Method
  - *IETF specification, see PATCH Method for HTTP* [🔗](#)
  - *Add, modify or delete selected elements of an entry*
- Examples
  - *To delete a description element and add a new title element*
  - **gd:fields** uses partial response syntax

```
1 PATCH /myFeed/1/1/  
2 Content-Type: application/xml  
3  
4 <entry xmlns='http://www.w3.org/2005/Atom'  
5     xmlns:gd='http://schemas.google.com/g/2005'  
6     gd:fields='description'>  
7     <title>New title</title>  
8 </entry>
```

- Rules
  - *Fields not already present are added*
  - *Non-repeating fields already present are updated*
  - *Repeating fields already present are appended*

# GData Protocol: Entity Tags

- Resource Versioning
  - *Conditional GET and PUT (concurrency control)*
    - See *Lecture 4 – scalability*
  - *Etags on atom and entry elements*

- Example

```
1  GData-Version: 2.0
2  ETag: W/"C0QBRXcycSp7ImA9WxRVFuk."
3  ...
4  <?xml version='1.0' encoding='utf-8'?>
5  <feed xmlns='http://www.w3.org/2005/Atom'
6      xmlns:gd='http://schemas.google.com/g/2005'
7      gd:etag='W/"C0QBRXcycSp7ImA9WxRVFuk."'>
8      ...
9      <entry gd:etag=' "CUUEQX47eCp7ImA9WxRVEkQ." '>
10         ...
11         </entry>
12     </feed>
13
```

- *It is possible to do a conditional GET/PUT on the entry by using the ETag "CUUEQX47eCp7ImA9WxRVEkQ."*

# GData Protocol: HTTP Methods Overriding

- Firewall restrictions
  - *Some firewall configurations do not allow to send HTTP request other than GET and POST*

- HTTP methods overriding through **POST**

X-HTTP-Method-Override: PUT  
X-HTTP-Method-Override: DELETE  
X-HTTP-Method-Override: PATCH

- Example

```
1 | POST /myfeed/1/1/  
2 | X-HTTP-Method-Override: PATCH  
3 | Content-Type: application/xml  
4 | ...
```