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

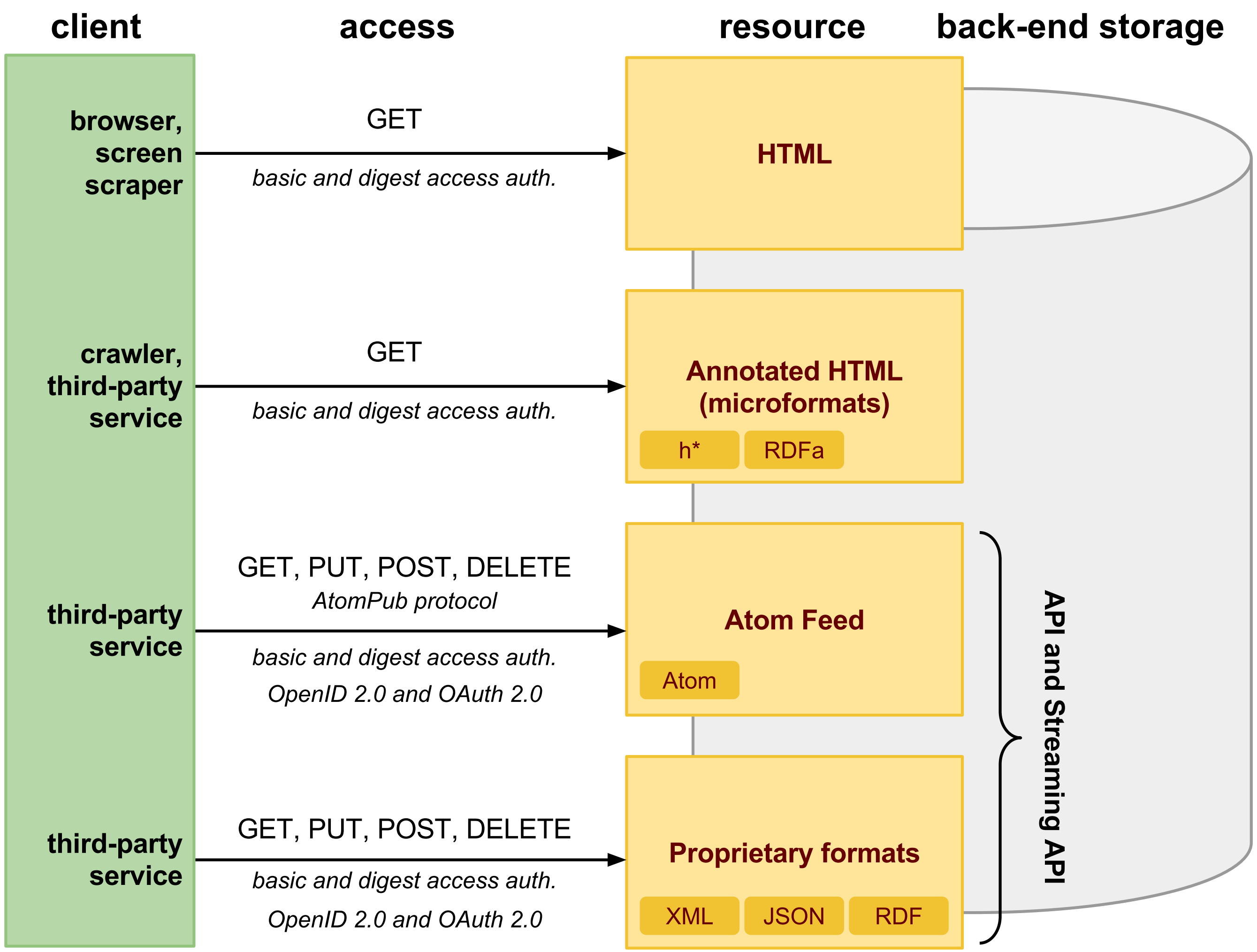
Evropský sociální fond
Praha & EU: Investujeme do vaší budoucnosti

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

Overview

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

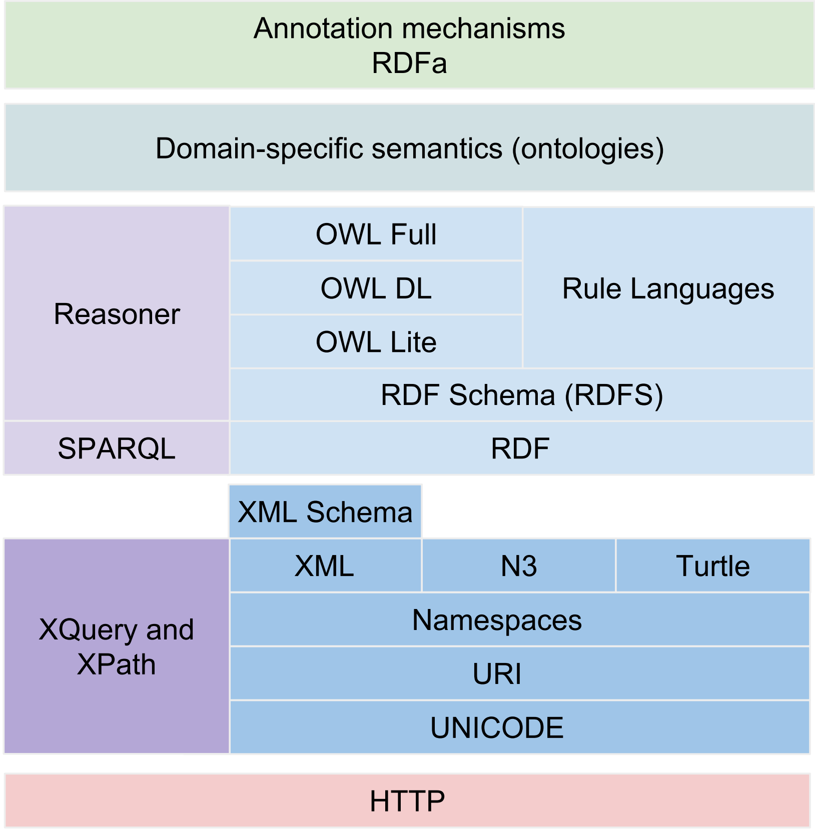
Data on the Web



Data Syntax, Structure and Semantics

Semantic Web Layered Cake

syntax and formal semantics



Annotation mechanisms
RDFa

Domain-specific semantics (ontologies)

Reasoner

OWL Full

OWL DL

OWL Lite

Rule Languages

RDF Schema (RDFS)

SPARQL

RDF

XML Schema

XQuery and XPath

XML

N3

Turtle

Namespaces

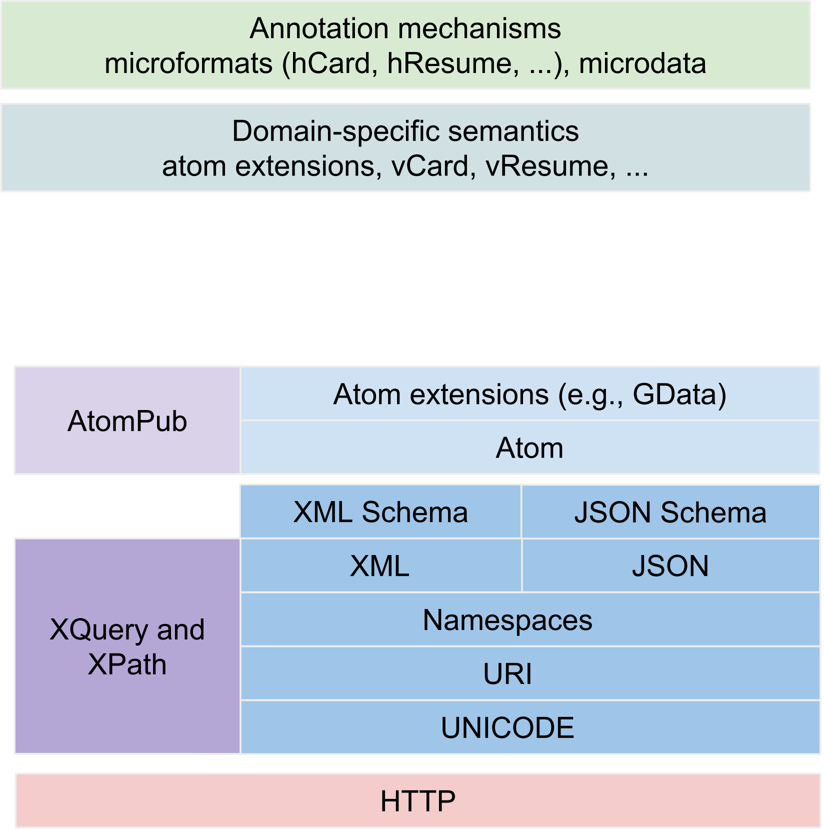
URI

UNICODE

HTTP

Web Data Formats

syntax and semantics (structure)



Annotation mechanisms
microformats (hCard, hResume, ...), microdata

Domain-specific semantics
atom extensions, vCard, vResume, ...

AtomPub

Atom extensions (e.g., GData)

Atom

XML Schema

JSON Schema

XQuery and XPath

XML

JSON

Namespaces

URI

UNICODE

HTTP

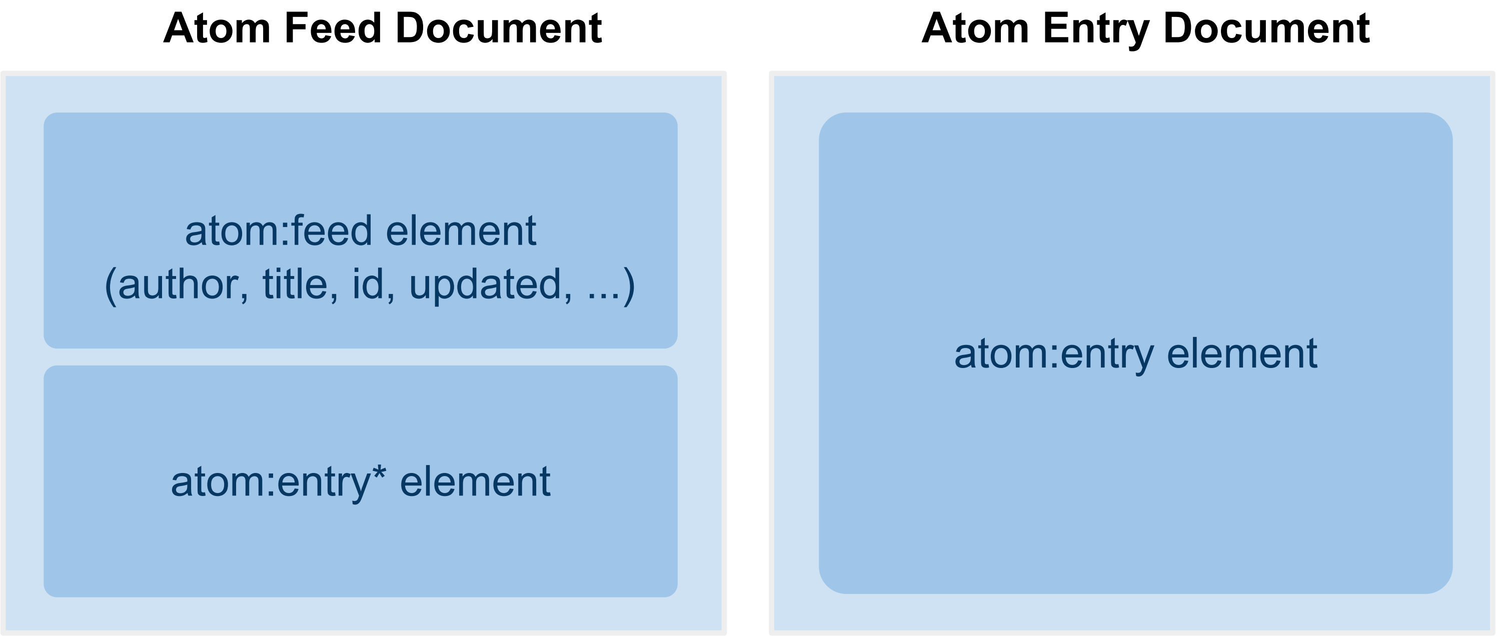
Atom Standard

- 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*
- IETF Atom Publishing Format and Protocol WG
  - *RFC 4287:*
  - *RFC 5023:*
- 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



- 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

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 occurencies – repeating fields*

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

  `( )`*: exactly one occurence – non-repeating fields*

Atom Elements – Atom Entry

- `atom:entry` element

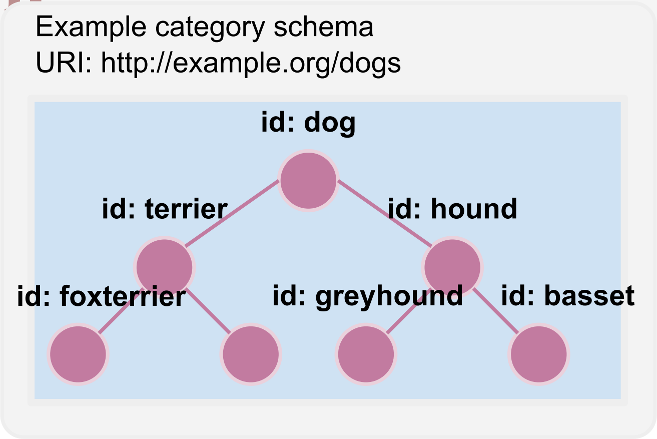
  `(*)`: *zero or more occurencies – repeating fields*

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

  `( )`: *exactly one occurence – non-repeating fields*

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:*


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*

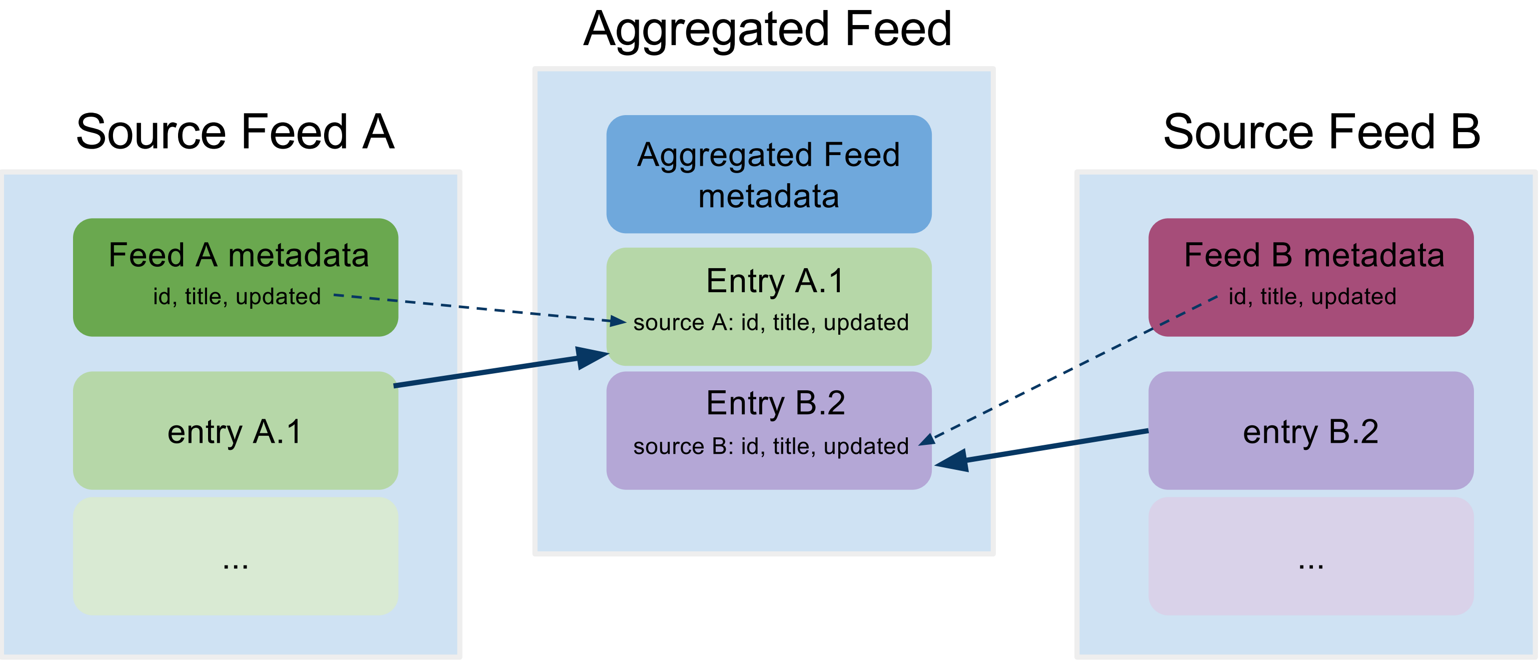
Atom Links

- 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*
  - *More details in Lecture 4 – HATEOAS.*

Encoding Textual Content

- Plain text
  - *simple text, must not contain child elements*
- HTML
  - *html text, must not contain child elements*
  - *any markup must be escaped,*
  - *should be possible to display it as HTML inside* `<div>` *element*
- XHTML
  - *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

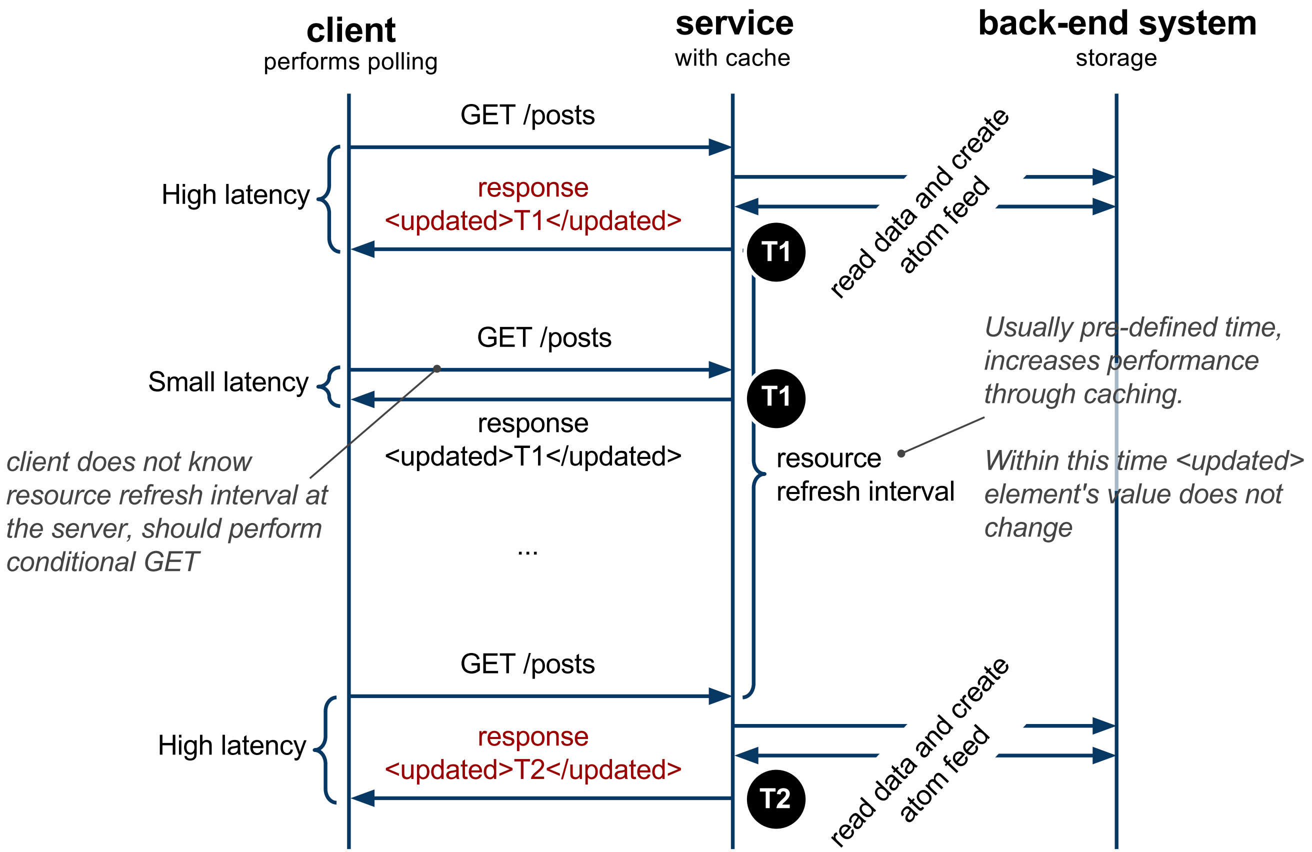
- Notion of time
  - *Atom document is a snapshot of resource in some time*
  - `updated` *(feed, entry) – last update of the resource*
  - `published` *(entry) – initial creation of the first availability of the resource*
- Data format
  - *Examples:*

    `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 serice*

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*

Overview

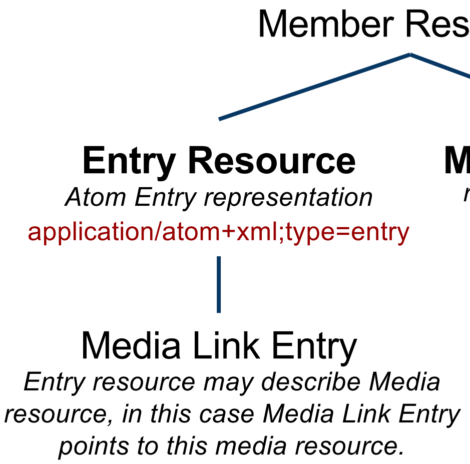
- Overview of Formats and Protocols
- Atom Syndication Format
- AtomPub Protocol
  - *Extensions*

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



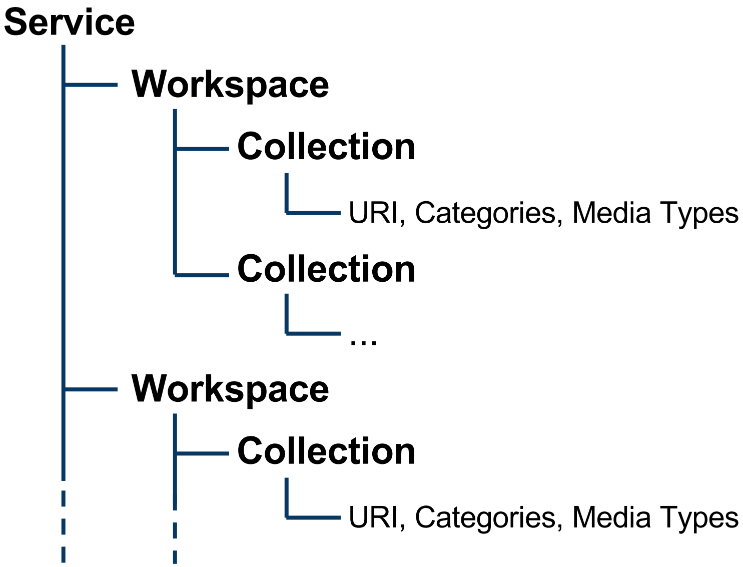
Member Resource

Entry Resource
Atom Entry representation
application/atom+xml;type=entry

Media Resource
representation in any Internet Media Type

Media Link Entry
Entry resource may describe Media resource, in this case Media Link Entry points to this media resource.

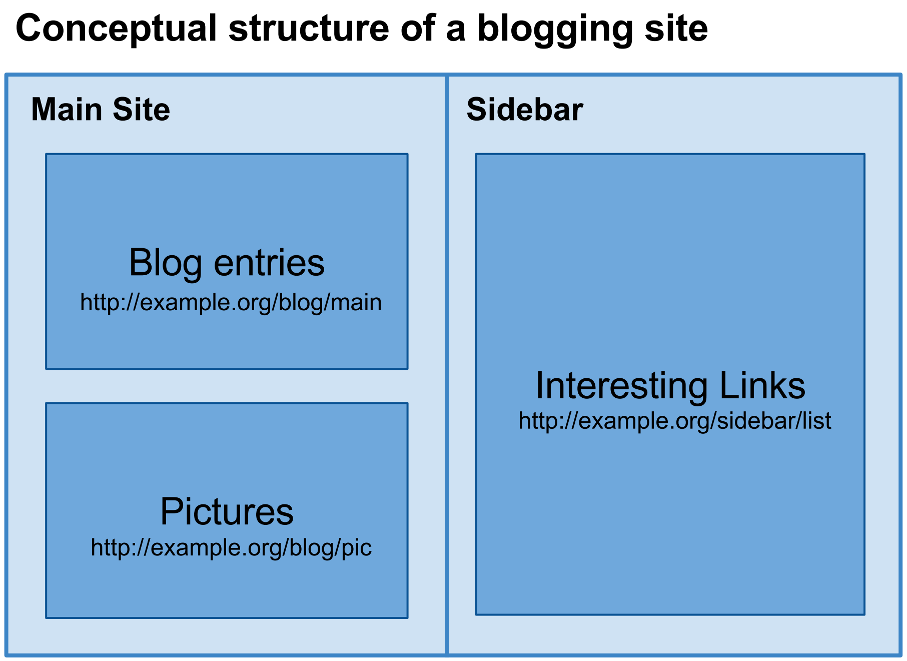
AtomPub Service Description



Service
Workspace
Collection
URI, Categories, Media Types
Collection
...
Workspace
Collection
URI, Categories, Media Types

- 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
Main Site
Sidebar
Blog entries
http://example.org/blog/main
Pictures
http://example.org/blog/pic
Interesting Links
http://example.org/sidebar/list

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

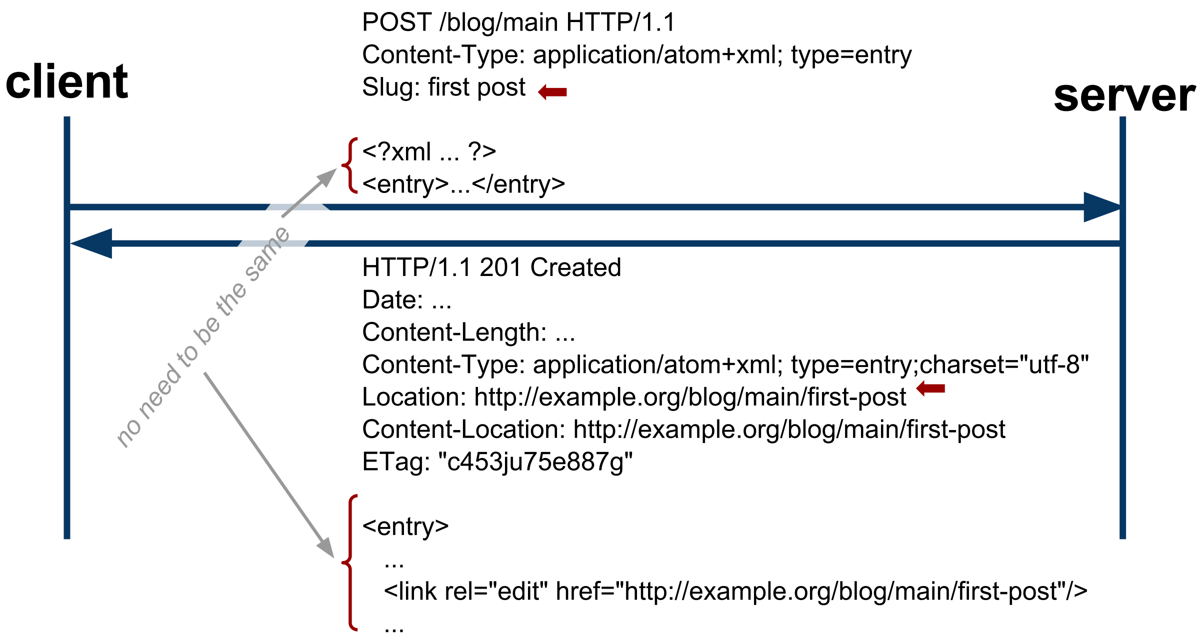
Example Blogging Site Description

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



client

server

```
POST /blog/main HTTP/1.1
Content-Type: application/atom+xml; type=entry
Slug: first post

<?xml ... ?>
<entry>...</entry>
```

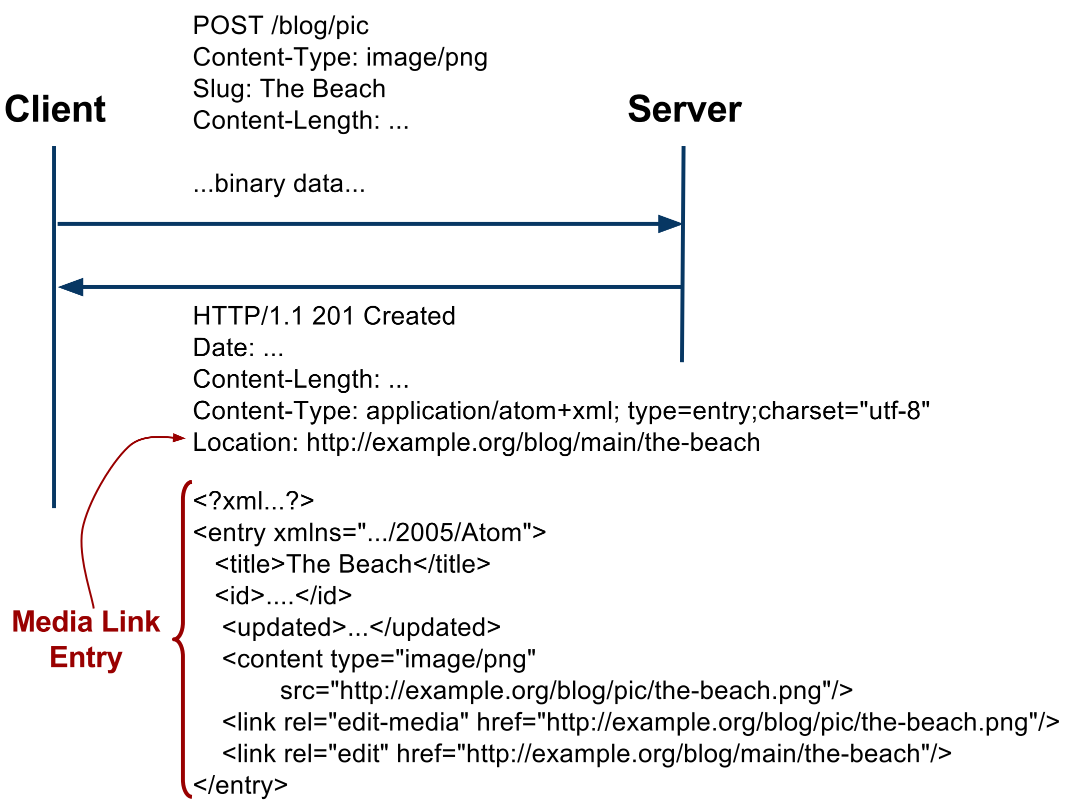
no need to be the same

```
HTTP/1.1 201 Created
Date: ...
Content-Length: ...
Content-Type: application/atom+xml; type=entry;charset="utf-8"
Location: http://example.org/blog/main/first-post
Content-Location: http://example.org/blog/main/first-post
ETag: "c453ju75e887g"

<entry>
  ...
  <link rel="edit" href="http://example.org/blog/main/first-post"/>
  ...
```

- *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



```
POST /blog/pic
Content-Type: image/png
Slug: The Beach
Content-Length: ...

...binary data...
```

```
HTTP/1.1 201 Created
Date: ...
Content-Length: ...
Content-Type: application/atom+xml; type=entry;charset="utf-8"
Location: http://example.org/blog/main/the-beach

<?xml...?>
<entry xmlns=".../2005/Atom">
   <title>The Beach</title>
   <id>....</id>
    <updated>...</updated>
    <content type="image/png"
            src="http://example.org/blog/pic/the-beach.png"/>
    <link rel="edit-media" href="http://example.org/blog/pic/the-beach.png"/>
    <link rel="edit" href="http://example.org/blog/main/the-beach"/>
</entry>
```

– *Server checks the constraints of the collection*

 → *may return* `415 Unsupported Media Type` *if not accapted*

– *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*

Overview

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

Extensions

- OpenSearch
  - *Specification:*
  - *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:
  - `searchTerms` *is a free text*

OpenSearch Response Document

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

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.*
- 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)*

GData Protocol: Partial Update

- `PATCH` HTTP Method
  - *IETF specification, see*
  - *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*
- 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 (concurrencyl control)*

    → *See Lecture 4 – scalability*
  - *Etgas on atom and entry elements*
- Example
  - *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`
- Example