Thomas Koch

`thomas@koch.ro`

matriculation number 7250371

January 21, 2012
Fernuniversität Hagen
Faculty of mathematics and computer science

Contents

1. Motivation

Experiences with Android and Data in the cloud `http://keithp.com/blogs/calypso`
Why vCard/CardDav: many clients
Why OpenSocial / Portable Contacts:

- used by Google, LinkedIn,

- used in Enterprise applications like Attlassian tools (Jira, Confluence, ...), Nuxeo CMS, ...

- OpenSocial can be used to implement inhouse portals and populate it with data from the companies GroupWare

2. Advantages of REST for Groupware data

The REST style is motivated by the argument that its application would help to provide certain attributes for an architecture.[Fie00, sec 5.1]
Maybe not so important for a companies internal Groupware: Scalability (in terms of users), Network performance, Efficiency,

- Cachability can keep the data available also in offline mode

- Simplicity helps to develop glue code between different systems

- Modifiability allows to adapt the Groupware to changes in the organization

- Reliability

- Administrativ scalability

3. Evaluation of APIs

3.0.1. Algermissen's Classification of HTTP-based APIs

Jan Algermissen, proposes a "Classification of HTTP-based APIs" in February 2010.[1] He identifies and names a five level order of HTTP based APIs. Each level adds one constraint to be obeyed. The last level obeys all five constraints and is RESTful. (Table 1)
In addition to that, Algermissen also provides a list of "Effect on System Properties and Cost" of the different API styles and acknowledges that the initial costs of developing a REST API may be higher compared to the other styles but lower in the long run.

4. Existing Groupware data APIs

4.1. IMAP (Kolab)

Kolab uses an IMAP server as the data store and synchronization protocol for calendar and contact informations. I want to compare this approach to a restful one.
Advantages of IMAP:

- already there, since Mail uses it

---

[1] `http://nordsc.com/ext/classification_of_http_based_apis.html` (2011-12-08)

level name	constraint violated	violation description
WS-*	Identification of Resources	Only service endpoint is identified by URI. No resources exposed.
RPC URI-Tunneling	Manipulation of Resources through Representations	SOAP body contains operation name, message not transferred to manipulate resource state.
HTTP-based Type I	Self-Descriptive Messages	Message semantics depend on action specified in message body.
HTTP-based Type II	Hypermedia as the Engine of Application State	Application state machine known at design time.

Table 1: Classification of HTTP-based APIs after Algermissen

- can store blobs/files so no need to map the iCal/vCard files to a relational scheme

- out of the box support for offline work and later synchronization (How does it solve editing conflicts?)

Disadvantages of IMAP:

- Complicate, 38 RFCs according to `http://de.wikipedia.org/wiki/Internet_Message_Access_Protocol` see also: `http://www.apps.ietf.org/rfc/ipoplist.html`

- All clients directly access the iCal/vCard files with no moderation layer in between. This means that no validation or normalization can be done. Schema updates can only be done if all clients cooperate.

- IMAP imposes a folder structure. Google's gmail is an example for another, tag based approach. Messages could have several tags. It is therefor hard to access Gmail via IMAP.

- Sam Varshavchik, the author of the courier Mail Transfer Agent explains the history of IMAP and claims that the IMAP standard is broken: http://www.courier-mta.org/fud/

- IMAP is so complicate that the IMAP wiki holds 10 pages of advises for IMAP client authors: http://www.imapwiki.org/ClientImplementation RFC 2683 "IMAP4 Implementation Recommendations" is a 23 pages document (cut 5 pages for verbosity) explaining how to implement another RFC standard. Is there any widely used standard that needs another RFC explaining how to implement it?

- `http://en.wikipedia.org/wiki/Internet_Message_Access_Protocol#Disadvantages`

- Some attempts to create a simpler alternative to IMAP:
    - http://en.wikipedia.org/wiki/POP4
    - `http://en.wikipedia.org/wiki/Simple_Mail_Access_Protocol` also here http://www.comta.org/cone/smap1.html
    - `http://en.wikipedia.org/wiki/Internet_Mail_2000`
    - HTTP restful: http://tools.ietf.org/id/draft-dusseault-httpmail-00.txt mailing list: https://www.ietf.org/mailman/listinfo/httpmail
    - BikINI is not IMAP http://bikini.caterva.org

- Outlook uses HTTP to communicate with Hotmail

- another rest mail proposal: http://www.prescod.net/rest/restmail/

- more rants: http://blog.gaborcselle.com/2010/02/how-to-replace-imap.html

- IMAP issues found by the chandler project http://chandlerproject.org/bin/view/Jungle/IntrinsicI

4.2. WebDAV, CalDAV, CardDAV

4.2.1. Is WebDAV restful?

Roy Fielding says no: `http://tech.groups.yahoo.com/group/rest-discuss/message/` `5874`

> PROP* methods conflict with REST because they prevent important resources from having URIs and effectively double the number of methods for no good reason. Both Henrik and I argued against those methods at the time. It really doesn't matter how uniform they are because they break other aspects of the overall model, leading to further complications in versioning (WebDAV versioning is hopelessly complicated), access control (WebDAV ACLs are completely wrong for HTTP), and just about every other extension to WebDAV that has been proposed.
> [...]
> The problem with MOVE is that it is actually an operation on two independent namespaces (the source collection and destination collection). The user must have permission to remove from the source collection and add to the destination collection, which can be a bit of a problem if they are in different authentication realms. COPY has a similar problem, but at least in that case only one namespace is modified. I don't think either of them map very well to HTTP.

The discussion also continued on the microformats mailing list `http://microformats.` `org/discuss/mail/microformats-rest/2006-April/thread.html#217`.

see [Amu10] for a restful approach to properties.

Is ATOM an alternative to WebDAV?

> AtomPub is different from DAV in two key respects:
> - The client doesn't control where things go, the server does
> - It is allowed and expected that an AtomPub server will look at the incoming information and change it (generate ID, timestamps, sanitize HTML, etc)

Tim Bray, http://www.imc.org/atom-protocol/mail-archive/msg11271.html

4.3. PortableContacts aka OpenSocial

4.3.1. Is OpenSocial restful?

Roy Fielding wrote a blog post about the "SocialSite REST API", stating that it isn't restful at all but clearly an RPC style API.[2] Fielding was referring to SocialSite, which is however an implementation of the OpenSocial specification. Dave Johnson, a contributor to SocialSite, reacted on this critique by opening a discussion on an OpenSocial mailing list:[3]

---

[2]`http://roy.gbiv.com/untangled/2008/rest-apis-must-be-hyperhtext-driven` (2011-12-06)

[3]`http://groups.google.com/group/opensocial-and-gadgets-spec/browse_thread/thread/` `aff4ba7373e21284/201a413efa67c26e` (2011-12-06)

> I must admit, it is not clear to me how OpenSocial REST API violates the six rules that Roy stated.

The above quote warrants a short comment. I also thought before, that REST would be so simple that there wouldn't be much need for further studying. Every web developer has some understanding of URIs, HTTP and a bit less of Hypermedia. So it is easy to fall into the trap that everything build on top of HTTP would be restful. Now however, after some more reading about REST, I can easily find violations of the REST constraints in the OpenSocial specification.

Restful APIs are modeled around resources, their representations and links between them. The authors of the OpenSocial API however seem to have modeled their API around the concept of services:[Ope11, Social API Server, sec 2, Services]

> OpenSocial defines several services for providing access to a container's data.

Fielding's critique Fielding listed some rules that a restful API must obey, but did not give explicit examples how OpenSocial violates this rules. The following section will provide such examples.

> A REST API should not be dependent on any single communication protocol, [. . .] any protocol element that uses a URI for identification must allow any URI scheme to be used for the sake of that identification. *[Failure here implies that identification is not separated from interaction.]*

OpenSocial defines a construct called "REST-URI-Fragment" which is a clear violation of the above rule. This URI fragment is simply an encoding of procedural parameters as elements of an HTTP URI:[Ope11, Core API Server, sec 2.1.1.2.2, REST-URI-Fragment]

> Each service type defines an associated partial URI format. The base URI for each service is found in the URI element associated with the service in the discovery document. Each service type accepts parameters via the URL path. Definitions are of the form:
>
> `{a}/{b}/{c}`

URIs can contain a query component that would be more appropriate to contain parameters. This would also have made it clearer to see that the specification actually defines services instead of resources. One test showing the misfit is to ask how dot-segments ('.' and '..') inside the URI fragment are interpreted and whether this conforms with the letter and spirit of the URI standard.[BLFM05, sec 3.3] Another misfit can be seen in the URI fragment to retrieve one or multiple albums. In this case the 'c' part in the quoted definition above is actually a list of albums to retrieve separated by a slash.

Fielding's second bullet point most likely refers to the `X-HTTP-Method-Override` header. This header is a widely used[4] workaround to allow the use of other HTTP methods than GET and POST from HTML forms or through firewalls.

The next two points again refer to a more serious issue:

> A REST API should spend almost all of its descriptive effort in defining the media type(s) used for representing resources and driving application state[. . .]. *[Failure here implies that out-of-band information is driving interaction instead of hypertext.]* A REST API must not define fixed resource names or hierarchies[. . .] *[Failure here implies that clients are assuming a resource structure due to out-of band information[. . .]].*

---

[4] `http://www.subbu.org/blog/2008/07/another-rest-anti-pattern` (2011-12-06)

The OpenSocial specification contains a lot of out-of-band information describing how to form URIs to access information or which methods to use on which URIs for different actions. This means that the OpenSocial API is not simple or intuitive to use but requires a client developer to read a lot of specification, thus violating the simplicity property of a restful architecture. Since the URIs are fixed in the specification and necessarily also in clients, the modifiability property is also violated.[Fie00, sec 2.3]

The following tables give some examples of the specified URIs:

URI fragment	Method	Description
/people/{User-Id}/@self	GET	profile for User-Id
/people/{User-Id}/@self	DELETE	remove User
/people/{User-Id}/{Group-Id}	GET	full profiles of group members
	POST	Create relationship, target specified by `<entry><id>` in body
	POST	Update Person
/people/{Initial-User-Id}/ {Group-Id}/{Related-User-Id}	GET	???
/people/@supportedFields	GET	list of supported person profile fields
/groups/{User-Id}[/{Group-Id}]	GET	one or all groups of a user
	PUT	update group
	DELETE	delete group
/groups/{User-Id}	POST	create group

Table 2: URI fragments for peoples and groups in the OpenSocial REST API

URI fragment	Method	Description
/albums/{User-Id}/@self	POST	create album
/albums/{User-Id}/{Group-Id}[/Album-Id]*	GET	one or multiple albums
/mediaItems/{User-Id}/{Group-Id}/{Album-Id}/ {MediaItem-Id}	GET	one mediaitem
/mediaItem/{User-Id}/@self/{Album-Id} (sic!)	POST	create mediaitem

Table 3: URI fragments for albums and mediaitems in the OpenSocial REST API

The last URI in Table 3 is obviously missing an "s" behind `mediaItem`. This typo is present and unfixed in the OpenSocial spec since Version 1.0, released in march 2010. This is of course not a big issue in itself, but rather a sign that the specification is too verbose and does over-specify things that should rather be auto-discovered through hyperlinks.

Fielding mentions in a comment to the same blog post that the OpenSocial API "could be made so [restful] with some relatively small changes" but does not specify these changes. However some issues can easily be identified.

First the data structures defined in OpenSocial do not use URIs to refer to other resources. Instead they use Object-Ids that must then be inserted in the appropriate URI templates. Examples are the `recipients`, `senderId`, `collectionIds` of messages and the `ownerId` of albums. The person structure does not contain fields referencing other resources. Thus it does not obviously violate REST like the albums and messages. However it does so even worse since there are hidden references only defined out-of-band in the specification. One can retrieve the albums, relations or messages of a user by filling in the `userId` in one of the specified URI templates. If Users would just contain references to other resources related to a user, the specification could already be shortened a lot.

Another missed opportunity for a much more intuitive API is the relation of media items and albums. This seems to be poster child example for a collection (album) to collection-

element (media item) relation which could have made use of the hierarchical character of URI paths. OpenSocial however requires the client developer to use two different URI templates. (Table 3)

A not so small change to OpenSocial would be to either use already standardized and registered media types where possible or to register new types where necessary. It seems that there are some already existing media types that could be a good fit for OpenSocial but only miss a canonical json representation for easy consumption by javascript applications. These are vCard for persons,[5] ATOM entries[NS05] for messages, activities and media items and ATOM categories, collections or workspaces[Gh07] for albums and groups. It would probably be necessary to add extensions to the mentioned media types but vCard and ATOM both already anticipated this need and provided mechanisms to do so.

The use of the ATOM format could promote the adoption of OpenSocial because developers could either reuse existing knowledge about ATOM or would be more motivated to learn about a system that is based on an already widespread format. In fact OpenSocial already mentions ATOM as a way to wrap OpenSocial data. However this wrapping does not build extend and reuse ATOM semantics as proposed above but just puts the OpenSocial data structures inside the entry/content element of ATOM. This kind of misuse of ATOM does of course not deliver any advantage on top of the existing plain JSON or XML representations.[6] Consequently the newest OpenSocial specification deprecates any reference to the ATOM format.

In Algermissen's classification (3.0.1), the OpenSocial REST API would actually be "HTTP-based Type I" due to the lack of media types and direct hyper links between related resources. Algermissen writes that this level has the lowest possible initial cost of all HTTP APIs. Or in other words: The OpenSocial specification authors did not have to invest a lot to come up with this API specification but maintenance and evolution cost may be medium or high.

4.4. Others

> [. . .] CAP failed because most of the active implementors at the time felt it had become too complex to implement - partly because it required implementing a "brand new" protocol (BEEP). The upshot of that was the CalDAV effort, which was based on an existing fairly well understood protocol - WebDAV. One of the benefits of using WebDAV was that it was easy to take "off the shelf" WebDAV software (server and client) and get a basic CalDAV implementation done very quickly. Indeed, several months after CalDAV was published, CalConnect held an interop event at which several server and client implementations were present and demonstrated basic interoperability - something that would have been hard to achieve with CAP.

– Cyrus Daboo[7]

5. ATOM

Google's Data API extends ATOM? Calendar feeds are available as ATOM with the published entry set to the date of the event.

---

[5] OpenSocial persons are based on portable contacts which in turn borrowed field names from vCards.

[6] compare Bill de Hora, Extensions v Envelopes. 11/2009 `http://www.dehora.net/journal/2009/11/28/extensions-v-envelopes` (2011-21-07)

[7] `http://lists.calconnect.org/pipermail/caldeveloper-l/2012-January/000135.html` (2012-01-04)

WebDAV vs. ATOM: `http://intertwingly.net/wiki/pie/WebDav` `http://intertwingly.net/wiki/pie/WebDavVsAtom` google webdav atom

Why didn't ATOM succeed (more)? `http://bitworking.org/news/425/atompub-is-a-failure` (2012-01-06)

> Were I to propose CalDAV today it would probably be CalAtom

– Lisa Dusseault, February 29, 2008[8]

> I gave a big sigh of relief when I read that, and I hope that the CardDAV folks take this to heart. Some parts of WebDAV (e.g., properties [...]) deserve to be taken out back and shot – although, as Lisa says, they were necessary because of the state of the art at the time. That doesn't mean we can't do better now.

– Ted Leung, March 6, 2008[9]

6. Design and implementation of a restful Contacts API

6.1. Requirements

These requirements include the features of CardDAV.[Dab11, sec. 1] The CardDAV RFC mentions its disadvantage of a missing specification for "change notification". An ATOM based approach on the other hand can make use of already existing protocols and infrastructure for this purpose, most notable Pubsubhubbub.[10]

Versioning and Locking, which are provided by WebDAV are not a requirement. Optimistic locking as provided by "conditional" put requests in HTTP is preferred first because it is an already widespread standard.

6.2. Related Work

Rob Yates `mailto:robert_yates@us.ibm.com`: CalATOM `http://robubu.com/?p=6` (2012-01-05) Draft: CardATOM `http://robubu.com/?p=10` (2012-01-05)

Rob Yates drafted a CalATOM specification.[Yat07] It references another old draft (draft-snell-atompub-feature-12) to mark a collection as a CalATOM collection. This seems unnecessary since we can specify that a collection accepts calendar resources and thus marking a collection as a calendar.

Furthermore the draft mentions a capability to query for events in a given time range. There is no equivalent for such a request for contacts.

In rest the draft only repeats or references the atompub standard and the xcal format. This draft could therefor be seen as indication that the existing standards are almost sufficient to define GroupWare APIs.

DOAP-over-ATOM `http://web.archive.org/web/20070927230857/http://www.codezoo.com/about/doap_over_atom.csp` (2012-01-05)

Google Data API also uses ATOM, but extends the ATOM format with new elements instead of putting all of the content in the content element of an atom entry.[11]

---

[8] `http://nih.blogspot.com/2008/02/nearly-two-years-ago-i-made-prediction.html` (2012-1-6)

[9] `http://www.sauria.com/blog/2008/03/06/google-contacts-and-carddav/` (2012-1-6)

[10] `http://code.google.com/p/pubsubhubbub/` (2012-1-5)

[11] `http://web.archive.org/web/20081120001246/http://www.snellspace.com/wp/?p=314` (2012-01-05)

7. Persistency for Groupware Data

Relational Databases vs. NoSQL databases vs. plain files

Relational databases are not practical for contacts, events or todos. Common patterns in systems that use relational DBs for that purpose:

- artificial limits of entries, e.g. only 3 email addresses per contact, because there are only three columns email1, email2 and email3.

- Fields for custom data like custom1 to customX

- EAV pattern: tables like: id, foreign_id, type, value

8. Synchronizing a large collection

How to efficiently synchronize a large collection of contacts with the server without checking each contact for changes?

Portable Contacts has a filter "updatedSince".

How is synchronization done in CardDAV?

"Feed Paging for Efficient Feed Synchronization" thread at ATOM-Syntax `http://www.imc.org/atom-syntax/mail-archive/msg20060.html`

HTTP Delata encoding with Feed: `http://www.wyman.us/main/2004/09/using_rfc3229_w.html` (2012-1-6), implementations: `http://www.wyman.us/main/2004/09/implementations.html` (2011-1-6)

"There's also an advanced technique involving etags which keeps track of when each etag was issued and if an old etag is sent with the request then everything since then is sent back in the response. Someone will chime in with the reference." Eric Scheid, `http://www.imc.org/atom-syntax/mail-archive/msg19832.html` (2012-1-6)

Microsoft specification FeedSync (formerly Simple Sharing Extension) `http://feedsyncsamples.codeplex.com`, comment `http://notes.kateva.org/2008/01/microsoft-feedsync-what-heck-is-html` `http://web.archive.org/web/20081114142152/http://www.snellspace.com/wp/?p=818`

9. Mediatypes

> To some extent, people get REST wrong because I failed to include enough detail on media type design within my dissertation. – Roy T. Fielding

from: Rest APIs must be hypertext driven[12]

9.1. Syntax vs. Semantic

The usage of standardized media types is one key difference between an API and a restful API. [Fie00, sec. 5.2.1.2] Only if the client has knowledge about the media type can it do something meaningful with it besides just receiving it. In that sense, the often used mime types application/xml or application/json are not really media types. They don't tell the browser or user anything meaningful beside the *syntax* of the data.[13]

---

[12] `http://roy.gbiv.com/untangled/2008/rest-apis-must-be-hypertext-driven` (2011-12-20)

[13] `http://blog.programmableweb.com/2011/11/18/rest-api-design-putting-the-type-in-content-type` (2011-21-20) and Web Resource Modeling Language `http://www.wrml.org` (2011-12-20) both by Mark Massé

```
<lang pref="1"          "languages": [                        "languages": {
     id="fr" />             {"id":"fr","pref":"1"},            "fr":1,
<lang pref="3"             {"id":"en","pref":"3"}             "en":3
     id="en" />         ]                                 }
```

| (a) XML fragment | (b) unfriendly JSON | (c) less(?) unfriendly JSON |

Figure 1: different ways to map XML to JSON

To do anything meaningful with plain json or xml, the client programmer must normally look up the meaning or *semantic* of the data in the API documentation. The data therefor fails the self-descriptive constraint of REST.

Compare this with a mime type like `application/atom+xml`. It specifies the syntax (xml) and the semantic (atom) of the data. Of course somebody once needed to read the atom specification and program the client with the knowledge of how to process this media type. The purpose of standardized media types however is that there is a limited number of them and that they are reused by many sites and clients.

Large sites like Google, Facebook or Twitter can successfully attract enough developers to read their specifications and program clients accordingly. Usually one can also find client libraries that can help a lot. REST however envisions a decentralized web in which parties can interact without previous knowledge of each other. This becomes possible through the usage of well known predefined media types.

9.2. xml vs. json

The application section of the IANA mime type registration has 294 entries ending in "+xml" and only 3 ending in "+json".[14] This stands in contrast to the rise of public json APIs and the decline of XML APIs.[15]

JSON is often preferred as a format over XML because it is perceived as easier to parse, process and smaller in size. XML in comparison is seen as complicate, slow to process and larger in size. A strong argument for JSON as the preferred format for JavaScript applications is that JSON is a subset of JavaScript.

A drawback of this mismatch between the preference of media type designers and API consumers is a possible duplication of work and incompatibilities across different APIs. If two developers independently take the same XML media type as a model to develop a JSON format, they can still come up with different results. There is no standard mapping from XML schemes to JSON structures.

Instead, possible mappings have to trade of the preservation of all structural information against the "friendliness" of the resulting JSON structure.[BGM+11] Without going into detail, a JSON structure can be seen as friendly if it makes best use of JSON's data types, is compact and easy to process. Figure 1 shows two different examples how to map data from XML to JSON with one of them using JSON number values, being more compact and probably easier to process.

in the area of semantic standardization. Questions whether and how contact information should also hold space for the place of birth and place and date of death of a person[16] are independent from a data serialization format. So is the question whether and how to specify

---

[14] http://www.iana.org/assignments/media-types/application/index.html (2011-12-20)

[15] http://blog.programmableweb.com/2011/05/25/1-in-5-apis-say-bye-xml/ (2011-12-20) http://www.readwriteweb.com/cloud/2011/03/programmable-web-apis-popping.php (2011-12-21)

[16] https://datatracker.ietf.org/doc/draft-ietf-vcarddav-birth-death-extensions/ (2011-12-20)

type of data	XML	JSON	semantic	microformat
Calendar	xCal	Google calendar API	http://www.w3.org/TR/rdfcal	hCalendar
Contact	xCard	portable contacts, jCard	friend of a friend	hCard
Resume	HR XML		Description of a Career	hResume

Table 4: data in different formats

the sex or gender of a person.[17]

Version 3 of vCard was published in 1998[HSD98] only a few months after the W3C published Version 1.0 of XML[PSMB98] and eight years before JSON became an official standard.[Cro06]

XML standards: xCal, xCard, XMPP, Atom

http://en.wikipedia.org/wiki/ECMAScript_for_XML (E4X) makes XML a first class language construct

Open-Xchange provides an (unrestful) HTTP/JSON API which is used by its javascript frontend.[18] The comprehensive documentation does not indicate whether the data structures for tasks, appointments, reminders and contacts were inspired by any standards. In any case the API documentation is a good example of the need for standard mime types in JSON format. The use case for this API also shows similarities to the use of OpenSocial for intranet frontends.

9.3. Updates with alternative Mediatypes

How to handle updates, if the mediatypes are not isomorph?

How does Google handle PATCH in the calendar API?

9.4. Mediatype conversion

JSON formats for collections: Collection+JSON Mime-Type (approved in July 2011) by Mike Amundsen[19] JSON ATOM serialization implemented by Apache Abdera[20]

Some problems in loss-less conversion of ATOM to json:[Sne08]

- JSON has no equivalent for the xml:lang attribute.

- Dereferencable IRIs must be transformed to URIs.

- URIs relative to an xml:base attribute must be resolved, also inside XHTML content elements.

- Repeatable elements must be converted to arrays.

- The ATOM date format (RFC 3339) differs from the JavaScript Date serialization.

- ATOM content elements are versatile but should be represented more meaningful in JSON then just a plain String.

- ATOM supports arbitrary extensions via namespaces.

---

[17]http://www.ietf.org/mail-archive/web/vcarddav/current/msg01778.html (2011-12-20)

[18]http://oxpedia.org/index.php?title=HTTP_API (2011-19-12)

[19]http://amundsen.com/media-types/collection/ (2012-1-7)

[20]http://www.ibm.com/developerworks/library/x-atom2json/index.html (2012-1-7) https://cwiki.apache.org/ABDERA/json-serialization.html (2012-1-7)

RFC 5023 "Collections are represented as Atom Feeds"

Is conneg (content negotiation) useful? No: Norman Walsh, 2003, it can lead to hard to debug bugs http://norman.walsh.name/2003/07/02/conneg (2011-1-9), Joe Gregorio, 2003, I can't communicate the mime type to request to a third service if I can only give an URI http://bitworking.org/news/WebServicesAndContentNegotiation (2011-1-9) Yes: Jerome Louvel, 2006, I could additionally provide URIs that override the accept headers with query parameters like ?format=json. http://blog.noelios.com/2006/11/15/reconsidering-content (2011-1-9)

Which fields of portable contacts are derived from vCard: http://wiki.portablecontacts.net/w/page/17776141/schema

> No single data representation is ideal for every client. This protocol defines representations for each resource in three widely supported formats, JSON [RFC4627], XML, and Atom [RFC4287] / AtomPub [RFC5023], using a set of generic mapping rules. The mapping rules allow a server to write to a single interface rather than implementing the protocol three times.

[Ope11, Core API Server]

In 2007, a project called microjson wanted to standardize json representations of microformat data structures.[21]

The project identified the need for a json schema:[22]

> If there are standard microJSON formats for transfer of certain datasets, there will be a need to validate that data to ensure that it is infact valid format. To validate a format you need something that details the structure, data content types and required data. Sounds like we'll be needing a schema for each microJSON format.

jCard example from microjson.org[23]

```
{
"vcard":{
  "name":{
    "given":"John",
    "additional":"Paul",
    "family":"Smith"
  },
  "org":"Company Corp",
  "email":"john@companycorp.com",
  "address":{
    "street":"50 Main Street",
    "locality":"Cityville",
    "region":"Stateshire",
    "postalCode":"1234abc",
    "country":"Someplace"
  },
  "tel":"111-222-333",
  "aim":"johnsmith",
  "yim":"smithjohn"
}
```

---

[21] http://notizblog.org/2007/09/16/microjson-microformats-in-json/ (2011-12-19)

[22] http://web.archive.org/web/20080524003749/http://microjson.org/wiki/Schemas (2022-12-19)

[23] http://web.archive.org/web/20080517003233/http://microjson.org/wiki/JCard (2011-12-19)

9.5. Example: vCard

```xml
<?xml version="1.0" encoding="UTF-8"?>
<vcards xmlns="urn:ietf:params:xml:ns:vcard-4.0">
  <vcard>
    <fn><text>Simon Perreault</text></fn>
    <n>
      <surname>Perreault</surname>
      <given>Simon</given>
      <additional/>
      <prefix/>
      <suffix>ing. jr</suffix>
      <suffix>M.Sc.</suffix>
    </n>
    <bday><date>--0203</date></bday>
    <anniversary>
      <date-time>20090808T1430-0500</date-time>
    </anniversary>
    <gender><sex>M</sex></gender>
    <lang>
      <parameters><pref><integer>1</integer></pref></parameters>
      <language-tag>fr</language-tag>
    </lang>
    <lang>
      <parameters><pref><integer>2</integer></pref></parameters>
      <language-tag>en</language-tag>
    </lang>
    <org>
      <parameters><type><text>work</text></type></parameters>
      <text>Viagenie</text>
    </org>
    <adr>
      <parameters>
        <type><text>work</text></type>
        <label><text>Simon Perreault
2875 boul. Laurier, suite D2-630
Quebec, QC, Canada
G1V 2M2</text></label>
      </parameters>
      <pobox/>
      <ext/>
      <street>2875 boul. Laurier, suite D2-630</street>
      <locality>Quebec</locality>
      <region>QC</region>
      <code>G1V 2M2</code>
      <country>Canada</country>
    </adr>
    <tel>
      <parameters>
        <type>
          <text>work</text>
```

```xml
            <text>voice</text>
          </type>
        </parameters>
        <uri>tel:+1-418-656-9254;ext=102</uri>
      </tel>
      <tel>
        <parameters>
          <type>
            <text>work</text>
            <text>text</text>
            <text>voice</text>
            <text>cell</text>
            <text>video</text>
          </type>
        </parameters>
        <uri>tel:+1-418-262-6501</uri>
      </tel>
      <email>
        <parameters><type><text>work</text></type></parameters>
        <text>simon.perreault@viagenie.ca</text>
      </email>
      <geo>
        <parameters><type><text>work</text></type></parameters>
        <uri>geo:46.766336,-71.28955</uri>
      </geo>
      <key>
        <parameters><type><text>work</text></type></parameters>
        <uri>http://www.viagenie.ca/simon.perreault/simon.asc</uri>
      </key>
      <tz><text>America/Montreal</text></tz>
      <url>
        <parameters><type><text>home</text></type></parameters>
        <uri>http://nomis80.org</uri>
      </url>
    </vcard>
</vcards>

<?xml version="1.0" encoding="UTF-8"?>
<vcards xmlns="urn:ietf:params:xml:ns:vcard-4.0">
  <vcard>
    <fn><text>Simon Perreault</text></fn>
    <n>
      <surname>Perreault</surname>
      <given>Simon</given>
      <suffix>ing. jr</suffix>
      <suffix>M.Sc.</suffix>
    </n>
    <bday day="02" month="03" />
    <anniversary format="date-time">20090808T1430-0500</anniversary>
    <gender>M</gender>
    <lang pref="1">fr</lang>
    <lang pref="2">en</lang>
```

```
      <org type="work">Viagenie</org>
      <adr type="work">
        <label>Simon Perreault
2875 boul. Laurier, suite D2−630
Quebec, QC, Canada
G1V 2M2</label>
        <street>2875 boul. Laurier, suite D2−630</street>
        <locality>Quebec</locality>
        <region>QC</region>
        <code>G1V 2M2</code>
        <country>Canada</country>
      </adr>
      <tel>
        <type>work</type>
        <type>voice</type>
        <uri>tel:+1−418−656−9254;ext=102</uri>
      </tel>
      <tel>
        <type>work</type>
        <type>text</type>
        <type>voice</type>
        <type>cell</type>
        <type>video</type>
        <uri>tel:+1−418−262−6501</uri>
      </tel>
      <email type="work">simon.perreault@viagenie.ca</email>
      <geo type="work">
        <uri>geo:46.766336,−71.28955</uri>
      </geo>
      <key type="work">
        <uri>http://www.viagenie.ca/simon.perreault/simon.asc</uri>
      </key>
      <tz>America/Montreal</tz>
      <url type="home">
        <uri>http://nomis80.org</uri>
      </url>
    </vcard>
  </vcards>
```

9.6. HFactor

Mike Amundsen defines a method to asses media types that he calls "HFactor".[24] The
HFactor distinguishes different types of support for links and indicates which of those are
provided by a reviewed media type.

Amundsen did reviews of a couple of media types. Unfortunately these do not include
`vcard+xml` or `calendar+xml`. I'll try to identify the HFactors of both here.

The different types of link support have two letter acronyms and fall in two categories:
Link support values, with the first letter "L" and Control data support, first letter "C".

- Link Support for
  - `LE` embedded links (HTTP GET)

---

[24]`http://amundsen.com/hypermedia/` (2011-12-21)

- LO out-bound navigational links (HTTP GET)
- LT templated queries (HTTP GET)
- LN non-idempotent updates (HTTP POST)
- LI idempotent updates (HTTP PUT, DELETE)

- Control Data Support to
  - CR modify control data for read requests (e.g. `HTTP Accept-*` headers)
  - CU modify control data for update requests (e. g. `Content-*` headers)
  - CM indicate the interface method for requests (e.g. HTTP GET,POST,PUT,DELETE methods)
  - CL add semantic meaning to link elements using link relations (e.g. HTML rel attribute)

10. Hypermedia in RESTful applications

The model application is therefore an engine that moves from one state to the next by examining and choosing from among the alternative state transitions in the current set of representations.

[Fie00, sec. 5.3, p.103]

10.1. Hypermedia in OpenSocial

Webfinger, e.g. get a profile picture from an email address
Danger: One can trigger na http request by sending an email.

11. Selection of components

Apache Shindig for Open Social, includes client tests
http://code.google.com/p/kolab-android/
https://evolvis.org/projects/kolab-ws/
http://packages.ubuntu.com/source/maverick/dovecot-metadata-plugin https://launchpad.net/ubu
metadata-plugin/8-0ubuntu1

11.1. REST framework

Jersey recommended by [Kai11] above Restfulie and RESTeasy because of maturity and flexibility.
Jersey has a atompub-contact client/server example app.
Why not Jersey in the end?

- JAX-RS assumes, that Paths are defined on the classes that represent the resources.
  - This couples the "location" of a resource to its implementation.
  - This leads to copied code. Given an URL pattern like `/{AUTHORITY}/{COLLECTION}/{ENTRY}` In this case the resource classes for authority, collection and entry would each need to parse the authority section of the path.
  - If paths are not defined on resource classes, it is not possible to make use of JAX-RS' capabilities of declarative hyperlink building (@REF annotation).

- The dispatch to a request handler method has in our case three orthogonal parameters: HTTP verb, Media type, path. It would be preferable to handle these parameters independent of each other. The only way to handle at least the path dispatch separately is with the help of sub resources. This still leaves HTTP verb and Media type to be handled together.

  The sub resource mechanism additionally suffers from the shortcoming that it does not allow to specify an empty path.[25] This makes it impossible to return a sub resource and annotate a method that should handle the case that no additional path elements remain to be matched.

- Debugging is hard. It's not trivial to find out, why Jersey did not select a request handler or provider as the developer intended.

- Jersey's parameter injection can not be used together with a dependency injection framework like Guice or Spring.

11.2. VCard

ical4j best documented best code is used by most active also supports icalendar is immutable!!!

12. Testing

How to test the ReST/CardDAV interface?

Portable Contacts test client at plaxo `http://www.plaxo.com/pdata/testClient` `http://code.google.com/p/rest-assured/` `http://restfuse.com/`

A. Standards

A.1. Contacts / Persons

RFC 6450 vCard Format Specification
This document defines the vCard data format for representing and exchanging a variety of information about individuals and other entities (e.g., formatted and structured name and delivery addresses, email address, multiple telephone numbers, photograph, logo, audio clips, etc.). This is the new version and obsoletes RFCs 2425, 2426, and 4770, and updates RFC 2739.

RFC 6351 xCard: vCard XML Representation
This document defines the XML schema of the vCard data format.

Portable Contacts, OpenSocial
Portable Contacts defines contact data structures and a ReST API. It has been integrated in the OpenSocial standard.

Nepomuk Semantic Desktop Contact Ontology

Friend of a friend (FOAF)
FOAF is a

---

[25] `http://java.net/jira/browse/JERSEY-536` (2012-01-21)

hCard

jCard

A.2. Calendaring

RFC 5545 Internet Calendaring and Scheduling Core Object Specification

iCalendar is the core data schema for calendaring information. This is the new version and obsoletes RFC2445

RFC 6321 xCal: The XML format for iCalendar

This specification defines a format for representing iCalendar data in XML. More specifically, is to define an XML format that allows iCalendar data to be converted to XML, and then back to iCalendar, without losing any semantic meaning in the data. Anyone creating XML calendar data according to this specification will know that their data can be converted to a valid iCalendar representation as well.

CalWS RESTful Web Services Protocol for Calendaring

This document, developed by the XML Technical Committee, specifies a RESTful web services Protocol for calendaring operations. This protocol has been contributed to OASIS WS-CALENDAR as a component of the WS-CALENDAR Specification under development by OASIS.

Google Calendar API V3

While not being a standard, the Google Calendar API is RESTful and will surely be implemented by many client applications. It's remarkable that the API supports partial GETs returning only specified fields and the HTTP PATCH verb to update only specified fields.

Open Services for Lifecycle Collaboration (OSLC)

uses FOAF person `http://open-services.net/bin/view/Main/OSLCCoreSpecAppendixA?sortcol=table;up=#foaf_Person_Resource`

provides change management, some overlapping to iCal TODOs `http://open-services.net/bin/view/Main/CmSpecificationV2`

reference implementation: `http://eclipse.org/lyo`

A.3. Scheduling

RFC 5546 iCalendar Transport-Independent Interoperability Protocol (iTIP)

Scheduling Events, BusyTime, To-dos and Journal Entries; Specifies the mechanisms for calendaring event interchange between calendar servers. This is the new version and obsoletes RFC2446

RFC 6047 iCalendar Message-Based Interoperability Protocol (iMIP)

Specifies how to exchange calendaring data via e-mail. This is the new version and obsoletes RFC2447.

A.4. Relations and Links

Xhtml Friends Network (XFN)

One of the relations returned by Google's webfinger.

Webfinger
Webfinger in Firefox Contacts Add-On `http://mozillalabs.com/blog/2010/03/contacts-in-the-`

RFC 6415 Web Host Metadata

Extensible Resource Descriptor (XRD)

A.5. out of scope

HR XML

The HR-XML Consortium is the only independent, non-profit, volunteer-led organization dedicated to the development and promotion of a standard suite of XML specifications to enable e-business and the automation of human resources-related data exchanges.

OMA Converged Address Book V1.0

Standard by the Open Mobile Alliance defining data structures and synchronization of contact data. It references vCard.

Open Collaboration Services

Also contains data structures for persons and events but does not reuse any known standard. See this thread: `http://lists.freedesktop.org/archives/ocs/2011-December/000136.html`

W3C Contacts API

A standard on how address books cold be accessed on devices or from JavaScript inside a Web Browser. The standard references vCard, OMA Converged Address Book and Portable Contacts.

W3C vCard ontology

W3C PIM ontology

B. People, Groups and Organizations

People

Eran Hammer-Lahav
`http://hueniverse.com` Yahoo!, OAuth

Eliot Lear ¡lear@cisco.com¿
 IETF Calsify WG chair

James Snell
`http://chmod777self.blogspot.com/`
 Abdera committer

Julian Reschke ¡julian.reschke@gmx.de¿

Lisa Dusseault

Lisa Dusseault is a development manager and standards architect at the Open Source Applications Foundation, where she's involved in the Chandler, Cosmo and Scooby projects. Previously, Lisa came from Xythos, an Internet startup where she was development manager for four years. She has also been an IETF contributor on various Internet applications protocols for eight years now, and continues to do this kind of work at OSAF. She co-chairs the IETF IMAP extensions and CALSIFY (Calendaring and Scheduling Standards Simplification) Working Groups. She is also the author of a book on WebDAV and co-author of CalDAV, an open and interoperable protocol for calendar access and sharing.

Mark Nottingham

Mike Amundsen ¡mamund@yahoo.com¿
`http://amundsen.com`

Mike Conley

`http://mikeconley.ca/blog/` working on a new address book for Thunderbird: `https://wiki.mozilla.org/Thunderbird/tb-planning`

Peter Saint-Andre ¡stpeter@stpeter.im¿

 IETF Calsify WG area director

Joseph Smarr

former Plaxo now Google presentation about portable contacts at vcarddav wg http://tools.ietf.org/age 2.pdf http://josephsmarr.com http://anyasq.com/79-im-a-technical-lead-on-the-google+-team

C. Implementations

C.1. Servers

Cyn.in
 Python, Open Core

DAViCal

 PHP, SQL storage, CalDAV, CardDav

eGroupWare

eXo Platform
 Open Core, Java, AGPL, participates in OpenSocial?

Group-Office
 PHP, AGPL

Horde

OBM Groupware
 PHP, GPL

Open-Xchange
 Java, In 2006 a Debian packaging attempt was canceled because upstream decided not to publish security updates for the open source version anymore.[26]

owncloud

 ownCloud supports syncing of calendar and contacts information via the CalDAV and CardDAV protocols.

Scalix
 Open Core Scalix Public License (SPL) based on MPL, requires to show the Scalix Logo

Simple Groupware
 PHP, GPL, SQL

SOGo
 CalDAV and CardDAV, written in Objective-C

Tiki Wiki
 PHP, SQL Contacts `http://doc.tiki.org/Contacts`, Calendar `http://doc.tiki.org/Calendar` iCal export apparently no CardDAV/CalDAV many many features!

Tine 2.0
 Tine is not eGroupWare

Zarafa
 PHP, MySQL IIRC it uses an Entity-Attribute-Value pattern to store its data in the relational db.

---

[26]`http://web.archive.org/web/20100510133805/http://seraphyn.deveth.org/archives/10-Keine-Zukunft-in-der-freien-Version-von-Open-Exchange-auf-Debian.html` (2011-12-19)

Zimbra
Open Core, Own license (Zimbra Public License), RFP since 2008 open: http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=498316

C.2. Clients

Spicebird
built on top of Thunderbird with Calendar

Thunderbird

CardDAV via SoCO connector `http://www.sogo.nu/fr/downloads/frontends.html`

WebiCal
Java, YUI, Web frontend for a CalDAV server, uses iCal4J

Evolution, Evolution Data Server

KDE Kontact, Akonadi

more CardDAV
`http://wiki.davical.org/w/CardDAV/Clients` `http://en.wikipedia.org/wiki/CardDAV# Implementations`

more CalDAV
`http://wiki.davical.org/w/CalDAV_Clients` `http://en.wikipedia.org/wiki/CalDAV# Implementations`

C.3. Web Services

C.4. Others

D. Links

- `http://thesocialweb.tv`

- `http://www.vogella.de/articles/REST/article.html` REST with Java (JAX-RS) using Jersey - Tutorial

- `https://addons.mozilla.org/de/firefox/addon/restclient/`

- `http://dataportability.org/` still active?

- `http://tech.groups.yahoo.com/group/rest-discuss/messages/17242?threaded= 1&m=e&var=1&tidx=1` REST and Semantic

- `http://stackoverflow.com/questions/2669926/practical-advice-on-using-jersey-and-`

IANA link relations registry `http://www.iana.org/assignments/link-relations/link-relatic`
`xml`
ATOM landscape overview `http://dret.typepad.com/dretblog/atom-landscape.html`

D.1. Apache Shindig

RPC vs. REST API for Shindig/OpenSocial: `http://groups.google.com/group/opensocial-and-ga browse_thread/thread/a4ddf7cd09f90237/5cfa1658e1c1d698?lnk=gst&q=rest#5cfa1658e1c1d698` `http://groups.google.com/group/opensocial-and-gadgets-spec/browse_thread/thread/ d1a5627fb6e686ce/d27d47dee92a87b2` One argument was support for batching. A restful batching proposal didn't get consensus: `https://docs.google.com/View?docid=dc43mmng_ 23fdbpp7hd&pli=1`

Flow of REST requests in Shindig `https://sites.google.com/site/opensocialarticles/ Home/shindig-rest-java`

Google+ is likely to become OpenSocial enabled: `http://groups.google.com/group/ opensocial-and-gadgets-spec/browse_thread/thread/1187241df6759a9a`

Shindig issues to implement OpenSocial 2.0 `https://docs.google.com/spreadsheet/ ccc?key=0AihdZBncP3KzdGN3dVl3MFpIUlk2TXIyR3hfUDhHZUE&hl=en_US#gid=0`

How Shindig supports extensions: `https://cwiki.apache.org/confluence/display/ SHINDIG/Arbitrary+Extensions+to+Apache+Shindig%27s+Data+Model`

Videos about some 2.0 OS features `http://groups.google.com/group/opensocial-and-gadgets- browse_thread/thread/7b911edfb1bb3b4d`

OS and RDF `http://groups.google.com/group/opensocial-and-gadgets-spec/browse_ thread/thread/20f62d627003509b`

OpenSocial Development Environment (OSDE, Eclipse Plugin) `https://sites.google. com/site/opensocialdevenv`

D.2. Socialsite

Oracle's (former Sun's) extension to Apache Shindig. Blog `http://blogs.oracle.com/ socialsite`

E. TODO

- Does funambol.org has interesting implementations?

support Plain Text Format (text/plain), RFC5147 URI fragment identifier for plain text?

> Sowohl Atom als auch AtomPub definieren XML-Vokabulare, die eine Erweiterung mit zwei Mechanismen unterstützen. Zum einen ist im Standard definiert, dass neue Elemente in diesen Vokaularen selbst von standardkonformen Prozessoren ignoeriert werden müssen. [...] Gleichzeitig ist es überall dort, wo es nicht explizit verboten ist, möglich, Elemente ausanderen XML-Namespaces einzubetten.

[Til11, p. 102]

References

[Amu10]   AMUNDSEN, Mike: *Fielding Property Maps.* `http://amundsen.com/examples/ fielding-props/`, 3 2010

[BGM+11] BOYER, John ; GAO, Sandy ; MALAIKA, Susan ; MAXIMILIEN, Michael ; SALZ, Rich ; SIMEON, Jerome: Experiences with JSON and XML Transformations. In: *Workshop on Data and Services Integration* W3C, 2011

[BLFM05] BERNERS-LEE, T. ; FIELDING, R. ; MASINTER, L.: Uniform Resource Identifier (URI): Generic Syntax / RFC Editor. RFC Editor, January 2005 (3986). – RFC

[Cro06] CROCKFORD, D.: The application/json Media Type for JavaScript Object Notation (JSON) / RFC Editor. RFC Editor, July 2006 (4627). – RFC

[Dab11] DABOO, C.: CardDAV: vCard Extensions to Web Distributed Authoring and Versioning (WebDAV) / RFC Editor. RFC Editor, August 2011 (6352). – RFC

[Fie00] FIELDING, Roy T.: *REST: Architectural Styles and the Design of Network-based Software Architectures*, University of California, Irvine, Doctoral dissertation, 2000

[Gh07] GREGORIO, J. ; HORA, B. de: The Atom Publishing Protocol / RFC Editor. RFC Editor, October 2007 (5023). – RFC

[HSD98] HOWES, T. ; SMITH, M. ; DAWSON, F.: A MIME Content-Type for Directory Information / RFC Editor. RFC Editor, September 1998 (2425). – RFC

[Kai11] KAISER, Guido: *Modellierung und Vergleich von Implementierungen einer REST-Anwendung in verschiedenen Sprachen*, Fernuniversität Hagen, Master thesis, 1 2011. http://www.fernuni-hagen.de/dvt/studium/masterarbeiten_9.shtml

[NS05] NOTTINGHAM, M. ; SAYRE, R.: The Atom Syndication Format / RFC Editor. RFC Editor, December 2005 (4287). – RFC

[Ope11] OPENSOCIAL AND GADGETS SPECIFICATION GROUP: *OpenSocial Specification Version 2.0.1*. http://docs.opensocial.org/display/OSD/Specs. Version: 11 2011

[PSMB98] PAOLI, Jean ; SPERBERG-MCQUEEN, C. M. ; BRAY, Tim: XML 1.0 Recommendation / W3C. 1998. – first Edition of a Recommendation. – http://www.w3.org/TR/1998/REC-xml-19980210

[Sne08] SNELL, James M.: *Convert Atom documents to JSON*. http://www.ibm.com/developerworks/library/x-atom2json/index.html. Version: January 2008

[Til11] TILKOV, Stefan: *REST und HTTP: Einsatz der Architektur des Web für Integrationsszenarien*. 2. DPUNKT VERLAG, 2011

[Yat07] YATES, Rob: *CalAtom*. http://robubu.com/CalAtom/calatom-draft-00.txt. http://robubu.com/CalAtom/calatom-draft-00.txt. Version: April 2007