Format Specification of POIs (Points of Interests)

in OGC speak these are “Places of Interest”

Summary

Authors: Stefan Lemme (DFKI)
Status: Draft
Version: 0.4.1

Involved Parties

Interested parties of FIcontent WP3 (local information), WP4 (city-wide gaming), WP5 (standardization), WP6 (architecture decisions):

- Orange (WP3)
- FhG FOKUS (WP3)
- Disney (WP4)
- Black Rock (WP4)
- DFKI (WP3, WP4, WP6)
- pixelpark (WP3, WP4)
- …

External interested parties are

- WeX consortium (FI-WARE)
- ...

Introduction

From the charter of the OGC’s POI standard working group there is a very good introduction about POIs:

> *A “place of interest” (POI) is a location about which information is available. A POI can be as simple as a set of coordinates, a name, and a unique identifier, or more complex such as a three-dimensional model of a building with names in multiple languages, information about opening and closing hours, and a civic address. POI data has many uses including augmented reality browsers, location-based social networking games, geocaching, mapping and navigation systems.*
> *In practice, POIs are usually those places that serve a public function. As such, POIs generally exclude facilities such as private residences, industrial buildings and offices, but include many private facilities that seek to attract the general public such as retail businesses, amusement parks, etc. Most government buildings and important natural features are POIs as well.*

In FIcontent we try to match the requirements for the two use case WPs (Smart City Guide and Gaming) with our proposed format specification. Hence, we define a POI as follows:

> *POIs are some “Places of Interest” which are relevant for a user according to his preferences and profile. A POI is a unique identifiable entity in a spatio-temporal reference system. It has - probably multilingual - content attached to provide information about this “Place of Interest” to the user.*

This is a very generic definition of a POI and we try to achieve with our proposed format a huge flexibility to maintain such a generic definition. In the remaining sections of this document we collected the intended applications of POIs and the respective requirements from partners. Existing formats are briefly reviewed followed by our proposed data model. The appendix contains some more detailed examples of existing formats to ease the comparison.

Intended Applications

- Storage of POIs
- Querying based on POI attributes and related data
- Transferring POIs between applications
- Visualizing POIs in 2D maps and 3D environments
- Augmenting the real world with POI data

- Editing / Append POI content
- Creating POIs
- Annotating the real world with POI data

Requirements

This is a collection of all the input requirements received from interested parties. The proposed format should fulfill at least most and at best all of the mentioned requirements. They will be later prioritized to measure their importance regarding the fulfillment by the proposed format.

1. [Orange] Handle permanent geographic places as POIs (corresponds to a physical location, not time-dependent)
2. [Orange] Handle temporal events as POIs (time dependent, can have a physical location, can be linked to another POI)
3. [Orange] Handle crowds of people like your friends or persons that share common interests with you as POIs
4. [DFKI] Handle virtual, fictive objects placed in the real world as POIs
5. [DFKI] Handle movable objects as POIs, like trams, humans, virtual/hybrid world avatars
6. [Orange] Handle zones (e.g. a park) as POIs
7. [DFKI] Handle street, tracks, and paths as POIs
8. [Orange] POIs provide a name
9. [DFKI] POIs provide a label (to be shown on a map)

10. [Orange] POIs provide a geo-referenced location
11. [WeX] POIs provide some kind of a location description
    a. [WeX] longitude / latitude coordinates
    b. [WeX] somewhere in space (e.g. moon)
    c. [WeX] relative to another POI
    d. [WeX] using "soft coordinates", like "Mona Lisa" is located in "third room in the main hall inside Louvre POI"
12. [Orange] POIs provide a detailed description
13. [Orange] POIs provide a category
14. [DFKI] POIs provide tags
15. [WeX] POI related content should be freely customizable
    a. [WeX] Opening hours
    b. [WeX] Business Information
    c. [WeX] Sensor data using COAP URLs
    d. [WeX] 3D Model representing the POI, e.g. the building
    e. [WeX] Virtual counterpart, e.g. vending machines may have control interfaces in the (3D) Internet
16. [Orange] POIs provide information about their data source
17. [DFKI] POIs provide information about the author, license, and copyright
18. [Orange] POIs provide information about their relation to other POIs
19. [WeX] POIs can have a hierarchy – POIs inside POIs, e.g. museums with several exhibits inside
20. [DFKI] POIs have associated user generated content
21. [WeX] Content of POIs can be crowd-sourced – e.g. add new images
22. [Orange] POIs can have rating and comments from users attached
23. [WeX] Differentiate between actual POI location and recorded data location, e.g. for pictures taken hundreds of meters away from the POI but relate to them
24. [Orange] A collection of POIs can be selected with regard to their
    a. [Orange] location (near a point = distance, in a zone/area)
    b. [Orange] category
    c. [Orange] time (for events)
    d. [DFKI] tags
    e. [DFKI] similarity to another POI (based on tags, description, name, relations)
25. [Orange] Sort / filter by any of the fields available
26. [Orange] Manage personal lists of POIs
27. [WeX] POIs should be distinguishable – provide a proper identifier
28. [DFKI] POIs can be stored decentralized – they hold a globally unique identifier
    a. [DFKI] Whole POIs distributed with regard to their topic, e.g. all museums of modern art together in one data base
    b. [DFKI] Whole POIs distributed with regard to their spatial distribution, e.g. a city provides their own data source
    c. [DFKI] Parts of POIs distributed with regard to their content, e.g. get attached videos to a public POI from a special provider

    d. [DFKI] Parts of POIs distributed with regard to the confidentiality of their content, e.g. personal notes attached to a POI shared only with friends
    e. [DFKI] Parts of POIs for special feature support – e.g. an AR initiative disseminates markers to several existing POIs and further provides the feature descriptors to track these markers and use AR at the POI

29. [Orange] Track the name of the creator of a POI
30. [Orange] Allow for approval of a user-created POI before it is made public
31. [DFKI] Most parts of the POI hold information about their data source, author, license, and modification timestamps
32. [Orange] Manage visibility rights (public / private POI)
33. [DFKI] Enable sharing of POIs and POI contents with friends

Existing Formats

There is a variety of formats out there to handle POIs. Most of them are custom solutions to fit exactly one use case. They have common to be not generic and flexible enough to fulfill the mentioned requirements and none of them is well established. Thus, there were some attempts from W3C and OGC to create a standardized format of POIs. Since today there is only a first draft of a W3C POI Core specification which builds the foundation of our approach. In this section we want to bring up several other formats related to POIs beside the preferred W3C specification.

Orange's Format

The format proposed by Orange is just a view of a relational database packed into JSON. It really lacks of structured data and relies almost on hardcoded features, like the 3-tier categorization. Most of the fields are strings that make a respective filtering difficult. The relationship between different POIs is missed completely such as adjacent POIs or similar POIs. Search query related information (e.g. distance) is embedded directly into the data structure.
http://goo.gl/V56I9y [SharePoint; WP3, T3.1]

FOKUS' Format

The format used by FOKUS utilizes JSON to transfer POI data, too. The data structure and the content seems to be close to Google Places. Unfortunately, almost all information are encoded as character strings. Hence, it cumbers the querying, filtering and sorting of POIs. For instance, the nearby public transport stations are POIs as well, but not handled as such.
http://mashweb.fokus.fraunhofer.de:3008/api/pois?_id=4f4f4c27d4374e8001000008

W3C POI Core

The result of the trail of the W3C is quite a good starting point for a format specification of POIs. They described a data model in an early but also mature state of which data should be assigned to a POI. Furthermore, they illustrate the data model with the mapping onto xml data types. We took a lot of their concepts into our approach. This format is used with a mapping to

JSON by the openPOI initiative of the OGC and it natively supports multi-lingual content.
http://www.w3.org/2010/POI/documents/Core/core-20111216.html

Google Places

The Google Places API use JSON as well. The transferred information is indeed very limited, since it is designed for Google Maps as use case. But they apply some interesting schemes to generalize information that differs across countries, such as postal addresses. We adopted this fashion for instance as it surpasses all other used methods.
https://developers.google.com/places/documentation/details

schema.org

On the website schema.org big players from the domain of commerce provide suggestions about how to bring up information mostly related to business at best. This should support search engines to extract the relevant information from web resources. Some of their basic data structure we adopted in our approach, like the specification of opening hours, dates, and times.
http://www.schema.org/docs/full.html

OpenStreetMap

The OpenStreetMap project use a XML data structure consisting of ways, nodes and tags. It is too generic and less relevant for our purposes. But there are attempts to extract POIs from OpenStreetMap data, which could be interesting to gather more information from an additional provider.
http://www.openstreetmap.org/api/0.6/way/19046101

Geonames.org

In the geonames.org database are geo-located places, which are extracted from Wikipedia. The particular focus is on the data mining method of a reliable extraction. They support multiple formats to export the extracted information about such a place. These are almost always only basic information, such as a name with several aliases, a wgs84 position, a categorization based on their ontology, and the classification of the administrative level.
http://sws.geonames.org/6325497/about.rdf

Other File Exchange Formats

We did not have a detailed look to all of them. But they have common that they mainly lack of the possibility to attach generic content and only rely on a small selection of geographic reference coordinate systems which is almost always WGS84. This excludes all AR capabilities or some kind of probable indoor navigation.

- GPX
- KML, KMZ
- GML, CityGML
- Garmin Mapsource (.gdb)
- Pocket Street Pushpins (.psp)

- Maptech Marks (.msf)
- Maptech Waypoint (.mxf)
- Microsoft MapPoint Pushpin (.csv)
- OziExplorer (.wpt)
- TomTom Overlay (.ov2) and TomTom plain text format (.asc)

Proposed POI Concept

Initially we took essential ideas from the W3C POI Core Editor's Draft. We mapped this to a generic approach for our purposes and extended it to match the described requirements. First of all a POI is a unique entity referenced by a universally unique identifier somewhere on earth or the near atmospheric space. It further consists of several components.

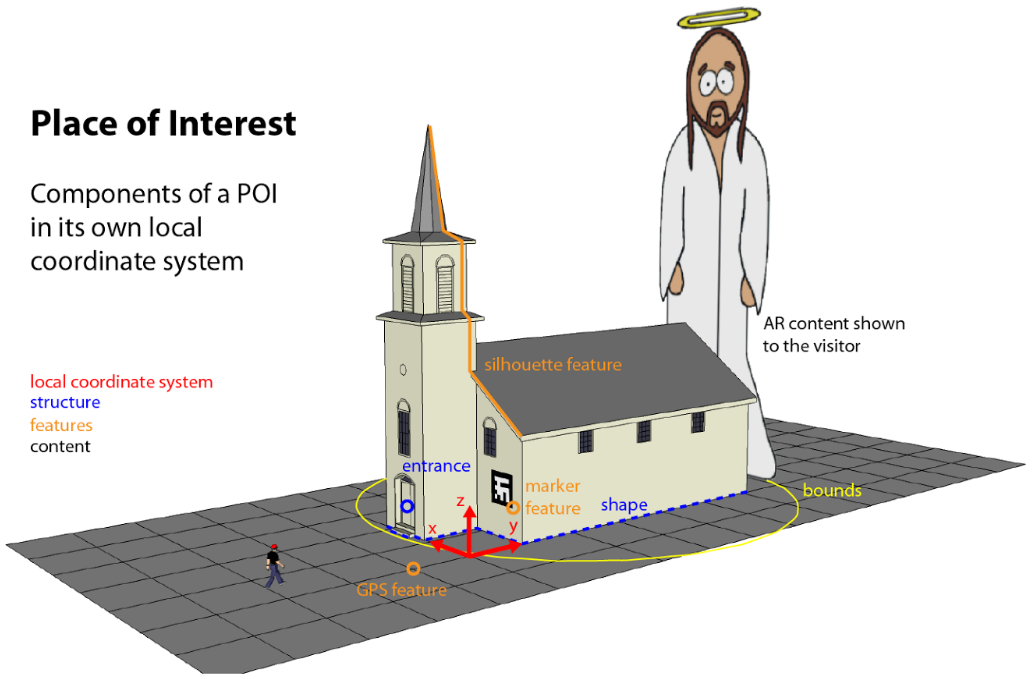
Components of a POI

The particular components of the POI format group some specific type of information or encapsulate some kind of features and associated functionality. These components are

1. the **identifier** of the POI,
2. the representation of the **structure** including the shape and some bounding information,
3. the **features** to find, locate, and map the POI,
4. the actual **content**, like descriptions, images, videos, and 3D model,
5. the **time constraints** for temporary active POIs, like a party at pixelpark or opening hours of a restaurant,
6. the **relations** to other POIs,
7. and some **meta data** consisting of measurements and automatic calculations related to or requested by the search query parameters

Reference Coordinate System

Each POI entity has its own local coordinate system. This coordinate system is defined once per POI and the origin is somewhere adjacent to the POI with its structure and its content. The actual shape of the POI e.g. the outline of a building is given within the POI's local coordinate system. The same applies to content attached to the POI, such as positioned images, texts, 3D-models, video sequences, etc.



We suggest to impose some restrictions to this coordinate system for a more convenient usage and simplified calculations.

- The coordinate system is left-handed with x-axis to the left, y-axis forward and z-axis up.
- The z-axis points in the opposite direction of gravity, i.e. up. Hence, it is collinear to the normal of the reference rotation ellipsoid used to approximate the earth.
- The scale of the basis vectors is normalized. It may correspond to some unit of measurements.

Hence, a POI entity is completely decoupled from any established coordinate reference system such as UTM, WGS84 or ETRS89. This makes it flexible to be used for all of these established reference systems and also additional ones.

Thus, it is possible to fluently change the used coordinate system to locate a POI. This is particular the case when we first locate a POI using our current GPS position and when we get closer switch over to some camera-based tracking method for a more precise positioning.

Features

We utilize so-called *features*, which are attached to a POI entity, to find, locate and map the POI. A feature is dedicated to a specific positioning method, such as GPS, Wifi-based or by address, marker tracking, fast feature tracking, etc. It holds the respective parameters and describes the transformation from the positioning method's coordinate system to the local coordinate system of the POI. In this way, we can use multiple positioning methods simultaneously and receive a very coherent pose of a POI relative to our own position. This is a necessary prerequisite to perform reliable augmented reality applications.

Bounding Volume of 2D and 3D POIs

The usage of the z-component in coordinates along the logical up-axis is optional since some use cases do not require elevation information and some data sources do not provide them. Moreover, the measurement of the altitude is almost always error-prone. The resulting distortion error is quite small due to the adjacent nature of all used coordinates. This simplifies the transformation of existing POI data sets and the handling as well as the computational effort.

For this reason, we decided to use a - somehow unusual - cylindrical bounding volume. Indeed, its advantage is that the radius stays the same for distance calculations in projected 2D and 3D space. In contrast to the radius of a bounding sphere, but the computational effort is almost the same. Furthermore, the bounding cylinder of a zero-height shape structure becomes a bounding circle. Using an axis-aligned bounding box instead would complicate calculations and is more error-prone due to the orientation.

Proposed POI Data Model

In this section we propose the foundation of a data model according to the presented concept of POIs. This data model will be later mapped onto established exchange formats such as XML and JSON. A POI entity is of type POI and consists of the previously described components.
We use several data types and we will specify them in the following sections except basic data types, which are string, float, int, array, boolean, enum, char, bit. The specification of these data types will be done during the mapping of the actual transfer format.
Attributes in bold font are mandatory attributes of the explained type, like id and source for the POI type. Furthermore, there is some kind of inheritance used, which is expressed by
`Subtype { BaseType ( baseattribute = value ) }`.

`Attributes of type POI:`

- **`id / _id : type UUID`**
- `structure : type Structure`
- `features : array of type Feature`
- `contents : array of type Content`
- `times : array of type TimeConstraint`
- **`source : type MainDataSource`**
- `relations : array of type Relation`
- `meta : array of type MetaData`

Identifier

The identifier component of a POI entity is mandatory and contains a universally unique identifier (UUID). We apply the RFC 4122 (Version 4) accordingly. This avoids conflicts when using multiple POI data providers simultaneously.

Structure

The structure component of a POI consists of information about the actual shape, if it is a building or some place or street; it holds information about the bounding volume of this shape, and some specific navigation points with regard to this POI.

`Attributes of type Structure:`

- `shape : type Shape`
- **`bounds : type Bounds`**
- `navigation_points : array of type NavigationPoint`
- `source : type DataSource`

Shape

The shape is given as a representation within the local reference coordinate system of the

POI. This representations is defined by a type, an array of vertices and an optional array of indices. Since the usage of the z-component in the coordinates is optional, a parameter indicates if the vertices include elevation data. As types for shape representation are defined so far:

- **Point** – e.g. for a specific place or small object.
  The array of vertices contains 2 or 3 floats.
- **Path** – e.g. for a hiking track, a promenade or a street.
  The array of vertices contains 2*n or 3*n floats – depending on the elevation data indicator – and they are connected in the provided order to a line string.
- **Polygon** – e.g. for a park, region or building complex.
  The array of vertices contains 2*n or 3*n floats – depending on the elevation data indicator – and they are connected in the provided order to enclose a polygon by adding an edge between the last and the first vertex.

We suggest to omit the usage of elevation data for shapes of type polygon unless you strongly ensure the planarity of the polygon – otherwise it will cumber a probable triangulation. Later extension for types of representations could be triangles to represent complete buildings instead of only the outline, for instance. In that case, this will take advantage of the optional array of indices, which is not yet necessary for the current types.

`Attributes of type Shape:`

- **`type : enum { point, path, polygon }`**
- `elevation : boolean (default = false)`
- **`vertices : array of float`** `(length of 2n or 3n; point: n = 1)`

Bounds

Additionally, the structure component holds information about the bounding volume of this shape. This is either a bounding cylinder or a bounding circle and can be derived from the "real" shape. Alternatively, it can be provided by the POI author and include some height in addition to the flat outline of a building.

`Attributes of type Bounds:`

- **`center : type vec2x`**
- **`radius : float`**
- `height : float (default = 0.0)`

Navigation Points

For the structure of a POI some points or facts might be important to navigate the user, such as the entrance or the location of the nearby parking lot. We attach these immanent points directly to the structure of the POI. If one of these points becomes more interesting it can be separated from this POI into a new one residing in this POI coordinate system. If only the type is provided, it might inform the user about the existence of a dedicated parking.

Attributes of type NavigationPoint:

- **type : enum {mainpoint, entrance, arrival, parking, ...}**
- coordinates : type vec2x
- href : type POIReference

Features

We utilize features to find, locate and map the POI. A feature is dedicated to a specific positioning method, such as GPS, Wifi-based or by address, marker tracking, fast feature tracking, etc. It holds the respective parameters and describes the transformation from the positioning method's coordinate system to the local coordinate system of the POI. Thus, a feature specifies its dedicated positioning method with their parameters, the transformation to the POI's reference coordinate system using a matrix, probably a deviation of the positioning method in this spatial area and maybe the data source of this feature.

Attributes of type Feature:

- **method : enum {**
  wgs84, etrs89, address, gln, nyidmarker, qrcode,
  relative, soft, ...
  }
- transformation : type Transformation (default = identity)
- deviation : type vec2x
- source : type DataSource

Geographical Coordinates

Several geographical coordinate reference systems are utilized in practice. One of the most used is the WGS84 coordinate system – usually known as GPS coordinates. Indeed, there are other coordinate systems used, such ETRS89 in the European area. We permit to use all of them to describe to localization of a POI. The two main geographical features are defined here.

Attributes of type WGS84Feature { Feature(method = wgs84) }:

- **longitude : float**
- **latitude : float**
- elevation : float (default = ground-bound)

The constructed coordinate system in a point defined by WGS84 coordinates has its x-axis and the y-axis as tangents of the reference rotation ellipsoid and thereby, the y-axis pointing north and the x-axis pointing west. The z-axis is collinear to the normal of this ellipsoid.

Attributes of type ETRS89Feature { Feature(method = etrs89) }:

- **`zone: int`**
- **`band: char`**
- **`east : float`**
- **`north : float`**

Civic Address

The postal or civic address is a unique identifier for a location. Hence, we allow to describe to localization of a POI using an address. Therefore, we utilize the fashion of Google Places to specify the components of the address.
https://developers.google.com/places/documentation/details#PlaceDetailsResults

`Attributes of type AddressFeature { Feature( method = address ) }:`

- **`components : array of type Google::AddressComponent`**
- `formatted : string`

For companies exist another method to retrieve their location. The Global Location Number (GLN, sometimes also referred to as International Location Number or ILN) of the respective organization, person, or place. The GLN is a 13-digit number used to identify parties and physical locations.

`Attributes of type GLNFeature { Feature( method = gln ) }:`

- **`gln : string (13-digit number)`**

Marker-based

If the physical POI object has a marker attached, we are able to provide reliable AR applications. This can be very different types of markers, e.g. a NyID-Marker of ARToolkit or a QR-Code.

`Attributes of type NyIDFeature { Feature( method = nyidmarker ) }:`

- **`id : int`**
- `model : enum { 2, 3 }`
- `bitfield : array of bits`

`Attributes of type QRFeature { Feature( method = qrcode ) }:`

- **`code : string`**
- `bitfield : array of bits`

With the QR-Code an additional scenario besides the usual tracking is imaginable. By encoding a URL into the QR-Code, which redirects the user to our POI web application, we can show the currently scanned POI using our application and hence, attract new user for our platform and application in that fashion.

Image or Video-based Features

We utilize specific algorithms to find, locate and map POIs. They will provide their own specification of how they transmit the necessary parameters. This is not yet finalized. Such methods could be based on:

- Characteristic image or silhouette features (e.g. facades of buildings)
- Geometry reconstruction
- Feature descriptors for specific algorithms, e.g. ORB

Relative Coordinates - Embedded / Inside POIs

A POI can be placed relative to another POI, e.g. the painting inside the museum or the apes in the zoo.

`Attributes of type RelativeFeature { Feature( method = relative ) }:`

- **`reference : POIReference`**

Soft Coordinates

We introduce some kind of meta localization for POIs using soft coordinates relative to another POIs. Therefore, we use human-readable instructions as description to navigate the user to a specific POI, such as “Third floor, second door on the left side”. This enables, for instance, a naïve implementation of indoor POIs.

`Attributes of type SoftFeature { Feature( method = soft ) }:`

- **`reference : POIReference`**
- `start_position : type vec2x`
- **`instructions : array of string`**

Contents

To one POI entity are almost always multiple content objects attached. A content object is structured data and must specify a predefined type. The preliminary list of content object types consists of name, label, description, icon, category, tag, image, video, xml3d, contact, and website. Further - not yet specified - content object types could be Google Place Page, Sound, Priority / Importance, Ratings, Aggregated Ratings, Likes, or User Reviews / Comments.
It is always possible to attach multiple content objects of the same type to a POI. They might be of different languages or cultural backgrounds, but could also be for one language. Which one will be preferred is selected by the client. At least one content object should be attached to a POI. Each content object can hold information about the data source, since an attached image might be from another user than the original POI author.

`Attributes of type Content:`

- **`type : enum {`**
  `name, label, description, icon, category, tag,`
  `image, video, xml3d, contact, website`
  `}`
- `lang : type Language`
- `source : type DataSource`

Name, Label, Description

A POI can have a name. This might be particularly used when showing the detailed page of the POI as title. Furthermore, the POI has a description, which is shown on this detailed page. Additionally, a POI can hold at least one label, which is shown on the map to visualize the POI instead of using the name. It can be more expressive rather than a shortened version of the long name.

`Attributes of type Name { Content( type = name ) }:`

- **`name : string`**

`Attributes of type Label { Content( type = label ) }:`

- **`label : string`**

`Attributes of type Description { Content( type = description ) }:`

- **`description : string`**
- `summary : string`

Icon, Symbol, Thumbnail

We can attach an icon or a thumbnail to the POI. It resides at a given URL and might be shared by several POIs.

`Attributes of type Icon { Content( type = icon ) }:`

- **`img : type URL`**

Category, Tags

POIs can be categorized with regard to a specific ontology. Hence, we can attach a category with the reference to this ontology.

`Attributes of type Category { Content( type = category ) }:`

- **`name : string`**
- `ontology : type URL`

Furthermore, it is possible to tag POIs and in this fashion attach meta-data. We can use this tags to find similar POIs with the same set of tags or a selection of POIs with a specific tag, e.g. *UNESCO_World_Heritage_Site*. These tags can be extracted from other resources linked to this POI and then they get the hashtag flag set to indicate this circumstance. Further, it is possible to weight the tags of a POI according to their relevance.

Attributes of type Tag { Content(type = tag) }:

- **name : string**
- hashtag : bool (default = false)
- relevance : float (default = 1.0)

Images, Video Sequences

An essential feature is the attachment of pictures and videos taken from or at a POI by users. They are given as URL and may provide a thumbnail to be previewed. Furthermore, a caption can be provided and the pose of the camera. A transformation can place an image or video within the local coordinate system of the POI. In addition that, Images and videos can be grouped into image_sets to create series of images attached to a POI.

Attributes of type Image { Content(type = image) }:

- **resource : type URL**
- caption : string
- thumbnail : type URL
- camera : type Camera
- transformation : type Transformation
- image_set : string

Attributes of type Video { Content(type = video) }:

- **resource : type URL**
- caption : string
- thumbnail : type URL
- preview : type URL
- camera : type Camera
- transformation : type Transformation
- image_set : string

3D Geometry / Scene

We reference a XML3D document to provide 3D content for a POI. In addition to that, an optional transformation given to position the 3D content within the POI coordinate system.

Attributes of type XML3D { Content(type = xml3d) }:

- **resource : type URL**
- transformation : type Transformation

Contact Point

The contact details of a POI are provided through a ContactPoint content object. It contains at least the addressee and probably provide more information, such as a mail address, phone numbers, a formatted postal address, or a URL of an online contact form.

`Attributes of type ContactPoint { Content( type = contact ) }:`

- **`addressee : string`**
- `email : type MailAddress`
- `telephone : type InternationalPhoneNumber`
- `formatted_address : string`
- `faxNumber : type InternationalPhoneNumber`
- `url : type URL`

Website

The website of this POI, like the website of a business, or other web resources e.g. wikipedia are provided by the website content object referencing a URL.

`Attributes of type Website { Content( type = website ) }:`

- **`url : type URL`**

Time Constraints

Permanent POIs, such as public places, will not have any time constraints. On the other hand, POIs representing an event, which happens only once, may use a time constraint to express the timeframe of this event by the use of validFrom and validThrough. Furthermore, for business it might be relevant to provide information about the opening hours. Thus, it is possible to attach opening hour specifications to express service times as intervals between open and close point in times. Irregular business hours or reoccurring events can be provided through multiple time constraints attached to the POI. It is also possible to provide particular opening hours for school holidays using a time constraint especially for this period.

`Attributes of type TimeConstraint:`

- `opening_hours : array of type OpeningHourSpecification`
- `validFrom : type DateTime`
- `validThrough : type DateTime`

`Attributes of type OpeningHourSpecification:`

-
  ```
  dayOfWeek : enum {
      monday, tuesday, wednesday, thursday, friday,
      saturday, sunday, publicholidays, weekday
  }
  ```
- `opens : type Time`

- `closes : type Time`

Relation between POIs

POIs have relations between each other. These relations can be provided using a tuple specifying the kind of relation and the corresponding POI. The specified relations are a loose coupling of POIs and the handling of the relation may differ according to the intended application. The preliminary list of relation types consist of

- **parent** – reference the superior POI
  e.g. the elephant enclosure refers to its zoo POI
- **child** – counterpart to the parent relation, reference its subordinate POIs
- **adjacent** – reference adjacent POIs, this might be based on some recommendation
- **instanceof** – reference its "class" or "template" POI, enables collections of POIs
  e.g. the POI "Waldspirale" in Darmstadt (Germany) has the instance-of relation to the POI "Building made by the contemporary Austrian artist Hundertwasser"
- **instance** – counterpart to instanceof, reference its instances
- **publictransport** – reference near public transport stations, which are in turn POIs, too
- **host** – reference the hosted POI
  e.g. a permanent POI representing a public place can "host" a temporary POI representing an event
- **locatedat** – counterpart to the host relation

Further kinds of relations can be easily added, like similar-to. Some of them should be later enclosed into this common format.

Attributes of type Relation:

```
● kind : enum {
       parent, child, adjacent, instanceof, publictransport,
       host, locatedat, ...
  }
● href : POIReference
```

Meta Data - Measurements / Requested Calculations / Constraints

We use the meta data structure to attach measurements or requested calculations to a POI. For instance, distances, directions, the visibility and presence of a POI, etc. are such meta data. Similar to the principle of features and content this work with subtypes.

Attributes of type MetaData:

```
● data : enum {
       distancefromposition, directionfromposition, distancetopoi,
       open_now, accessibility, visibility, ... }
```

Attributes of type DistanceFromPosition {

MetaData(data = distancefromposition) }:

- **distance : float**
- unit : string (default = meters)

Attributes of type DirectionFromPosition {
MetaData(data = directionfromposition) }:

- **angle : float**
- unit : string (default = degree)

Attributes of type DistanceToPOI {
MetaData(data = distancetopoi) }:

- **distance : float**
- **href : POIReference**
- unit : string (default = meters)

Attributes of type Accessibility { MetaData(data = accessibility) }:

- score : float
- disability_friendly : boolean (default = false)

Attributes of type Visibility { MetaData(data = visibility) }:

- **score : float**

The list of meta data need to be extended according to further requirements from the use case WPs.

Data Source

We reference the data source of a POI or parts of a POI such as specific content. Thereby, following information are provided, which are almost all not mandatory:

- author – one person or a provider,
- publisher – in case of user generated content this should be the provider,
- license – indicates the possible usage or contains the copyright notice,
- href – is a fully qualified URL to retrieve the POI or this part of the POI (self-reference)
- created – timestamp of the creation
- updated – timestamp of the last modification
- revision – indicates the number of changes

At least the POI need to provide this data source reference information including the self-reference attribute (href). For compound POIs it is possible to overwrite some of these information by providing specific ones for a component, e.g. an image as part of the attached content. Thus, it is possible to retrieve updated information for a POI by querying several providers using different URLs.

Attributes of type DataSource:

- `author : string`
- `publisher : string`
- `license : string`
- `href : type URL`
- `created : type DateTime`
- `updated : type DateTime`
- `revision : int`
- `change_frequency : enum {`
```
    always, hourly, daily, weekly, monthly, yearly, never
}
```

Attributes of type MainDataSource { DataSource }:

- **`href : type URL`**

Utility Types

Beside the basic type, such as string, int, float, etc., we utilize some more advanced types, which are not explicitly specified. We catch up on the missing utility types here with regard to existing standards.

- **UUID**
  A string containing an identifier as defined in RFC 4122, Version 4
- **URL**
  See http://schema.org/URL
- **POIReference**
  Union of { UUID, URL }
  It is either the pure identifier of a POI within the same data source or the fully qualified URL to retrieve the POI.
  It might be also possible to embed the referenced POI in place or at least a subset to significantly reduce the amount of queries.
- **DateTime**
  See http://schema.org/DateTime and http://www.w3.org/TR/NOTE-datetime
- **Time**
  See http://schema.org/Time and http://www.w3.org/TR/xmlschema-2/#time
- **Language**
  An IETF language tag as normalized in RFC 5646
- **Transformation**
  Attributes of type Transformation:
  - `translation : type vec2x (default = (0.0, 0.0, 0.0))`
  - `rotation : type quaternion (default = [1.0, (0.0, 0.0, 0.0)])`
  - `scaling : type vec2x (default = [1.0, 1.0, 1.0])`
  - `matrix : type Matrix (= MatTrans * MatRot * MatScale)`
- **Camera**

The camera structure holds information about the position and orientation of the camera with respect to pictures or videos taken from and associated to some POI. Moreover, the field of view can be provided in degrees horizontal (x) and vertical (y).

Attributes of type Camera:

- **`position : type vec2x`**
- `orientation : type quaternion`
- `fov : type vec2`

- **Matrix**
  An array of float (length of 16)
- **vec2**
  An array of float (length 2)
- **vec2x**
  An array of float (length 2-3) with an optional z-component. It is 0.0 by default.
- **vec3**
  An array of float (length 3) with a mandatory z-component.
- **quaternion**
  An array of float (length 4)
- **InternationalPhoneNumber**
  A string containing the phone number in the international format as recommended by the ITU in the format E.123
- **MailAddress**
  A string containing the email address as defined in RFC 6530

Conclusions

The proposal presents a very flexible format for the specification of POIs. It handles temporary POIs, such as events, in the same way as permanent POIs, like public places. The proposed approach supports out-of-the-box almost all intended use cases and is easily extensible to support the remaining ones.
The migration of existing POI databases to use the proposed format is as straightforward as all evaluated existing formats, including them of the FIcontent partners, utilize only a subset of the functionality provided by the proposed format.
This format is quite lightweight and has only a small overhead on the actual information. On the other hand it is so flexible and feature-rich to enable beyond state-of-the-art use cases, such as AR, embedded and relative POIs, alternative positioning methods and arbitrary content associated with a POI.
Altogether, it is the foundation for innovative and promising POI applications, especially in the smart city guide and gaming context.

Open Questions [not all are still open]

- Visibility [region] – some indicator for the presence, landmarks have a huge visibility
- Instancing – Bus/Tram, instances for Bus of specific line
- User generated Content in General
- Postal Readdressing
- 3D dimension – height of POI, bounding sphere instead of circle, extruded polygon (prism) for buildings - best fit is a cylindrical bounding volume
- Is a category more than a hierarchical tag? – e.g. own description, image, symbol, subcategories
- Orthogonal construct of tags and categories – who defines these across data sources? - reference external ontologies
- Keep Track of Data Source, Originator/Author, Processing
- Self reference
  - a POI search service could return a result list of URIs representing POIs - then the service holds only an index but not the actual POI data
  - these POIs could be indexed from different data sources
  - or it could return a list of complete POI data - then the client need some reference for each POI - is that the GUID?
- How to handle duplicates within one database and between data sources - how to merge/fuse the data?
- People could be a POI if they share common interest with you or are friends [Orange]
- Reference for the exact shape the CityGML data?
- License information, author, copyright
- language of the attached content - localization?
- API key with a specific result request profile? - e.g.
  - omit license information
  - only GPS localization feature
  - feature descriptors for specific algorithm of preprocessed marker image
  - only specific content language
  - GUID resolution to URIs or embedded JSON
- How to manage routes (list of POIs)?
- Manage news and information as POI?
- What about comments on pictures attached to a POI?
- Personal alias names for POIs - like DFKI -> Office 0.75 (My Office), @ Home

Appendix: Examples of Existing Solutions

Orange's Format

http://goo.gl/V56I9y [SharePoint; WP3, T3.1]

(basic information)

- id
- Name
- Phone
- E-mail
- Fax
- url - *Used to link to the web site of the original information provider, to a page dedicated to the POI*
- Urlphoto - *Image to illustrate the POI*
- codephoto - *Used only for weather information pictures*
- Imgalt - *Text to be written alternatively to the image*

(location information)

- Place - *Name of the place related to the POI*
- locality - *City name*
- zipcode
- address
- Address2 - *Used if needed to add extra address information*
- lat - *Latitude*
- lon - *Longitude*
- geoprecision - *Can be used to describe the precision of location information*
- dist - *Distance from the center of the circle in the case of a circular search (added to a search result)*

(detailed description)

- extract - *Summary/description about the POI*
- idcatmvl1 - *id of MVL level 1 category*
- Idcatmvl2 - *id of MVL level 2 category*
- idcatmvl3 - *id of MVL level 3 category*
- thematic - *Additional category information for the POI*
- cat118712 - *Addition category information (only for 118712 datasource)*
- idlayer - *id of a layer on the map, used to quickly access a category of POI*
- artist
- horaire - *Opening hours*
- Price - *Price information for a POI (e.g. entrance fee)*
- Services - *Additional information about POI*
- access - *Acces information (e.g. for disabled persons…)*

(event-related fields)

- date - *Date to be displayed (only for an event)*
- dateserveur1 - *Starting date (only for an event)*
- dateserveur2 - *End date (only for an event)*

(datasource information)

- iddatasource
- datasourcelibel - *Name of the datasource*

(link with other POI)

- listEvents - *POI objects associated with this POI (e.g.: concert POI linked to a concert hall POI)*

(rating and comments)

- Cv_nbofopinions - *Number of votes*
- Cv_avgscore - *Average rating*
- Cv_avisfirst - *Latest comment on the POI*
- Cv_urlavis - *Link to the datasource where the comment was done*

(misc)

- realtimeinfo - *1 or null depending on whether there is real-time information or not*

Example:

```
{
    "id": "FRTES00000000000018036848",
    "iddatasource": "99",
    "name": "Cinéma Les Baladins",
    "dateserveur1": null,
    "dateserveur2": null,
    "access": null,
    "extract": "",
    "thematic": "salles de cinéma",
    "locality": "Lannion",
    "free": null,
    "place": null,
    "price": "",
    "fax": null,
    "artist": null,
    "url": null,
    "wgs84": null,
    "date": null,
    "allcontent": null,
    "horaire": "",
    "zipcode": "22300",
    "address": "34 avenue du Général de Gaulle",
    "address2": null,
    "phone": "02 96 37 26 10",
    "email": null,
    "services": "",
    "sticker": null,
    "urlphoto": "",
    "imgalt": null,
    "lat": "48.7278685071015",
    "lon": "-3.46049574601966",
    "geoprecision": null,
    "idcatmvl1": "600",
    "idcatmvl2": "602",
    "idcatmvl3": "6011",
    "idlayer": null,
    "realtimeinfo": null,
    "code": null,
    "idplace": "3771",
    "idplaceevent": null,
    "cv_nbofopinions": "",
    "cv_avgscore": "0",
    "cv_avisfirst": "",
    "cv_urlavis": "",
    "cat118712": "191200",
    "datasourcelibel": "118712.fr",
    "favori": 0,
    "listevents": [
      {
        "id": "240625",
        "iddatasource": "10",
        "name": "La Nouvelle guerre des boutons",
        "dateserveur1": "2011-10-19T01:00:00Z",
        "dateserveur2": "2011-10-25T01:00:00Z",
        "access": null,
        "extract": null,
        "thematic": "Com\u00e9die\u00a0\u00a0|\u00a0\u00a0dur\u00e9e
1H40\u00a0\u00a0",
        "locality": "Lannion",
        "free": null,
        "place": "Les Baladins",
        "price": null,
        "fax": null,
        "artist": null,
        "url": "http:\/\/www.cine.orange.fr\/film\/la-nouvelle-guerre-des-boutons",
        "wgs84": null,
        "date": "Du 2011-10-19 au 2011-10-25",
        "allcontent": null,
        "horaire": "jeudi 14h30 dimanche 14h45 lundi mardi 18h30",
        "zipcode": "22300",
        "address": "34 avenue du G?n?ral de Gaulle",
        "address2": null,
        "phone": "02 96 37 26 10",
        "email": null,
        "services": null,
        "sticker": null,
        "urlphoto":
"http:\/\/cine.woopic.com\/photos_120_90\/232\/104\/la-nouvelle-guerre-des-boutons,551
144.jpg",
        "imgalt": null,
        "lat": "48.7278685",
        "lon": "-3.4604957",
        "geoprecision": null,
        "idcatmvl1": "600",
        "idcatmvl2": "604",
        "idcatmvl3": "6011",
        "idlayer": null,
        "realtimeinfo": null,
        "code": null,
        "idplace": null,
        "idplaceevent": "3771",
        "datasourcelibel": "Cin\u00e9ma -Cin\u00e9fil",
        "favori": 0
      }
    ]
}
```

FOKUS’ Format

http://mashweb.fokus.fraunhofer.de:3008/api/pois?_id=4f4f4c27d4374e8001000008

```
{
    "name": "Brandenburger Tor",
    "description": "Das Brandenburger Tor in Berlin steht ...",
    "image": "http://tld/Brandenburger_Tor_abends.jpg",
    "dbpedia": "http://dbpedia.org/page/Brandenburg_Gate",
    "city": "4f4f4c27d4374e8001000007",
    "_id": "4f4f4c27d4374e8001000008",
    "checkins": {
        "total": 0
    },
    "comments": {
        "total":1
    },
    "rating": 5,
    "entrancefees": [
        "Children: 0 EUR",
        " Adults: 20 EUR",
        " Students: 10 EUR"
    ],
    "openinghours": [
        "Monday-Friday: 10:00-20:00",
        " Saturday: 10:00-18:00",
        " Sunday: 10:00-14:00"
    ],
    "contact": {
        "website": "http://www.berlin.de/orte/sehenswert/brb-tor/",
        "email": "info@brandenburgertor.de",
        "address": "Pariser Platz, 11017 Berlin",
        "telephone": [
            "Tel: +49 30 1234567",
            " Fax: +49 30 1234568"
        ]
    },
    "publictransport": [
        "S-Bahn: Unter den Linden",
        " S-Bahn: Friedrichstraße",
        " U-Bahn: Friedrichstrße",
        "  S-Bahn: Lehrter Bahnhof"
    ],
    "location": {
        "latitude": 52.516289,
        "longitude": 13.377711
    }
}
```

W3C POI Core

http://www.w3.org/2010/POI/documents/Core/core-20111216.html

```
<poi id="StataCenter" xml:lang="en-US" >
```

```
    <label term="primary" xml:lang="en-US">Stata Center</label>
    <label term="primary" xml:lang="es">Stata Centro</label>
    <label>Ray and Maria Stata Center</label>
    <label>Building 32</label>
    <label>Gates Tower</label>
    <label>Dreyfoos Tower</label>

    <location id="location1">
        <point latitude="42.360890561289295" longitude="-71.09139204025269">
            <Point><posList>42.360890561289295 -71.09139204025269</posList>
            </Point>
        </point>
        <point id="mainpoint" latitude="27.174799" longitude="78.042111"
altitude="10m" srsName="http://www.opengis.net/def/crs/EPSG/0/4979">
            <Point><posList>27.174799 78.042111 10</posList></Point>
        </point>
        <point term="center" latitude="27.174799" longitude="78.042111">
            <Point><posList>27.174799 78.042111</posList></Point>
        </point>

        <polygon>
            <Polygon><posList>
                42.360890561289295 -71.09139204025269
                42.361176 -71.09018 42.36272976137689 -71.09049081802368
                42.36318955298668 -71.09677791595459
                42.363617631805496 -71.10156297683716
                42.360890561289295 -71.09139204025269
            </posList></Polygon>
        </polygon>

        <undetermined/>

    </location>

    <category
scheme="http://www.census.gov/cgi-bin/sssd/naics/naicsrch?search=2007%20NAICS%20Search
&amp;code=611310" xml:lang="en-US">
        <value>Colleges, Universities, and Professional Schools</value>
        <!-- <association type="child-of"
reference="http://www.census.gov/cgi-bin/sssd/naics/naicsrch?search=2007%20NAICS%20Sea
rch&amp;code=6113"/> -->
    </category>

</poi>
```

openPOI by OGC

http://openpoi.ogcnetwork.net/api.php

```
{
    "labels": [
    {
        "term": "primary",
        "scheme": null,
        "typename": "LABEL",
        "id": null,
        "value": "Hynes Convention Center",
        "href": null,
        "type": null,
        "created": "2013-04-03T16:30:52.813804-04",
        "updated": "2013-04-18T15:58:15.829972-04",
        "deleted": "",
        "author": {
            "term": "publisher",
            "scheme": null,
            "typename": "AUTHOR",
            "id": "http:\/\/geonames.org",
            "value": "GeoNames",
            "href": null,
            "type": "text\/plain",
            "created": "",
            "updated": "",
            "deleted": "",
            "author": null,
            "license": null,
            "lang": null,
            "base": null,
            "myid": "88f72b90-5cbe-4a7a-b1c1-6c9d83cd3998",
            "parentid": null,
            "changed": false
        },
        "license": null,
        "lang": null,
        "base": null,
        "myid": "890699b1-292b-46ec-a141-10afecb32761",
        "parentid": "f0aa9f2f-c091-4dc2-9282-3a5dbbec8314",
        "changed": true
    }
],

"descriptions": [
    {
        "typename":"DESCRIPTION",
        "id":null,
        "value":"The John B. Hynes Veterans...",
        "href":null,
        "type":null,
        "created":"2013-04-18T15:58:15.829972-04",
        "updated":"2013-04-18T15:58:15.829972-04",
        "deleted":"",
        "author":null,
            "license":null,
            "lang":null,
            "base":null,
            "myid":"d3b2ff1b-5405-4170-848e-03ebc137dd00",
            "parentid":"f0aa9f2f-c091-4dc2-9282-3a5dbbec8314",
            "changed":true
        }
    ],

    "categories": [
        {
            "term":"featureclass",
            "scheme":"http:\/\/www.geonames.org\/export\/codes.html",
            "typename":"CATEGORY",
            "id":null,
            "value":"S",
            "href":null,
            "type":null,
            "created":"2013-04-03T16:30:52.813804-04",
            "updated":"2013-04-18T15:58:15.829972-04",
            "deleted":"",
            "author": {
                ...
            },
            "license":null,
            "lang":null,
            "base":null,
            "myid":"959decfc-ec9d-4a57-a17a-5eff5ebee7e5",
            "parentid":"f0aa9f2f-c091-4dc2-9282-3a5dbbec8314",
            "changed":true
        },
        {
            "term":"featurecode",
            "scheme":"http:\/\/www.geonames.org\/export\/codes.html",
            "typename":"CATEGORY",
            "id":null,
            "value":"BLDG",
            "href":null,
            "type":null,
            "created":"2013-04-03T16:30:52.813804-04",
            "updated":"2013-04-18T15:58:15.829972-04",
            "deleted":"",
            "author": {
                ...
            },
            "license":null,
            "lang":null,
            "base":null,
            "myid":"051e4c1e-2cb3-4531-9e54-1400bbffdd19",
            "parentid":"f0aa9f2f-c091-4dc2-9282-3a5dbbec8314",
            "changed":true
        }
    ],
    "times":[
    ],
    "links": [
        {
            "term":"related",

"scheme":"http:\/\/www.iana.org\/assignments\/link-relations\/link-relations.xml",
            "typename":"LINK",
            "id":"4940211",
            "value":null,
            "href":"http:\/\/www.geonames.org\/4940211",
            "type":null,
            "created":"2013-04-03T16:30:52.813804-04",
            "updated":"2013-04-18T15:58:15.829972-04",
            "deleted":"",
            "author":null,
            "license":null,
            "lang":null,
            "base":"http:\/\/www.geonames.org",
            "myid":"04406ada-4b22-4a4b-a902-8272a3d25e88",
            "parentid":"f0aa9f2f-c091-4dc2-9282-3a5dbbec8314",
            "changed":true
        }
    ],
    "metadata":null,
    "location": {
        "points": [
            {
                "srsname":null,
                "poslist":"42.34732 -71.08505",
                "term":"centroid",
                "scheme":null,
                "typename":"POINT",
                "id":null,
                "value":null,
                "href":null,
                "type":null,
                "created":"2013-04-03T16:24:04.369944-04",
                "updated":"2013-04-03T16:24:04.369944-04",
                "deleted":"",
                "author": {
                    ...
                },
                "license":null,
                "lang":null,
                "base":null,
                "myid":"4c624857-6f3e-47c6-9d18-1315e86e9681",
                "parentid":"e34adf01-853c-4d52-a6b4-1ebd4437abde",
                "changed":false
            }
        ],
        "lines":[],
        "polygons":[],
        "address":null,
        "undetermined":null,
        "relationships":[],
        "typename":"LOCATION",
        "id":null,
        "value":null,
        "href":null,
        "type":null,
        "created":"2013-04-03T16:29:05.044526-04",
        "updated":"2013-04-03T16:29:05.044526-04",
        "deleted":"",
        "author":null,
        "license":null,
        "lang":null,
        "base":null,
        "myid":"e34adf01-853c-4d52-a6b4-1ebd4437abde",
        "parentid":"f0aa9f2f-c091-4dc2-9282-3a5dbbec8314",
        "changed":false
    },
    "typename":"POI",
    "id":"f0aa9f2f-c091-4dc2-9282-3a5dbbec8314",
    "value":null,
    "href":null,
    "type":null,
    "created":"2013-04-03T16:29:44.827204-04",
    "updated":"2013-04-03T16:29:44.827204-04",
    "deleted":"",
    "author":null,
    "license":null,
    "lang":null,
    "base":"http:\/\/openpoi.ogcnetwork.net\/pois",
    "myid":"f0aa9f2f-c091-4dc2-9282-3a5dbbec8314",
    "parentid":null,
    "changed":false
}
```

Google Places

https://developers.google.com/places/documentation/details

- address_components[] *is an array of separate address components used to compose a given address. For example, the address "111 8th Avenue, New York, NY" contains separate address components for "111" (the street number, "8th Avenue" (the route), "New York" (the city) and "NY" (the US state). Each address_component typically contains:*
    - types[] is an array indicating the *type* of the address component.
    - long_name is the full text description or name of the address component.
    - short_name is an abbreviated textual name for the address component, if available. For example, an address component for the state of Alaska may have a long_name of "Alaska" and a short_name of "AK" using the 2-letter postal abbreviation.
- *An* events[] array *or one or more <event> elements provide information about current events happening at this Place. Up to 10 events are returned, ordered by start time. For information about events, please read Events in the Places API. Each event contains:*
    - event_id: The unique ID of this event.
    - start_time: The event's start time, expressed in Unix time.
    - summary: A textual description of the event. This property contains a string, the contents of which are not sanitized by the server. Your application should be prepared to prevent or deal with attempted exploits, if necessary.
    - url: A URL pointing to details about the event. This property will not be returned if no URL was specified for the event.
- formatted_address *is a string containing the human-readable address of this place. Often this address is equivalent to the "postal address," which sometimes differs from country to country. This address is generally composed of one or more address_component fields.*
- formatted_phone_number *contains the Place's phone number in its local format. For example, the formatted_phone_number for Google's Sydney, Australia office is (02) 9374 4000.*
- geometry *contains the following information:*
    - location contains the geocoded latitude,longitude value for this place.
- icon *contains the URL of a suggested icon which may be displayed to the user when indicating this result on a map*
- id *contains a unique stable identifier denoting this place. This identifier may not be used to retrieve information about this place, but can be used to consolidate data about this Place, and to verify the identity of a Place across separate searches. As ids can occasionally change, it's recommended that the stored id for a Place be compared with the id returned in later Details requests for the same Place, and updated if necessary.*
- international_phone_number *contains the Place's phone number in international*

format. International format includes the country code, and is prefixed with the plus (+) sign. For example, the formatted_phone_number for Google's Sydney, Australia office is +61 2 9374 4000.

- name *contains the human-readable name for the returned result. For establishment results, this is usually the canonicalized business name.*
- opening_hours *may contain the following information:*
    - open_now is a boolean value indicating if the Place is open at the current time.
    - periods[] is an array of opening periods covering seven days, starting from Sunday, in chronological order. Each period may contain:
        - open contains a pair of day and time objects describing when the Place opens:
            - day a number from 0–6, corresponding to the days of the week, starting on Sunday. For example, 2 means Tuesday.
            - time may contain a time of day in 24-hour hhmm format (values are in the range 0000–2359). The time will be reported in the Place's timezone.
        - close contains a pair of day and time objects describing when the Place closes.
- photos[] — *an array of photo objects, each containing a reference to an image. A Place Details request may return up to ten photos. More information about Place Photos and how you can use the images in your application can be found in the Place Photos documentation. A photo object is described as:*
    - photo_reference — a string used to identify the photo when you perform a Photo request.
    - height — the maximum height of the image.
    - width — the maximum width of the image.
    - html_attributions[] — *contains any required attributions. This field will always be present, but may be empty.*
- price_level — *The price level of the Place, on a scale of 0 to 4. The exact amount indicated by a specific value will vary from region to region. Price levels are interpreted as follows:*
    - 0 — Free
    - 1 — Inexpensive
    - 2 — Moderate
    - 3 — Expensive
    - 4 — Very Expensive
- rating *contains the Place's rating, from 0.0 to 5.0, based on user reviews.*
- reference *contains a token that can be used to query the Details service in future. This token may differ from the reference used in the request to the Details service. It is recommended that stored references for Places be regularly updated. Although this token uniquely identifies the Place, the converse is not true: a Place may have many valid reference tokens.*
- reviews[] *a JSON array of up to five reviews. If a language parameter was specified in*

the Place Details request, the Places Service will bias the results to prefer reviews written in that language. Each review consists of several components:
    - aspects contains a collection of AspectRating objects, each of which provides a rating of a single attribute of the establishment. The first object in the collection is considered the primary aspect. Each Aspect
        - Rating is described as:type the name of the aspect that is being rated. eg. atmosphere, service, food, overall, etc.
        - rating the user's rating for this particular aspect, from 0 to 3.
    - author_name the name of the user who submitted the review. Anonymous reviews are attributed to "A Google user".
    - author_url the URL to the users Google+ profile, if available.
    - text contains the user's review. When reviewing a location with Google Places, text reviews are considered optional; therefore, this field may by empty.
    - time the time that the review was submitted, measured in the number of seconds since since midnight, January 1, 1970 UTC.
- types[] *contains an array of feature types describing the given result. See the list of supported types for more information. XML responses include multiple <type> elements if more than one type is assigned to the result.*
- url *contains the official Google Place Page URL of this establishment. Applications must link to or embed the Google Place page on any screen that shows detailed results about this Place to the user.*
- utc_offset *contains the number of minutes this Place's current timezone is offset from UTC. For example, for Places in Sydney, Australia during daylight saving time this would be 660 (+11 hours from UTC), and for Places in California outside of daylight saving time this would be -480 (-8 hours from UTC).*
- vicinity *lists a simplified address for the Place, including the street name, street number, and locality, but not the province/state, postal code, or country. For example, Google's Sydney, Australia office has a vicinity value of 48 Pirrama Road, Pyrmont.*
- website *lists the authoritative website for this Place, such as a business' homepage.*

Example:
```
{
   "html_attributions" : [],
   "result" : {
      "address_components" : [
         {
            "long_name" : "48",
            "short_name" : "48",
            "types" : [ "street_number" ]
         },
         {
            "long_name" : "Pirrama Road",
            "short_name" : "Pirrama Road",
            "types" : [ "route" ]
         },
         {
            "long_name" : "Pyrmont",
            "short_name" : "Pyrmont",
            "types" : [ "locality", "political" ]
         },
         {
            "long_name" : "NSW",
            "short_name" : "NSW",
            "types" : [ "administrative_area_level_1", "political" ]
         },
         {
            "long_name" : "AU",
            "short_name" : "AU",
            "types" : [ "country", "political" ]
         },
         {
            "long_name" : "2009",
            "short_name" : "2009",
            "types" : [ "postal_code" ]
         }
      ],
      "events" : [
        {
          "event_id" : "9lJ_jK1GfhX",
          "start_time" : 1293865200,
          "summary" : "<p>A visit from author John Doe, who will read from his latest
book.</p>
                       <p>A limited number of signed copies will be available.</p>",
          "url" : "http://www.example.com/john_doe_visit.html"
        }
      ],
      "formatted_address" : "48 Pirrama Road, Pyrmont NSW, Australia",
      "formatted_phone_number" : "(02) 9374 4000",
      "geometry" : {
         "location" : {
           "lat" : -33.8669710,
           "lng" : 151.1958750
         }
      },
      "icon" :
"http://maps.gstatic.com/mapfiles/place_api/icons/generic_business-71.png",
      "id" : "4f89212bf76dde31f092cfc14d7506555d85b5c7",
      "international_phone_number" : "+61 2 9374 4000",
      "name" : "Google Sydney",
      "rating" : 4.70,
      "reference" :
"CnRsAAAA98C4wD-VFvzGq-KHVEFhlHuy1TD1W6UYZw7KjuvfVsKMRZkbCVBVDxXFOOCM108n9PuJMJxeAxix3
WB6B16c1p2bY1ZQyOrcu1d9247xQhUmPgYjN37JMo5QBsWipTsnoIZA9yAzA-0pnxFM6yAcDhIQbU0z05f3xD3
m9NQnhEDjvBoUw-BdcocVpXzKFcnMXUpf-nkyF1w",
      "reviews" : [
         {
            "aspects" : [
               {
                  "rating" : 3,
                  "type" : "quality"
               }
            ],
            "author_name" : "Simon Bengtsson",
            "author_url" : "https://plus.google.com/104675092887960962573",
            "text" : "Just went inside to have a look at Google. Amazing.",
            "time" : 1338440552869
         },
         {
           "aspects" : [
              {
                 "rating" : 3,
                 "type" : "quality"
              }
             ],
            "author_name" : "Felix Rauch Valenti",
            "author_url" : "https://plus.google.com/103291556674373289857",
            "text" : "Best place to work :-)",
            "time" : 1338411244325
         },
         {
           "aspects" : [
              {
                 "rating" : 3,
                 "type" : "quality"
              }
             ],
            "author_name" : "Chris",
            "text" : "Great place to work, always lots of free food!",
            "time" : 1330467089039
         }
      ],
      "types" : [ "establishment" ],
      "url" : "http://maps.google.com/maps/place?cid=10281119596374313554",
      "vicinity" : "48 Pirrama Road, Pyrmont",
      "website" : "http://www.google.com.au/"
   },
   "status" : "OK"
}
```

schema.org

http://www.schema.org/docs/full.html

- http://schema.org/Event
- http://schema.org/Place
    - http://schema.org/LandmarksOrHistoricalBuildings
    - http://schema.org/TouristAttraction

Properties of type TouristAttraction:

Property	**Expected Type**
Properties from Thing	
additionalType	URL
description	Text
image	URL
name	Text
sameAs	URL
url	URL
Properties from Place	
address	PostalAddress
aggregateRating	AggregateRating
containedIn	Place
event	Event
events	Event
faxNumber	Text
geo	GeoCoordinates or GeoShape
globalLocationNumber	Text
interactionCount	Text
isicV4	Text
logo	ImageObject or URL
map	URL
maps	URL
openingHoursSpecification	OpeningHoursSpecification
photo	ImageObject or Photograph
photos	ImageObject or Photograph
review	Review
reviews	Review
telephone	Text

OpenStreetMap

http://www.openstreetmap.org/api/0.6/way/19046101

```
<?xml version="1.0" encoding="UTF-8"?>
<osm version="0.6" generator="OpenStreetMap server" copyright="OpenStreetMap
and contributors" attribution="http://www.openstreetmap.org/copyright"
license="http://opendatacommons.org/licenses/odbl/1-0/">
  <way id="19046101" visible="true" timestamp="2013-07-28T10:16:27Z"
version="31" changeset="17123376" user="Shmias" uid="88961">
    <nd ref="197325375"/>  <nd ref="2166130493"/>
    <nd ref="2166130497"/>  <nd ref="2166130500"/>
    <nd ref="2166130504"/>  <nd ref="2166130502"/>
    <nd ref="2166130499"/>  <nd ref="2166130495"/>
    <nd ref="197325380"/>  <nd ref="2245301690"/>
    <nd ref="2268185839"/>  <nd ref="2268185809"/>
    <nd ref="284557644"/>  <nd ref="2268185779"/>
    <nd ref="2245301689"/>  <nd ref="2166130489"/>
    <nd ref="2166130486"/>  <nd ref="2166130482"/>
    <nd ref="2166130478"/>  <nd ref="2166130475"/>
    <nd ref="2166130465"/>  <nd ref="2166130461"/>
    <nd ref="2166130458"/>  <nd ref="2166130454"/>
    <nd ref="2166130450"/>  <nd ref="2166130452"/>
    <nd ref="2166130456"/>  <nd ref="2166130460"/>
    <nd ref="2166130463"/>  <nd ref="2166130473"/>
    <nd ref="2166130477"/>  <nd ref="2166130480"/>
    <nd ref="2166130484"/>  <nd ref="2166130487"/>
    <nd ref="2166130491"/>  <nd ref="284557960"/>
    <nd ref="197325375"/>
    <tag k="building" v="tower"/>
    <tag k="building:part" v="yes"/>
    <tag k="building:shape" v="sphere"/>
    <tag k="ele" v="40"/>
    <tag k="height" v="237"/>
    <tag k="min_height" v="205"/>
    <tag k="roof:shape" v="dome"/>
    <tag k="wheelchair" v="no"/>
  </way>
</osm>
```

geonames.org

http://www.geonames.org/6944620/hotel-continental-saarbruecken.html

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<rdf:RDF
 xmlns:cc="http://creativecommons.org/ns#"
 xmlns:dcterms="http://purl.org/dc/terms/"
 xmlns:foaf="http://xmlns.com/foaf/0.1/"
 xmlns:gn="http://www.geonames.org/ontology#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:wgs84_pos="http://www.w3.org/2003/01/geo/wgs84_pos#">

<gn:Feature rdf:about="http://sws.geonames.org/6944620/">
 <rdfs:isDefinedBy rdf:resource="http://sws.geonames.org/6944620/about.rdf"/>
 <gn:name>Hotel Continental Saarbrücken</gn:name>
 <gn:featureClass rdf:resource="http://www.geonames.org/ontology#S"/>
 <gn:featureCode rdf:resource="http://www.geonames.org/ontology#S.HTL"/>
 <gn:countryCode>DE</gn:countryCode>
 <wgs84_pos:lat>49.2367</wgs84_pos:lat>
 <wgs84_pos:long>6.99658</wgs84_pos:long>
 <gn:parentFeature rdf:resource="http://sws.geonames.org/2842635/"/>
 <gn:parentCountry rdf:resource="http://sws.geonames.org/2921044/"/>
 <gn:parentADM1 rdf:resource="http://sws.geonames.org/2842635/"/>
 <gn:nearbyFeatures
  rdf:resource="http://sws.geonames.org/6944620/nearby.rdf"/>
 <gn:locationMap rdf:resource="http://www.geonames.org/6944620/hotel-
  continental-saarbruecken.html"/>
</gn:Feature>

<foaf:Document rdf:about="http://sws.geonames.org/6944620/about.rdf">
 <foaf:primaryTopic rdf:resource="http://sws.geonames.org/6944620/"/>
 <cc:license rdf:resource="http://creativecommons.org/licenses/by/3.0/"/>
 <cc:attributionURL rdf:resource="http://sws.geonames.org/6944620/"/>
 <cc:attributionName
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">GeoNames</cc:attribution
Name>
 <dcterms:created rdf:datatype="http://www.w3.org/2001/XMLSchema#date">
  2009-08-04</dcterms:created>
 <dcterms:modified rdf:datatype="http://www.w3.org/2001/XMLSchema#date">
  2009-08-04</dcterms:modified>
</foaf:Document>

</rdf:RDF>
```