Web APIs for environmental data - state of the art investigation

Technical Report · June 2016

DOI: 10.13140/RG.2.2.17817.83047

1 author:

Peter Taylor
The Commonwealth Scientific and Industrial Research Organisation

Web APIs for environmental data

State of the art investigation

Peter Taylor

30 June 2016

Contents

Acknowledgments

This work has, for the most part, been funded through a Water Information Research and Development Alliance (WIRADA) between CSIRO and the Australian Bureau of Meteorology. The author wishes to thank Simon Cox (CSIRO), Dominic Lowe (Bureau of Meteorology) and Robert Power (CSIRO) for their thorough reviews.

Executive summary

Environmental data provides huge value to society. It is used in many ways, from local planning and monitoring, to global modelling and analysis. Organisations publish environmental data using a range of techniques, but increasingly it involves the use of the Internet (web sites, email, social media, mobile devices, automated notifications etc.). Publishing data for use in current and future software is an important way value is gained from environmental data. Globally, organisations are increasingly using Web APIs (Application Programming Interfaces) to supply data to multiple audiences, and creating new businesses in the process.

This report summarises existing practices for the publication of environmental data using Web APIs. The aim is to provide organisations with a well-informed basis to design consistent, cross-organisation Web APIs to access a range of environmental data products. While this report was conducted in a hydrology-specific activity (WIRADA), it takes a broader look at environmental data to contribute to a cross-domain view of Web APIs.

The Web APIs covered in this report include a selection from the USGS, NOAA, the UK Met Office, and the UK environmental agency, among others. There are also some commercial and open source offerings that have no direct affiliation with environmental agencies. In our selection we focussed on organisations with similar data holdings to the Australian Bureau of Meteorology and those that had progressed to public facing Web APIs.

Generally the practices varied quite a lot across these organisations. The specifics of their Web API depend on the time they were developed and their target domain. Some use older Web API practices (e.g. simple key-value pair APIs), while others use newer approaches (e.g. linked data). This raises an important point: organisations must find a balance between adopting widely implemented Web API practices and those that are still emerging, but which may offer greater capabilities in the future.

Some organisations (e.g. NOAA and UK Met Office) look to be using Web APIs as a way of bringing existing legacy systems together and providing a more cohesive API experience for developers. However these approaches are in their infancy. They do not appear to be using well-defined, reusable APIs, which we would suggest to be adopted as-is. Yet there are lessons to be learned: some have identified basic abstractions that may be reusable, such as separation of value-types (text, image, scalar); observations and forecasts; points, grids, and trajectories, among others. These may then be used as generic access mechanisms across sub-domains, where such concepts are generally consistent.

In this report we identify and provide brief overviews of the state of the art in Web API description languages (5.1), common response types (5.2) and management of Web APIs (5.3). This is an area of huge growth, following the proliferation of APIs on the Web. Consequently many of these practices are still immature, so we suggest being cautious when selecting API description languages and common response types.

The suite of available management tools reflects the need to monitor and manage access to APIs. These tools appear more mature than service description languages, as there are distinct commercial opportunities in providing such services. The open source offerings have a range of functionality out of the box, and appear to be easily adapted for use in particular scenarios.

We provide general recommendations in three areas of provisioning Web APIs:

- General: documentation styles, API presentation, building developer community;
- Technical: use of automated API documentation, API description languages, API managers, use of the HTTP standard, error and link encoding, versioning, being consistent;
- Modelling the domain: modelling resources, use of domain-driven modelling, involving domain experts, use of abstractions, and relevant external activities.

These recommendations coupled with the details of existing domain APIs and tools should provide a basis for more detailed design of environmental Web APIs.

1 Introduction

Environmental monitoring agencies deliver huge value through data delivery. From primary industries and infrastructure to the general public, there are many consumers of environmental data. Industries such as aviation, agriculture, emergency management, mining, defence, planning, and health all make use of environmental data. People access environmental data through a variety of channels but delivery is increasingly driven by software. Whether powering websites, social media, email, or publications, data must be timely, reliable and fit for use.

Increasingly organisations are providing web-based access to data to allow development of innovative applications and third-party products and services. This report reviews current practices of Web-based Application Programming Interfaces (APIs) for delivery of environmental data. Also referred to as Web services, APIs are increasingly being used for data delivery through multiple media channels. We focus on practices that emerged through the 'Web 2.0' progression of Web technologies. These technologies addressed rapid delivery of data to Web-based applications, such as Facebook, Twitter, and so on. These approaches have strongly influenced data delivery on the modern Web.

All environmental organisations have an opportunity to improve their ability to assist industries and drive creation of new businesses by providing clear and descriptive access to its data through Web APIs. In this report we cover four distinct aspects of the problem, as shown in Figure 1.

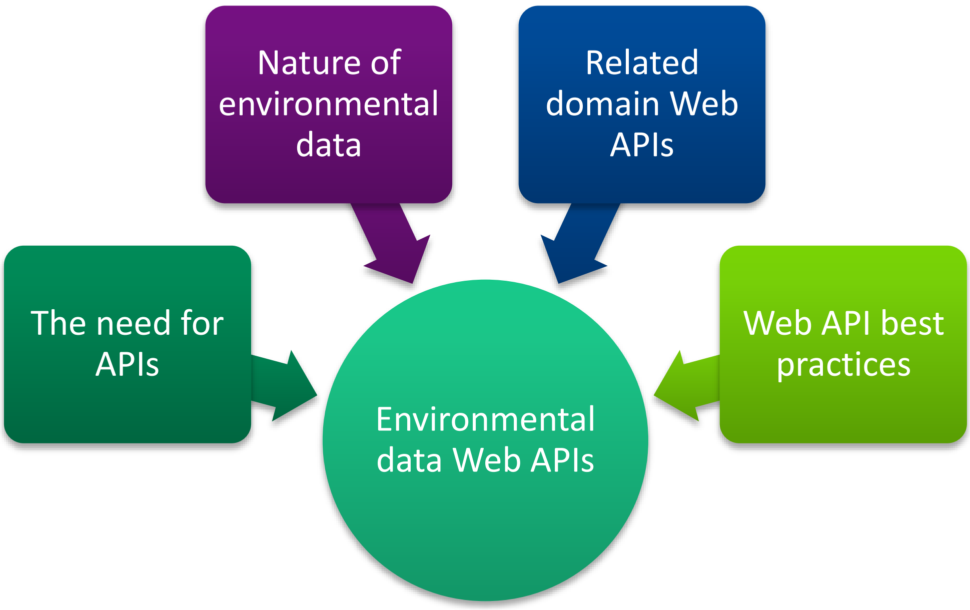


Figure 1 - Method for this report

The structure follows from this: Section 2 defines what Web APIs are and why they are needed; Section 3 provides a brief overview of the challenges associated with environmental data; Section

4 reviews how other environmental agencies are using Web APIs for data delivery; Section 5 provides a summary of associated Web API practices, which are independent of the specific domain of application; and Section 5 concludes the report.

1.1 Scope

While this report originates within a hydrology-specific activity (WIRADA), it takes a broader look at publishing environmental data on the Web. The report does not target any specific areas, but attempts to look across a range of areas within environmental monitoring and data acquisition.

This report does not review existing standards of the Open Geospatial Consortium (OGC), as these have been covered in other activities (Swain et al., 2015)(Percivall, 2010) and within the OGC test beds. We do touch on a number of relevant activities related to the OGC (e.g. the GeoServices API). Additionally the report does not investigate vocabulary (e.g. code lists) management and delivery services. We also do no provide an in-depth review of Linked Data services (Heath and Bizer, 2011), but acknowledge there is influence occurring between Linked Data and Web APIs (see following section).

2 The need for Web APIs

The term 'Web API' has no widely agreed formal definition. It generally refers to a server-side Application Programming Interface (API). Such APIs provide software interfaces for message systems based on a request-response pattern.

For the purposes of this report, the following statements apply to the term Web API:

- Mostly associated with the REpresentational State Transfer (REST) architectural style;
- Mostly use JSON encodings;
- The focus tends to be on *web and mobile developers* as consumers.
- There tends to be less focus on service-to-service interaction, and more on serving web applications directly.

While Web APIs can be described using the more traditional term 'Web Services', these are often associated with the suite of World Wide Web Consortium (W3C) Web Services standards. In this sense, 'Web Services' typically use some or all of the following standards:

- Simple Object Access Protocol (SOAP);
- Web Service Definition Language (WSDL);
- Universal Description Discovery and Integration (UDDI);
- XML encodings.

The two terms also usually reflect a different user base. Table 1 provides a summary of service types and related technologies, their typical users and origin. While this report focuses on the 'Web 2.0' style of APIs, there appears to be increasing convergence and cross-pollination of ideas between Linked Data and Web APIs.

Table 1 - Service type categorisation

Primary usage domain/origin	Technologies utilised	Core attributes
Enterprise computing	XML (XSD, XSLT), W3C Web Services, SOAP, enterprise service bus, relational databases.	• Supports integration of different business platforms • Testable information models with conformance checks • Quality of service mechanisms directly supported • Detailed metadata • Strict validation of content

Web 2.0	Javascript (client + server-side), HTTP, JSON, document and in-memory databases	• Supports development of dynamic web applications • Loose validation of content • Simplified information models
Research	Semantic Web, Linked Data, RDF	• Interlinked data across the web • Web scale data integration • Distributed, potentially unstable, infrastructure • Open, extensible, content models • Validation is an ongoing challenge[1]

2.1 Why provide APIs?

Web APIs are becoming crucial to any company operating on the Internet. The growth in number of APIs provides some indication of how important they are becoming. Figure 2 shows the growth in APIs registered on http://www.programmableweb.com/.

[1] http://www.w3.org/2014/data-shapes/charter

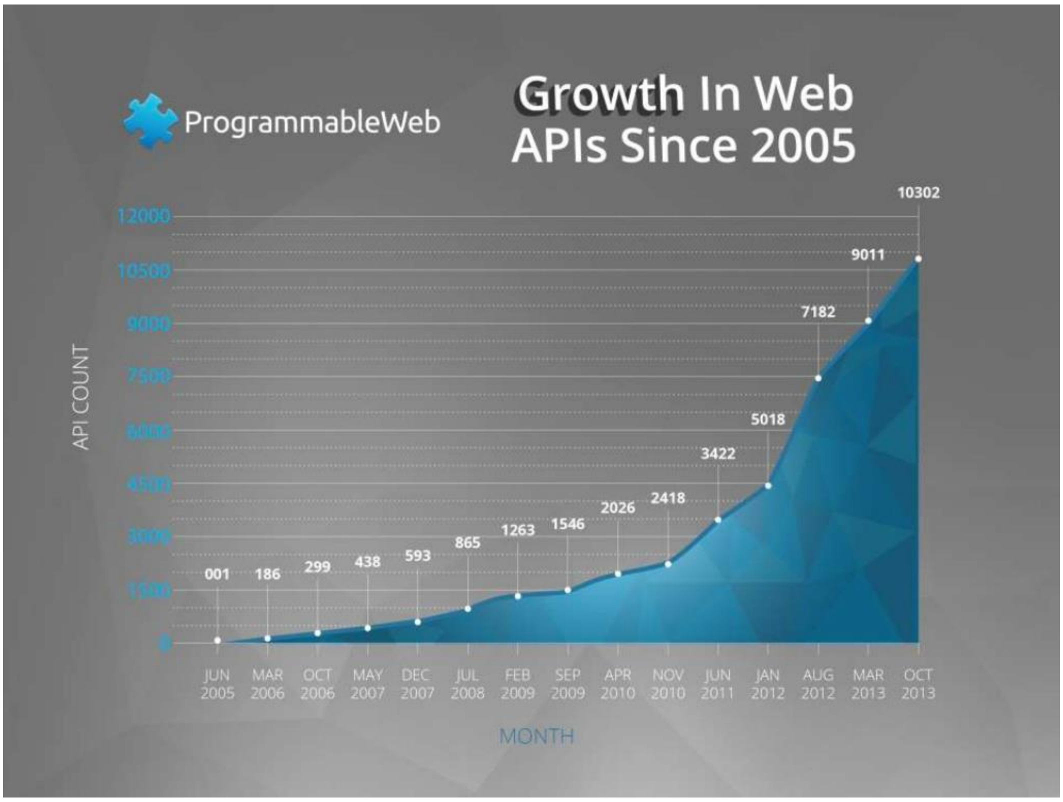


Figure 2 - Growth in APIs (image supplied with permission from ProgrammableWeb.com)

Companies are using APIs as platforms for building their own services, as well as building a community around their services. It is becoming an engagement model with their customers that create positive feedback into their businesses.

When a company provides an API, it is providing a service to the developer community. This involves making data available in some form. Designing Web APIs should be about minimising friction for developers. This involves carefully balancing complexity with functionality. If a balance is achieved then developers will use the API, build applications, tools, and write documentation. An ecosystem begins to form. This greatly benefits the service provider as such an ecosystem exploits the inherent value in data. It is also begins a feedback loop to services provided by the service provider.

There are many reasons environmental organisations should provide APIs, including:

- **Build mobile apps more quickly**. Having a managed API that provides easy access to data lowers the entry for developers of mobile applications. The application may be developed externally or internally; the API is part of the platform.
- **Customers increasingly want direct access to environmental data**. Environmental data is used in a myriad of ways. And the demand is only increasing. From agricultural services providing decision support to farmers, to aquaculture companies managing fish pens, environmental data is crucial to making informed decisions.
- **Avoid screen-scraping.** Organisations typically provide many web sites with HTML data tables, and graphics showing data as graphs and maps. Developers sometimes try to get the underlying data by 'scraping' the web-page, sometimes even involving inferring data from graphics. This is a brittle way to share data. Changes to websites break software;

there is no visibility on how people are accessing the data, or clear methods on controlling their access.

- **Works for external and internal users.** Many companies discover that once a solid API is put in place it becomes equally important for users within the company. This greatly increases the user base and momentum of the API.

2.2 Who are the users of Environmental Web APIs?

The direct users of Web APIs are software developers. But software is not the end game – it's the service provided by the software that creates value. So while developers interact with the API, their requirements differ according to the driver behind writing the software or application. This is illustrated in Figure 3.

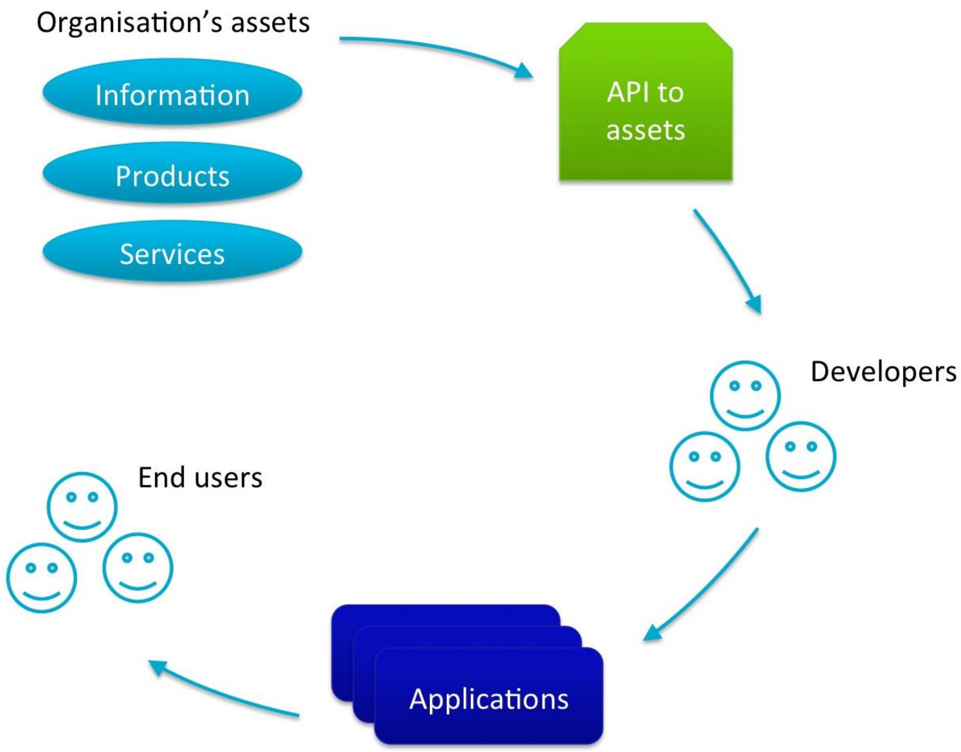


Figure 3 - API value chain

Understanding the user base (end users) gives us an idea of the requirements for specific applications. Some of these requirements we may know already (applications exist), some may have been requested (indicates a need), others we may never predict (novel uses of data). Describing the potential user groups is a useful starting point for understanding requirements.

A report for the Australian Water Resource Information System (AWRIS) (National Water Commission, 2006) includes an initial analysis of user requirements for *hydrological data.* The scope of this report is wider than hydrological data, however the AWRIS report serves as a useful starting point.

2.3 Supporting new data-driven businesses

An important part of providing Web APIs across a range of products is the creation of 'ecosystems' for business to use environmental data. While building an application on a single product is useful, there is huge potential in the combination of APIs with other third parties. This can assist in creation of new business models.

The AWRIS user requirements report describes example scenarios that are used to define requirements. We reuse the 'Blue Sky Industries' fictional scenario - reproduced in the breakout box below - to explore how APIs can support different businesses through different project phases.

Blue Sky Industries

Blue Sky Industries are hoping to build a new chemical production plant, and are targeting a number of locations across the globe, including Australia. They are looking to identify regions that would be best suited for this type of plant. Prior to detailed investigation, Blue Sky Industries have developed a set of criteria they hope will provide a short list of **suitable regions**. One of the key criteria is access to significant amounts of **suitable water** for production. Blue Sky Industries need to know where **water is being used** in a similar way, where **water is available** for use, and if the **quality of the available water** is suitable for their requirements.

The bold terms in the scenario description identify data requirements. We further break the scenario into two phases: an analysis phase focused on viability of the plant, and an operational phase once a plant is in place. It would be sensible for the company consider other non-water data sources. Figure 4 describes a fictional way in which different Web APIs could support these two phases. Third parties are introduced to identify the role of data brokers/service providers. These providers may offer analytical services to Blue Sky to support the project phases. Blue Sky or the originating organisation may even provide this role. A scenario like this also provides interesting options for cost structures for data, analytics and support.

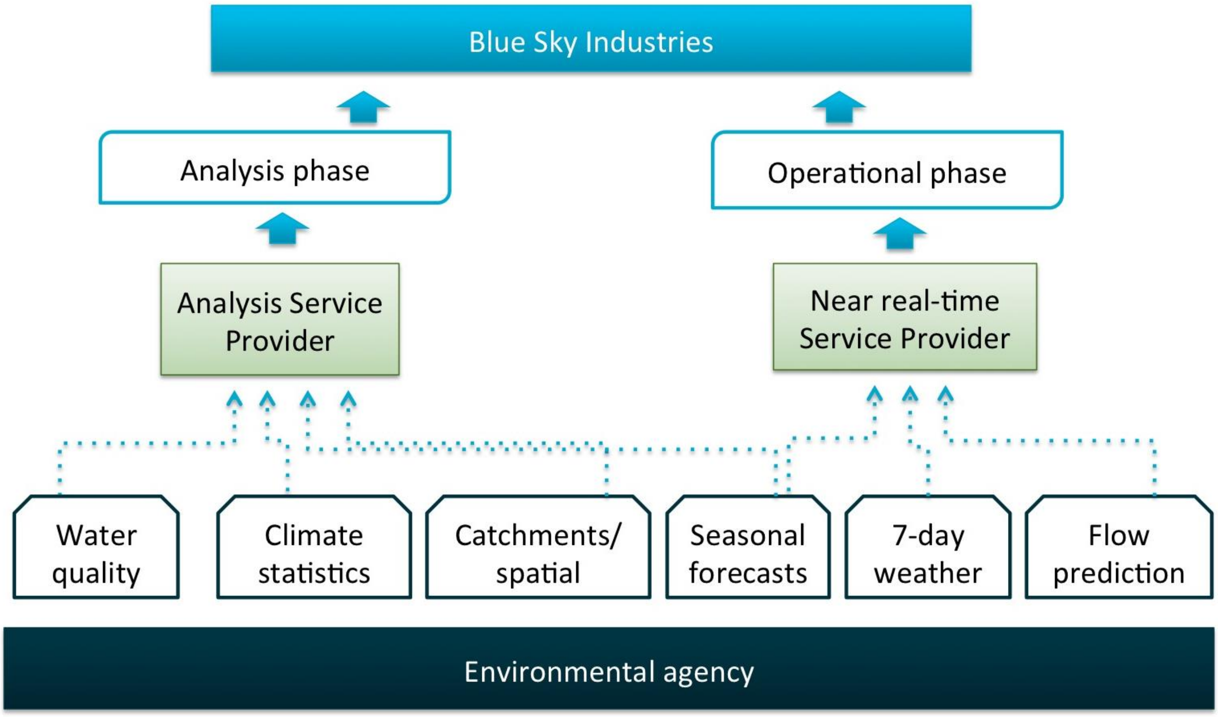


Figure 4 - Scenario data requirements across two phases

The following environmental data products have potential for use in such a scenario:

Analysis phase

- **Seasonal forecasts** – what are the mid-term expectations around water availability?
- **Climate statistics** – how can previous conditions inform analysis of the plant viability?
- **Water quality –** does the water quality match plant operation requirements?
- **Catchments/spatial regions –** analysis of suitable regions would require an understanding of catchment areas and general spatial features (nearby assets, hydrological features etc.).

Operational phase

- **Short-term weather –** monitoring for severe weather events;
- **Flow predictions –** manage quantities for intakes;
- **Seasonal forecasts –** medium-term production forecasts based on access to water;
- **Water quality –** ensure adequate water quality is being maintained.

These are fictional usages of data, but most already exist in industry in some form. The scenario is used to indicate the new ways in which organisations are becoming more data-driven. Increasing the ways and ease in which organisations can access data is an important catalyst in the creation of these businesses.

2.4 What are the challenges for environmental data Web APIs?

Conway's Law[2] afflicts many organisations:

> "*organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations*"

This is typical for organisations with long histories - there are systems that are isolated and not connected to other relevant parts. The result is data access over the Internet is segregated: there are multiple ways to get to similar data, there is overlap between reporting services, and it's difficult for developers to find where best to access data.

Additionally, environmental agencies deal in data that are inherently complex. Providing a single, simple Web API to access data is not trivial. We cannot simply follow the 'develop a REST service for your products' tutorial. Environmental data are diverse and dynamic. From a simple analysis of a typical environmental agency's data products and services, we identified 63 distinct Internet offerings. These range from observations to forecasts, weather to hydrology, and point data to gridded data.

This leads to the primary question explored in this report: **how do organisations provide consistent Web APIs across a range of environmental data products?**

[2] https://en.wikipedia.org/wiki/Conway%27s_law

3 The nature of environmental data

Environmental data suffers from all the Big Data challenges, known as the ‘Vs’ of Big Data. This provides a useful frame to describe the challenges environmental agencies face in handling its data. Figure 5 provides some examples of how environmental data relate to each of the ‘V’ challenges, which are discussed below.

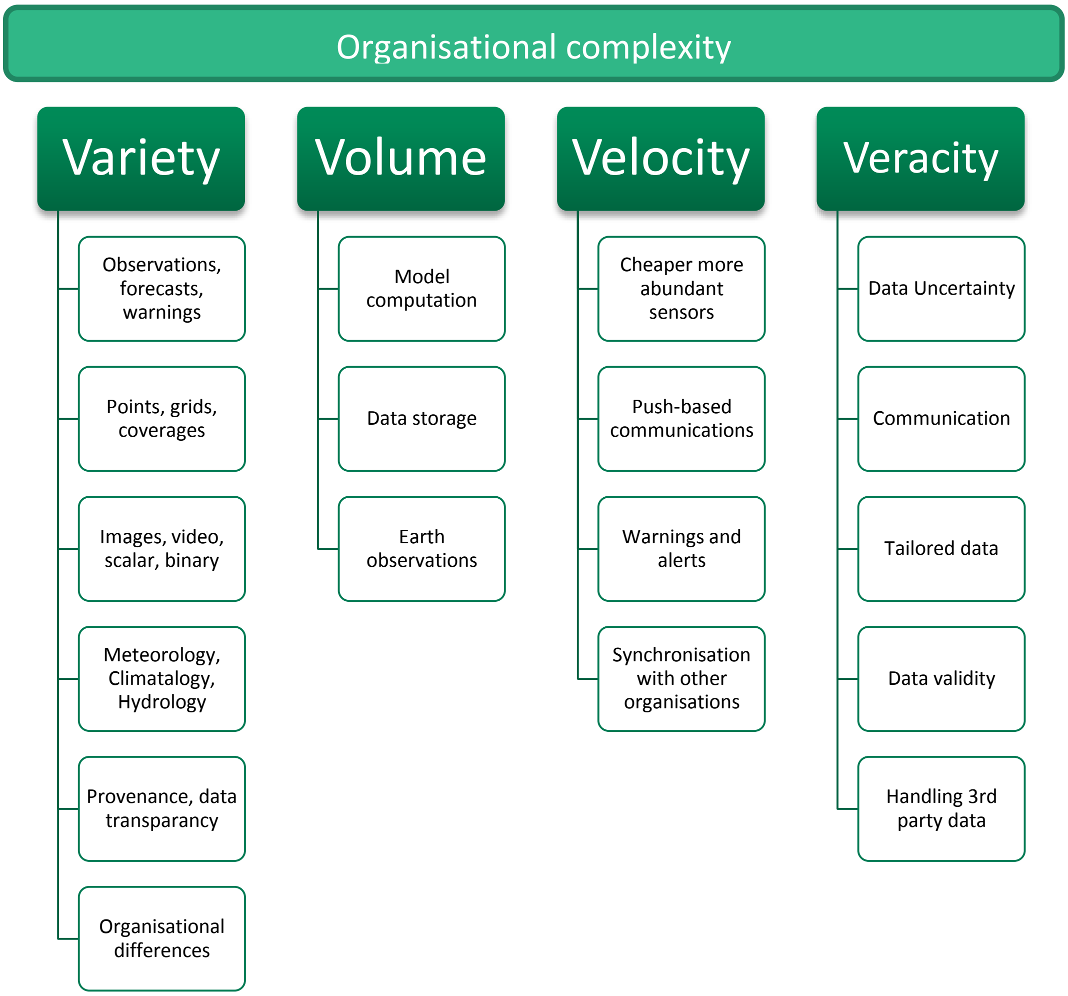


Figure 5 - Complexities in environmental data

Variety – There is huge variety in the data managed by environmental agencies. Issues requiring consideration include:

- Temporal nature: observations and forecasts;
- Structure: point and gridded;
- Multi-modal: images, video, spectra, point clouds etc.
- Domain specific: meteorological, hydrological, climatological, oceanography, etc.

- Organisational: use of specific codes, vocabularies, methods, semantics and systems.
- Provenance: tracking data lineage and post-processing to provide transparency to data users.

Volume – Environmental models can generate huge amounts of data. Models are also coupled with other models and data sets that are equally large. For example, satellite data that are inputs to global circulation models that are inputs to regional meteorological models.

Velocity – Data moves quicker than ever. Sensor communications are switching from traditional, pull-based mechanisms, to near real-time push-based communications. The temporal and spatial resolution of models is increasing as compute power is able to handle higher loads. Interdependencies of data moving at different rates create a complex data management environment. The speed at which people, and machines, demand data has also increased. User's expectations around the timeliness of data have increased dramatically over the last 10 years.

Veracity – Introduced later in the definition of Big Data challenges, Veracity – or uncertainty of data – is a big issue for environmental data. From issues of positioning, and calibration of sensors, to the inherent uncertainty in any forecast, environmental agencies are in the business of handling uncertainty. Communication of quality and uncertainty to users of data is a huge challenge, and one that has a large social component.

Overlaid across these are the organisational and social complexities that exist in any organisation. Differing data management approaches, systems, language and culture all contribute to hampering approaches to cohesive data management and publication.

4 Related domain APIs

There is an enormous number of APIs on the Web. ProgrammableWeb[3] lists over 14,000 available APIs in their catalog of APIs. These include APIs for sports, social, weather, finance, design and so on. Many of these APIs are location or context specific – for example providing an API to Norway's library catalog, or an event search API for Europe.

There are environmental APIs within this catalog. Many are registrations of federal/national data services, such as NOAA, USGS for the US. There has been enormous growth in weather prediction services, and 182 APIs listed are associated with weather data. In turn, this has spawned services that aggregate and compare forecast quality across services[4].

Reviewing all these APIs was not feasible for this analysis. In this section we cover *some* of the main environmental data APIs we have been able to identify. The focus is on environmental agencies operating meteorological, hydrological, oceanographic, and related monitoring and forecasting programs. We seek to categorise the key characteristics of these APIs with the aim of describing the current practices, and direction that Web APIs are heading. In Appendix A we provide a summary of the APIs and their key characteristics.

4.1 National Oceanic and Atmospheric Administration (NOAA)

The US National Oceanic and Atmospheric Administration (NOAA) contains six organisations that cover the following areas:

- Environmental Satellite, Data, and Information Service
- Marine Fisheries Service
- Ocean Service
- Weather Service
- Marine & Aviation Operations
- Oceanic and Atmospheric Research

The primary role for data access falls within the Environmental Satellite, Data, and Information Service. It previously hosted three data centres that are now being brought together into the National Centers for Environmental Information (NCEI). This includes the (National) Climatic Data Center, the Geophysical Data Center, and the Oceanographic Data Center.

There are attempts at bringing data and services together. The primary example of this is https://data.noaa.gov/, which is a CKAN-driven (see section 4.11) data discovery and access portal.

---

[3] http://ww.programmableweb.com

[4] http://www.weatherapi.net/compare-forecasts/

This is still under development. However, it acts as a catalog and data upload service rather than a specific data access point – the URLs for data access are submitted as part of registering a dataset.

The NCEI covers many categories of data as shown in Figure 6.

Figure 6 - NOAA NCEI data categories

Most service links lead to a specific data centre's web site and associated web service, which, understandably, shows integration is a long-term goal. There is a large number of APIs available; a selection from some of the categories are shown in Table 2. The Climate API v2 provides a RESTful API and is summarised in the following section.

Table 2 - Selection of NOAA APIs

ONLINE APIS	DESCRIPTION	FORMAT(S)
Climate Data API v2	Search and access datasets and station and/or location data.	JSON
Climate API Legacy	Station time-series data.	XML, CSV
Climate REST Services	Station network locations and climatological GIS products.	JSON, XML
Catalog Services (CSW)	A catalog service that conforms to the HTTP protocol binding of the OpenGIS Catalog Service ISO Metadata Application Profile specification.	XML
Gridded data service NCEI TDS	Satellite/Radar/Model Data.	NetCDF, XML, CSV

Numerical model gridded service: **NOMADS TDS**	Numerical Model Data.	NetCDF, XML, CSV
Severe Weather Data Inventory	NEXRAD Storm Cell Attributes, Storm Reports.	XML, CSV, KMZ, Shapefile
Historic Observing Metadata Repository Web Services	Integrated station history database providing in situ metadata.	JSON

4.1.1 Climate Data API v2.0

The basic resources of the NOAA Climate API are summarised in Figure 7.

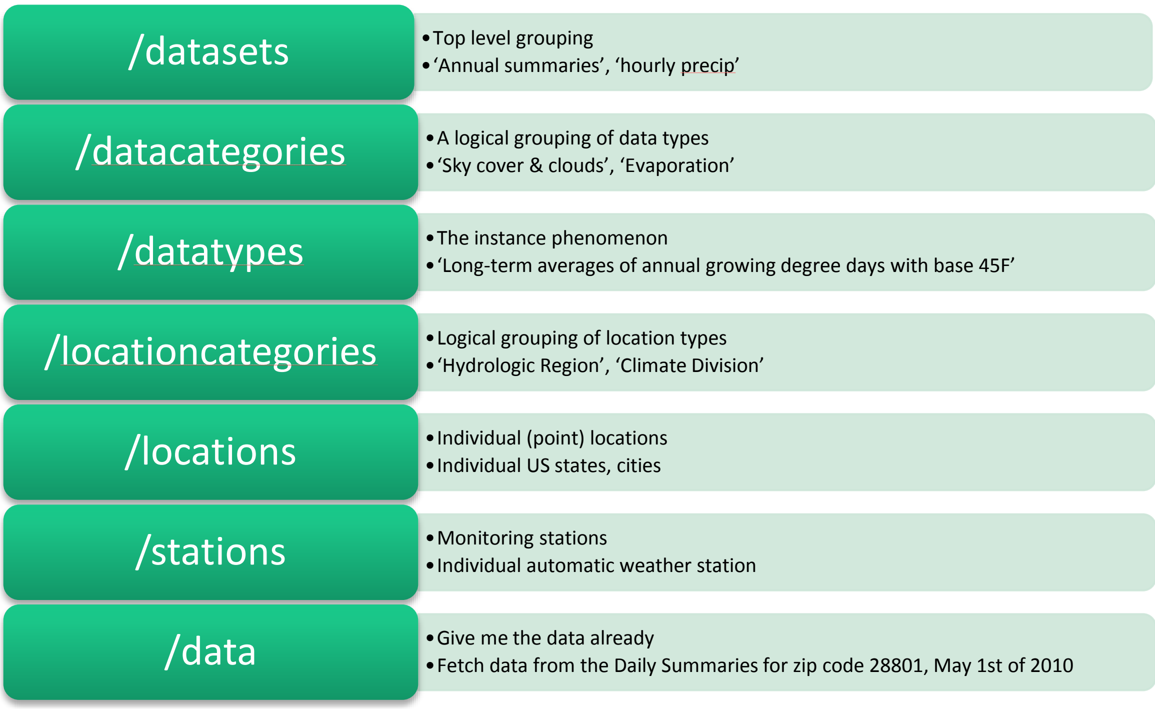


Figure 7 - Climate data online RESTful API overview

The API provides a landing page (Code listing 1) that describes the available resource end points, available parameters and whether they are required. This is good practice and generally recommended within RESTful APIs.

Code listing 1 - Landing page for Climate API v2


```
[
  {
    "path": "v2/lcd/availablereporttypes",
    "method": "GET",
    "parameters": [
      {
        "name": "stationid",
        "required": false
      },
      {
        "name": "locationid",
        "required": false
      },
      {
        "name": "startdate",
        "required": true
      },
      {
        "name": "enddate",
        "required": true
      }
    ]
  },
```


The resources in this API do not make use of embedded links within the JSON response documents, which would allow automated traversal of the API. It also requires the client understand the relationship between relevant resources. For example to find precipitation data, one must use the 'datatype' resource to retrieve a list of all the codes; see example HTTP request and JSON response in Listing 2.

Listing 2 - example datatype listing

```
GET http://www.ncdc.noaa.gov/cdo-web/api/v2/datatypes?limit=1000 (auth header required)
```


```
{
  "metadata": {
    "resultset": {
      "offset": 1,
      "count": 1461,
      "limit": 1000
    }
  },
  "results": [
    {
      "mindate": "1994-03-19",
      "maxdate": "1996-05-28",
      "name": "Average cloudiness midnight to midnight from 30-second ceilometer data (percent)",
      "datacoverage": 1,
      "id": "ACMC"
    },
    {
      "mindate": "1965-01-01",
      "maxdate": "2005-12-31",
      "name": "Average cloudiness midnight to midnight from manual observations (percent)",
      "datacoverage": 1,
      "id": "ACMH"
    },
    {
      "mindate": "1994-02-01",
      "maxdate": "1996-05-28",
      "name": "Average cloudiness sunrise to sunset from 30-second ceilometer data (percent)",
      "datacoverage": 1,
      "id": "ACSC"
    },
```

There are many (1461) data types, some with very specific definitions such as 'Long-term averages of number of days during December-February with precipitation >= 0.50 inches' and '50th percentiles of daily nonzero snow depth for 29-day windows centred on each day of the year'. There are ways of filtering, e.g. return a list of stations that support a specific datacategoryid (see Listing 3), but this requires the code identifier for the category ID.

Handling code lists and definitions is a common problem in Web APIs. There are some approaches typically used that can help: the service may provide embedded links to allow direct navigation to the definition, or provide a definition object with basic properties, plus links for further details. These approaches can assist by reducing the amount of manual lookups required from clients. There is increasing interest in the use of vocabulary services to assist in the publication and management of vocabularies (Cox et al., 2014; Schandl and Blumauer, 2010; Yu et al., 2011). Review of such services is out of scope for this report. However, it is recommended that such services be investigated when deploying Web APIs that use code lists and/or structured vocabularies.

Listing 3 – query stations that support specific datatype

```
GET http://www.ncdc.noaa.gov/cdo-web/api/v2/stations?datatypeid=ANN-PRCP-AVGNDS-GE001HI (auth
header required)
```


```
{
  "metadata": {
    "resultset": {
      "offset": 1,
      "count": 7480,
      "limit": 25
    }
  },
  "results": [
    {
      "elevation": 408.4,
      "mindate": "1980-01-01",
      "maxdate": "2012-08-31",
      "latitude": -14.31667,
      "name": "AASUFOU, US",
      "datacoverage": 1,
      "id": "GHCND:AQC00914000",
      "elevationUnit": "METERS",
      "longitude": -170.76667
    },
    {
      "elevation": 3.7,
      "mindate": "1945-08-01",
      "maxdate": "2015-12-30",
      "latitude": -14.33056,
      "name": "PAGO PAGO WEATHER SERVICE OFFICE AIRPORT, US",
      "datacoverage": 1,
      "id": "GHCND:AQW00061705",
      "elevationUnit": "METERS",
      "longitude": -170.71361
    },
```


In addition to the datatype ID, a dataset ID is required, in the example case this is requested as shown in Listing 4.

Listing 4 - requesting a dataset for a specific datatype

```
GET http://www.ncdc.noaa.gov/cdo-web/api/v2/datasets?datatypeid=ANN-PRCP-AVGNDS-GE001HI (auth
header required)
```

```
{
  "metadata": {
    "resultset": {
      "offset": 1,
      "count": 1,
      "limit": 25
    }
  },
  "results": [
    {
      "uid": "gov.noaa.ncdc:C00821",
      "mindate": "2010-01-01",
      "maxdate": "2010-01-01",
      "name": "Normals Annual/Seasonal",
      "datacoverage": 1,
      "id": "NORMAL_ANN"
    }
  ]
}
```


This then allows a query to be made against the 'data' resource itself, as shown in Listing 5.

Listing 5 - requesting data for multiple stations of a particular datatype, within a time window

```
GET http://www.ncdc.noaa.gov/cdo-web/api/v2/data?datatypeid=ANN-PRCP-AVGNDS-
GE001HI&startdate=2010-01-01&enddate=2010-02-06&datasetid=NORMAL_ANN (auth header required)
```



```
{
  "metadata": {
    "resultset": {
      "offset": 1,
      "count": 7484,
      "limit": 25
    }
  },
  "results": [
    {
      "date": "2010-01-01T00:00:00",
      "datatype": "ANN-PRCP-AVGNDS-GE001HI",
      "station": "GHCND:AQC00914000",
      "attributes": "P",
      "value": 2275
    },
    {
      "date": "2010-01-01T00:00:00",
      "datatype": "ANN-PRCP-AVGNDS-GE001HI",
      "station": "GHCND:AQW00061705",
      "attributes": "C",
      "value": 2495
    },
    continues for 7848 points
```


The use of disjoint resources allows query flexibility, but also requires the client to understand the resource model and which queries are logical. This is done through a combination of reading documentation and making example queries.

Key characteristics:

- Incomplete metadata.
- Uses code lists without much description of code specifics.
- No use of links/hypermedia.

- Implicit CRS for lat-lon pairs.

4.2 UK Met Office – DataPoint

The UK Met Office runs a beta RESTful API called DataPoint[5], which provides access to observations and forecasts, using JSON, XML and imagery and text formats. The API generalises along three concepts: the product, data type and content type. This is summarised in Figure 8.

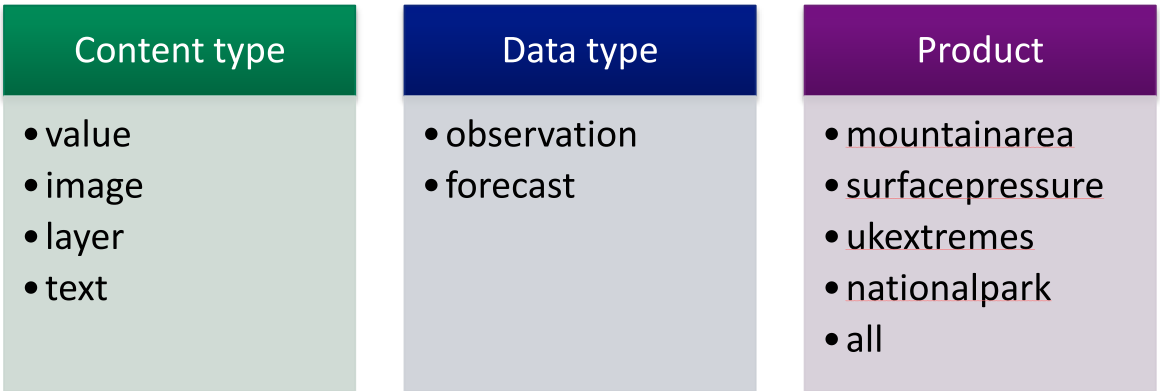


Figure 8 - DataPoint API overview

The service does not provide links and thus lacks the hypermedia component of RESTful APIs. The JSON representation appears to be a direct translation from XML, and thus mimics XML's structure. This in itself is not a problem, however in some cases it maps attributes using an ampersand character (e.g. "@name"), which is OK for plain JSON but may conflict in Linked Data JSON (JSON-LD) where ampersands are used for semantic mark-up.

The capabilities requests provide a summary of available datasets within a particular product; see Listing 6 for an example listing of forecast layers. An example site listing response is shown in Listing 7. There is no paging applied to this response, which results in all sites being returned (1.4MB JSON response). Listing 8 shows an example of a regional text forecast encoded in JSON.

Listing 6 - example capabilities for forecast layers (modified for formatting)

```
GET
http://datapoint.metoffice.gov.uk/public/data/layer/wxfcs/all/json/capabilities?res=hourly&key=<K
EY>
```

[5] http://www.metoffice.gov.uk/datapoint

```
{
  "Layers": {
    "@type": "Forecast",
    "BaseUrl": {
      "@forServiceTimeFormat": "Timesteps",
      "$":
"http://datapoint.metoffice.gov.uk/public/data/layer/wxfcs/{LayerName}/{ImageFormat}?RUN={DefaultTime}Z
&FORECAST={Timestep}&key={key}"
    },
    "Layer": [
      {
        "@displayName": "Rainfall",
        "Service": {
          "@name": "UKPPBEST",
          "LayerName": "Precipitation_Rate",
          "ImageFormat": "png",
          "Timesteps": {
            "@defaultTime": "2016-01-04T21:00:00",
            "Timestep": [
              0,3,6,9,12,15,18,21,24,27,30,33,36
            ]
          }
        }
      },
      {
        "@displayName": "Cloud",
        "Service": {
          "@name": "UKPPBEST",
          "LayerName": "Total_Cloud_Cover",
          "ImageFormat": "png",
          "Timesteps": {
            "@defaultTime": "2016-01-04T21:00:00",
            "Timestep": [
              0,3,6,9,12,15,18,21,24,27,30,33,36
            ]
          }
        }
      },
      {
        "@displayName": "CloudAndRain",
        "Service": {
          "@name": "UKPPBEST",
          "LayerName": "Total_Cloud_Cover_Precip_Rate_Overlaid",
          "ImageFormat": "png",
          "Timesteps": {
            "@defaultTime": "2016-01-04T21:00:00",
            "Timestep": [
             0,3,6,9,12,15,18,21,24,27,30,33,36
            ]
          }
        }
      },
      {
        "@displayName": "Temperature",
        "Service": {
          "@name": "UKPPBEST",
          "LayerName": "Temperature",
          "ImageFormat": "png",
          "Timesteps": {
            "@defaultTime": "2016-01-04T21:00:00",
            "Timestep": [
             0,3,6,9,12,15,18,21,24,27,30,33,36
            ]
          }
        }
      }
        <snipped>
     }
    ]
  }
```

Listing 7 – Example location listing in JSON

GET `http://datapoint.metoffice.gov.uk/public/data/val/wxfcs/all/json/sitelist?key=<KEY>`


```
{
  "Locations": {
    "Location": [
      {
        "elevation": "933.0",
        "id": "3072",
        "latitude": "56.879",
        "longitude": "-3.42",
        "name": "Cairnwell",
        "nationalPark": "Cairngorms National Park",
        "region": "ta",
        "unitaryAuthArea": "Perth and Kinross"
      },
      {
        "elevation": "134.0",
        "id": "3088",
        "latitude": "56.852",
        "longitude": "-2.264",
        "name": "Inverbervie",
        "region": "gr",
        "unitaryAuthArea": "Aberdeenshire"
      },
      {
        "elevation": "4.0",
        "id": "3094",
        "latitude": "57.698",
        "longitude": "-2.121",
        "name": "Rosehearty Samos",
        "region": "gr",
        "unitaryAuthArea": "Aberdeenshire"
      }
```


…continues

Listing 8 - example text regional forecast, encoded in JSON

GET `http://datapoint.metoffice.gov.uk/public/data/txt/wxfcs/regionalforecast/json/507?key=<KEY>`

```
{
  "RegionalFcst": {
    "createdOn": "2016-01-04T13:11:06",
    "issuedAt": "2016-01-04T16:00:00",
    "regionId": "nw",
    "FcstPeriods": {
      "Period": [
        {
          "id": "day1to2",
          "Paragraph": [
            {
              "title": "Headline:",
              "$": "Rather cloudy with occasional showers,"
            },
            {
              "title": "This Evening and Tonight:",
              "$": "This evening and tonight will be cloudy with showery outbreaks of rain which could
be heavy at times. The showers could merge to give some longer spells of rain, particularly over the
Pennines. Minimum Temperature 5C."
            },
            {
              "title": "Tuesday:",
              "$": "Tuesday will be another rather cloudy day with further showery outbreaks of rain
which could be heavy at times. Maximum Temperature 8C."
            }
          ]
        },
        {
          "id": "day3to5",
          "Paragraph": {
            "title": "Outlook for Wednesday to Friday:",
            "$": "Mainly cloudy but drier on Wednesday. Heavy rain on and strong winds Wednesday night,
clearing to sunshine and heavy showers early on Thursday, with further heavy showers on Friday."
          }
        },
        {
          "id": "day6to15",
          "Paragraph": {
            "title": "UK Outlook for Saturday 9 Jan 2016 to Monday 18 Jan 2016:",
            "$": "The weekend will be unsettled and often windy with showery conditions interspersed
with some more organised spells of rain, and snow on northern hills. The heaviest rain and strongest
winds occurring in the southwest and the northeast, where large rainfall totals and gales are expected.
Next week will continue unsettled, with further showers or longer spells of rain, interspersed by some
brief dry and brighter periods. It will become windy at times, particularly at the start of next week
when severe gales are possible. Temperatures through the period are likely to be near or a little above
average. However, there is the risk of a brief much colder spell at the end of next week, with frost
and snow showers."
          }
        },
        {
          "id": "day16to30",
          "Paragraph": {
            "title": "UK Outlook for Tuesday 19 Jan 2016 to Tuesday 2 Feb 2016:",
            "$": "A generally unsettled spell of weather is expected through this period with winds
mainly coming from the west. Showers or longer spells of rain are expected, interspersed by some dry
and brighter interludes. The rain being heaviest and most persistent in the west where rainfall totals
are likely to be above average. The east should be more sheltered, getting the best of the drier and
brighter interludes, so rainfall totals here may by slightly below average. There are indications of
some longer dry spells later in January, especially in the south. Temperatures through this period
should be near or a little above average for most areas."
          }
        }
      ]
    }
  }
}
```


Listing 9 - example text observations of UK extremes (showing one region only)

GET `http://datapoint.metoffice.gov.uk/public/data/txt/wxobs/ukextremes/json/latest?key=<KEY>`

```
{
  "UkExtremes": {
    "extremeDate": "2016-01-04",
    "issuedAt": "2016-01-04T23:03:09Z",
    "Regions": {
      "Region": [
        {
          "id": "sw",
          "name": "South West England",
          "Extremes": {
            "Extreme": [
              {
                "locId": "03853",
                "locationName": "Yeovilton",
                "type": "HMAXT",
                "uom": "degC",
                "$": "11.0"
              },
              {
                "locId": "03710",
                "locationName": "Liscombe",
                "type": "LMAXT",
                "uom": "degC",
                "$": "6.6"
              },
              {
                "locId": "03647",
                "locationName": "Little Rissington",
                "type": "LMINT",
                "uom": "degC",
                "$": "3.1"
              },
              {
                "locId": "03823",
                "locationName": "Cardinham",
                "type": "HRAIN",
                "uom": "mm",
                "$": "23.2"
              },
              {
                "locId": "03853",
                "locationName": "Yeovilton",
                "type": "HSUN",
                "uom": "hours",
                "$": "2.2"
              }
            ]
          }
        }
```


Key characteristics:

- Observations and forecasts supported
- Capabilities for discovery of available time slices
- Text-based documentation
- No linking
- JSON or XML responses available
- Supports imagery and descriptive text outputs
- Implicit CRS for lat/lon values

4.3 Forecast.io

Forecast.io provides an API[6] to access its aggregated forecast data. This is used to drive its mobile application DarkSky[7]. The API it provides is a simple RESTful service that provides JSON encoded data of:

- Current conditions
- Minute-by-minute forecasts out to 1 hour (where available)
- Hour-by-hour forecasts out to 48 hours
- Day-by-day forecasts out to 7 days.

The API is divided into two main calls: current forecast for next 7 days at a specific location (lat, lon), and observed or forecast conditions for a specific location (lat, lon) and time (60 years into the past to 10 years into the future).

The response structure is simple: arrays of data blocks containing both hourly and daily data points. Each data point contains a UNIX time with associated forecast values. See listing below for example data point structure.

```
{
        "time": 1448283600,
        "summary": "Mostly cloudy throughout the day.",
        "icon": "partly-cloudy-day",
        "sunriseTime": 1448303752,
        "sunsetTime": 1448356686,
        "moonPhase": 0.43,
        "precipIntensity": 0.0152,
        "precipIntensityMax": 0.0381,
        "precipIntensityMaxTime": 1448344800,
        "precipProbability": 0.03,
        "precipType": "rain",
        "temperatureMin": 11.63,
        "temperatureMinTime": 1448298000,
        "temperatureMax": 17.55,
        "temperatureMaxTime": 1448334000,
        "apparentTemperatureMin": 11.63,
        "apparentTemperatureMinTime": 1448298000,
        "apparentTemperatureMax": 17.55,
        "apparentTemperatureMaxTime": 1448334000,
        "dewPoint": 8.92,
        "humidity": 0.71,
        "windSpeed": 6.49,
        "windBearing": 283,
        "cloudCover": 0.61,
        "pressure": 1008.45,
        "ozone": 305.7
      }
```

Key characteristics:

[6] https://developer.forecast.io/docs/v2

[7] http://darkskyapp.com/

- Use of UNIX time in responses. ISO8601 are available for time-based requests.
- All values are JSON numbers with no metadata. All metadata resides in the documentation. E.g. 'pressure: A numerical value representing the sea-level air pressure in millibars.'
- Units are grouped using a 'unit' query parameter. This makes it possible to request SI units using a query parameter and all units are translated. There is an 'auto' request, which will lookup the units associated with geolocation specified.
- Latitude and longitude are assumed to be in a (non-specified) hardcoded CRS.

4.4 USGS Water Data Services

The USGS offer a range of Web APIs[8] for water data including a site service, daily values service, groundwater service, water quality service, instantaneous values service, and a statistics service. These services tend to follow a key-value pair style of API. The documentation is not generated from the code interfaces, but very precise in its definition of available parameters and queries. Test tools are available for all services, which provide an HTML form-based tool for creating queries to the services.

The services support multiple output formats (XML, JSON, CSV), along with support for some data transfer standards, e.g. WaterML2.0. The services are consistent in their presentation and use of common practices.

The JSON response format appears to be a mapping of the WaterML1.1 XML structure; see Listing 10 for an example response from the instantaneous value service. WaterML1.1 is non-OGC version of WaterML that didn't make use of existing standards, such as Observations & Measurements.

Listing 10 - example instantaneous value request in JSON

```
GET
http://waterservices.usgs.gov/nwis/iv/?format=json,1.1&sites=01646500&parameterCd=00060,00065
```

[8] http://waterservices.usgs.gov/rest/

```
{
        "name": "ns1:timeSeriesResponseType",
        "declaredType": "org.cuahsi.waterml.TimeSeriesResponseType",
        "scope": "javax.xml.bind.JAXBElement$GlobalScope",
        "value": {
                "timeSeries": [{
                        "sourceInfo": {
                                "siteName": "POTOMAC RIVER NEAR WASH, DC LITTLE FALLS PUMP STA",
                                "siteCode": [{
                                        "value": "01646500",
                                        "network": "NWIS",
                                        "siteID": null,
                                        "agencyCode": "USGS",
                                        "agencyName": null,
                                        "default": null
                                }],
                                "timeZoneInfo": {
                                        "defaultTimeZone": {
                                                "zoneOffset": "-05:00",
                                                "zoneAbbreviation": "EST"
                                        },
                                        "daylightSavingsTimeZone": {
                                                "zoneOffset": "-04:00",
                                                "zoneAbbreviation": "EDT"
                                        },
                                        "siteUsesDaylightSavingsTime": false
                                },
                                "geoLocation": {
                                        "geogLocation": {
                                                "srs": "EPSG:4326",
                                                "latitude": 38.94977778,
                                                "longitude": -77.12763889
                                        },
                                        "localSiteXY": []
                                },
                                "elevationM": null,
                                "verticalDatum": null,
                                "note": [],
                                "extension": null,
                                "altname": null,
                                "siteType": [],
                        },
                        "variable": {
                                "variableCode": [{
                                        "value": "00060",
                                        "network": "NWIS",
                                        "vocabulary": "NWIS:UnitValues",
                                        "variableID": 45807197,
                                        "default": true
                                }],
                                "variableName": "Streamflow, ft&#179;/s",
                                "variableDescription": "Discharge, cubic feet per second",
                                "valueType": "Derived Value",
                                "dataType": null,
                                "generalCategory": null,
                                "sampleMedium": null,
                                "unit": {
                                        "unitName": null,
                                        "unitDescription": null,
                                        "unitType": null,
                                        "unitAbbreviation": "ft3/s",
                                        "unitCode": null,
                                        "unitID": null
                                },
```


continues…

The USGS has also recently introduced a statistics web service and is similar in design (key-value pairs). It provides daily, monthly and annual statistics for continuous monitoring stations. It currently only supports CSV responses, with JSON and XML planned for future work.

Key characteristics:

- Detailed documentation.
- Multiple services, with general consistent behaviour and practices (across the water data services).
- Support for standard encodings.
- Key-value pair style API.
- Separation of services by temporal dimension (e.g. instantaneous and daily) and sub-domain (surface water, groundwater, and water quality).
- No hypermedia/links in responses.
- Explicit CRS.

4.5 Atlas of Living Australia

The Atlas of Living (ALA) has a comprehensive API for accessing species data, including occurrences, taxonomies, spatial data (distributions, points of interest etc.) and plots. It provides both output and input options to support crowd-sourced data.

Service type	Description	Example queries
Species Profile	Taxonomic name, concept lookups, autocomplete services.	Search for all 'Acacia' species
Occurrence	Specimen & observation data searching.	Search for the records for the genus Macropus returning taxonomic, geospatial, temporal information for occurrences.
Geospatial	Get/put spatial layers. Get/put points of interest. Get/put geometries. Including intersection services and gazetteer information.	Download a shape object as GeoJSON using its primary ID.
Mapping	Creating maps with WMS services, static heat maps.	Generates a static density heat map in PNG format.
Endemism	Services for reports on endemism for an area.	List endemic species within an area.

Collections	Collections metadata including taxonomic scope, attribution.	List details of Australian National Insect Collection.
Data Resources	Data resource metadata including taxonomic scope, attribution.	Get metadata associated with a species list.
Data Providers	Data provider metadata including taxonomic scope, attribution.	Get data from the CSIRO National Fish Collection.
Institution	Institution metadata including taxonomic scope, attribution.	Get details of the Tasmanian Museum and Art Gallery.
Crowd sourcing	Crowd sourced data – contributions, validation, automated harvesting, stats for user contributions.	Retrieve number of user contributions by month.
General data	Dashboard data feeds, data licences, region lists etc.	Get list of all licences used by ALA.
Taxonomy	Taxonomic services. E.g. name lookups, higher order lookups.	Higher taxa for Macropus rufus
.Distributions	Services for accessing species distribution polygons	Display expert distribution map for Monacanthus chinensis
Annotations	Annotations, assertions for data.	List all assertion codes used.
Scatterplots	Services for the generation of scatterplots for occurrence data.	Example for Macropus Rufus.
Species lists	Web services for storage, creation and querying of lists.	Get a species list for 'Red Kangaroo'.

Listing 11 - example listing for an institution

```
GET http://collections.ala.org.au/ws/institution/in25
```

```
{
	"name": "Tasmanian Museum and Art Gallery",
	"acronym": "TMAG",
	"uid": "in25",
	"guid": "urn:lsid:biocol.org:col:34362",
	"address": {
		"street": "Dunn Place",
		"city": "Hobart",
		"state": "Tasmania",
		"postcode": "7000",
		"country": "Australia",
		"postBox": null
	},
	"phone": "(03) 6165 7000",
	"email": "herbarium@tmag.tas.gov.au",
	"pubDescription": "Primarily, the collections are of the Tasmanian fauna.\r\n\r\nThe Tasmanian Museum and Art Gallery is making a range of species information and data from its natural history collections available through the Atlas of Living Australia.",
	"techDescription": "Some primary types held and stored separately.",
	"focus": null,
	"latitude": -42.8817500000,
	"longitude": 147.3321520000,
	"state": "Tasmania",
	"websiteUrl": "http://www.tmag.tas.gov.au/",
	"alaPublicUrl": "http://collections.ala.org.au/public/show/in25",
	"imageRef": {
		"filename": "Astacopsis_gouldi_300px.jpg",
		"caption": "Tasmanian Giant Freshwater Lobster",
		"copyright": null,
		"attribution": "Photo courtesy: Tasmanian Museum and Art Gallery",
		"uri": "http://collections.ala.org.au/data/institution/Astacopsis_gouldi_300px.jpg"
	},
	"logoRef": {
		"filename": "TasGov_HorC.png",
		"caption": "Tasmanian Govt, TMAG logo",
		"copyright": null,
		"attribution": null,
		"uri": "http://collections.ala.org.au/data/institution/TasGov_HorC.png"
	},
	"networkMembership": [{
		"name": "Council of Heads of Australian Faunal Collections",
		"acronym": "CHAFC",
		"logo": "http://collections.ala.org.au/data/network/CHAFC_sm.jpg"
	}, {
		"name": "Council of Heads of Australian Entomological Collections",
		"acronym": "CHAEC",
		"logo": "http://collections.ala.org.au/data/network/chaec-logo.png"
	}],
	"hubMembership": [{
		"uid": "dh1",
		"name": "Online Zoological Collections of Australian Museums",
		"uri": "http://collections.ala.org.au/ws/dataHub/dh1"
	}],
	"attributions": [{
		"name": "Biodiversity Collections Index",
		"url": "http://www.biodiversitycollectionsindex.org"
	}, {
		"name": "Council of Heads of Australasian Herbaria",
		"url": "http://www.chah.gov.au/resources/index.html"
	}, {
		"name": "Tasmanian Museum and Art Gallery",
		"url": "http://www.tmag.tas.gov.au/"
	}],
	"dateCreated": "2010-08-31T05:52:27Z",
	"lastUpdated": "2014-10-28T02:05:53Z",
	"userLastModified": "miles.nicholls@csiro.au",
	"institutionType": null,
	"collections": [{
		"name": "Tasmanian Museum and Art Gallery Invertebrate Collection",
		"uri": "http://collections.ala.org.au/ws/collection/co198",
		"uid": "co198"
	}, {
		"name": "Tasmanian Museum and Art Gallery Vertebrate Collection",
		"uri": "http://collections.ala.org.au/ws/collection/co111",
		"uid": "co111"
```

Listing 12 - example spatial object request (state/territory)

```
GET http://spatial.ala.org.au/ws/object/3742602
```


```
{
	"name": "Australian Capital Territory",
	"id": "Australian Capital Territory",
	"description": "Australian Capital Territory, Territory",
	"pid": "3742602",
	"wmsurl":
"http://spatial.ala.org.au/geoserver/wms?service=WMS&version=1.1.0&request=GetMap&layers=ALA:Objects&format=image/png&viewparams=s:3742602",
	"area_km": 2428.0110849586404,
	"bbox": "POLYGON((148.761582 -35.92208,148.761582 -35.1119459999999,150.772781 -35.1119459999999,150.772781 -35.92208,148.761582 -35.92208))",
	"fid": "cl22",
	"fieldname": "Australian States and Territories"
}
```

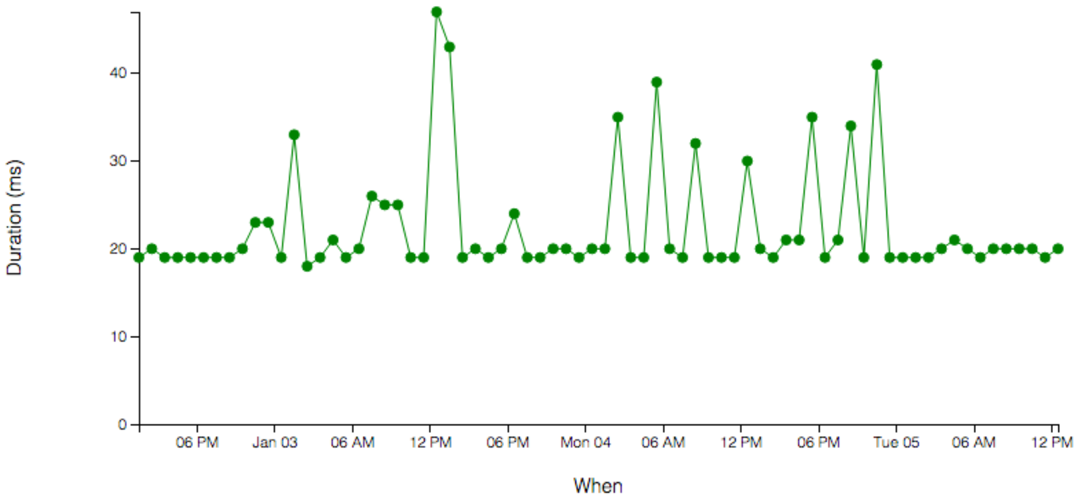

The ALA documentation also provides some interesting features, such as the ability to view historical calls to the public API, including frequency, slow requests, error requests etc. (Figure 9).

The ACT object

Fetch the details for the ACT object.

Responses for the past 72 hours

Response graph



Slow Responses

URL	When	Duration	Response Code	Response Body

Unsuccessful HTTP Responses

URL	When	Duration	Response Code	Response Body
http://spatial.ala.org.au/ws/intersect/poi/pointradius/-35.30821/149.12444/5	Sat Jan 02 13:29:01 EST 2016	29 milliseconds	500	Response Body
http://spatial.ala.org.au/ws/intersect/poi/pointradius/-35.30821/149.12444/5	Sat Jan 02 14:29:01 EST 2016	30 milliseconds	500	Response Body
http://spatial.ala.org.au/ws/intersect/poi/pointradius/-35.30821/149.12444/5	Sat Jan 02 15:29:01 EST 2016	42 milliseconds	500	Response Body
http://spatial.ala.org.au/ws/intersect/poi/pointradius/-35.30821/149.12444/5	Sat Jan 02 16:29:01 EST 2016	42 milliseconds	500	Response Body
http://spatial.ala.org.au/ws/intersect/poi/pointradius/-35.30821/149.12444/5	Sat Jan 02 17:29:01 EST 2016	29 milliseconds	500	Response Body

Slow Responses

URL	When	Duration	Response Code	Response Body
http://biocache.ala.org.au/ws/occurrence/facets?facets=taxon_name&q=data_resource_uid:dr359&flimit=100	Mon Jan 04 19:29:20 EST 2016	14 seconds and 28 milliseconds	200	Response Body
http://biocache.ala.org.au/ws/occurrence/facets?q=data_resource_uid:dr359&facets=taxon_name&flimit=100	Tue Jan 05 11:29:32 EST 2016	15 seconds and 562 milliseconds	200	Response Body

Figure 9 - Example request frequency and call history for specific API call

Key characteristics:

- A lot of use of linked resources.
- Good level of metadata.

- Extensive functionality (maps, parsing data, crowd sourced data, autocomplete services etc.).
- Dynamic documentation with examples.[9]
- Supports insertion, deletion, and updating of data.
- Implicit CRS.

4.6 OGC SensorThings API

SensorThings is a proposed OGC standard that addresses data access for the Internet of Things (IoT). The vision of IoT is having devices in the world connected to the Internet to allow data retrieval and control. Part 1 is the current proposed standard and deals with access to data; part 2 is about control of devices, and is planned for future work.

SensorThings part 1 (referred to as SensorThings from here on in) provides a RESTful, JSON encoded API to retrieve data and metadata about 'things' that generate streams of data. The underlying information/resource model for the API uses O&M and has been influenced by the other SWE standards. The API follows patterns defined by the OData protocol[10]. Figure 10 shows an overview of the core resource model.

[9] http://api.ala.org.au/

[10] http://www.odata.org/

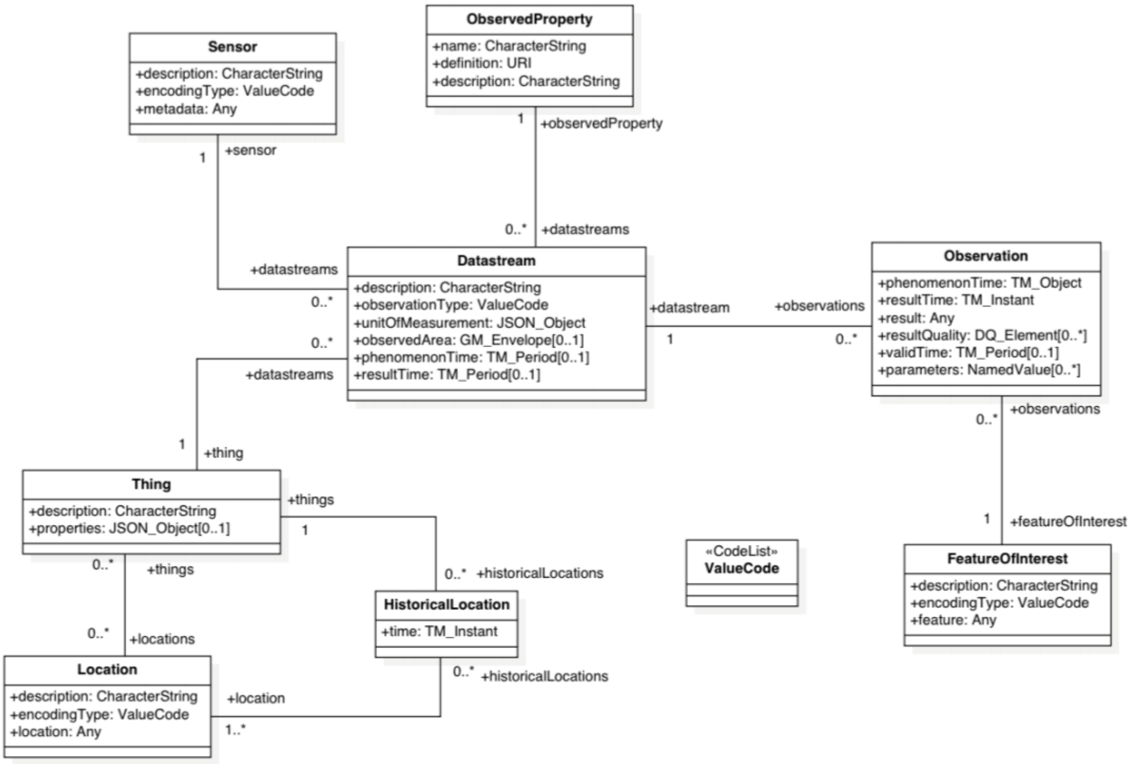


Figure 10 - SensorThings resource model

SensorThings splits the O&M model across two classes: Datastream and Observation. Datastreams are the top-level class with a subset of the O&M Observation properties: observedProperty, resultTime, phenomenonTime (see example in Listing 13). The observations property is analogous to result in the OM_Observation type, but in this case contains a set of Observation objects (see example in Listing 14). Each of these is typed according to its observation type from O&M (Measurement, Geometry etc.).

Listing 13 – A SensorThing Datastream

```
{
 "@iot.id": 1,
 "@iot.selfLink": "http://example.org/v1.0/Datastreams(1)",
 "Thing@iot.navigationLink": "HistoricalLocations(1)/Thing",
 "Sensor@iot.navigationLink": "Datastreams(1)/Sensor",
 "ObservedProperty@iot.navigationLink": "Datastreams(1)/ObservedProperty",
 "Observations@iot.navigationLink": "Datastreams(1)/Observations",
 "description": "This is a datastream measuring the temperature in an oven.",
 "unitOfMeasurement": {
 "name": "degree Celsius",
 "symbol": "°C",
 "definition": "http://unitsofmeasure.org/ucum.html#para-30"
 },
 "observationType": "http://www.opengis.net/def/observationType/OGCOM/2.0/OM_Measurement",
 "observedArea": {
 "type": "Polygon",
 "coordinates": [[[100,0],[101,0],[101,1],[100,1],[100,0]]]
 },
 "phenomenonTime": "2014-03-01T13:00:00Z/2015-05-11T15:30:00Z",
 "resultTime": "2014-03-01T13:00:00Z/2015-05-11T15:30:00Z"
}
```

Listing 14 - Observation

```
{
 "@iot.id": 1,
 "@iot.selfLink": "http://example.org/v1.0/Observations(1)",
 "FeatureOfInterest@iot.navigationLink": "Observations(1)/FeatureOfInterest",
 "Datastream@iot.navigationLink":"Observations(1)/Datastream",
 "phenomenonTime": "2014-12-31T11:59:59.00+08:00",
 "resultTime": "2014-12-31T11:59:59.00+08:00",
 "result": 70.4
}
```

The structure of the JSON encoding follows patterns defined by OData. It does not strictly follow the OData specification (Liang et al., 2015, p. 19), but reuses the common patterns, identifier, and link structures.

Key characteristics:

- Uses OData as a pattern for response encoding and link handling.
- Generic IoT problem space – applicable to most sensor applications.
- Light metadata.
- Linked resources.
- Implicit CRS.

4.7 UK National Biodiversity Network

The UK National Biodiversity Network (NBN) provides an API similar to that of the ALA, with a detailed API providing access to sites, taxon information, observations, organisations etc. It provides XML and JSON responses, with a service description provided in WADL[11] (see section 5.1.4).

4.8 GeoServices API

The GeoServices API is a suite of RESTful APIs are a slightly modified version of the ESRI APIs that are offered as part of the ESRI commercial GIS software packages. They were proposed as OGC standards in 2013, but subsequently withdrawn from the submission process. They provide a range of functionality, summarised in Table 3, which have some overlap with existing OGC web services.

These services encode JSON responses but make very little use of links, and thus lack the hypermedia attributes of RESTful services.

Table 3 - summary of GeoServices APIs

Service	Description	Related OGC

[11] https://data.nbn.org.uk/Documentation/Web_Services/Web_Services-REST/resources/restapi/application.wadl

		services/standards
Core	Defines common requirements classes across all services. This includes: • Common return structures. • Extensibility mechanisms. • Use of HTTP headers and return codes. • Idempotency. • JSON-P support (facilities cross-domain JSON requests). • Cross-origin resource sharing support. • Encoding coordinate reference systems. • Geometry encodings. • Feature encodings, limited to one geometry property. • Coded values. • Symbol encodings.	OWS Common; Web Feature Service, Web Mapping Service. OGC Abstract Topics.
Catalog	Provides a simple catalog of available services.	Catalog Service for the Web (CS/W)
Map Service	Access to maps as images, tiled maps, maps with coordinate transformations, time-indexed maps, request/identify features using specific geometries, search maps using query parameters.	Web Mapping Service (WMS)
Feature Service	Retrieve, add, update and delete spatial features. Add attachments to features. Search relationships between features.	Web Feature Service (WFS)
Geometry Service	Provides common geometry operations, including: project, simplify, length and area calculations, distance, convex hull, cut, difference, intersect etc.	Web Processing Service (WPS),
Image Service	Service to retrieve and query imagery data (e.g. satellite imagery). Supports	Web Coverage Service (WCS), Web Mapping Service (WMS).

	mosaics and coordinate transformations.	
Geoprocessing Service	Execute and monitor geoprocessing tasks asynchronously or synchronously.	Web Processing Service (WPS)
Geocoding Service	Lookup addresses relating to locations and visa versa (reverse geocoding).	OGC Open Location Services (OpenLS)

Key characteristics:

- Wide geospatial functionality.
- No use of links.
- No use of existing OGC standards.
- Uses JSON Schema to describe responses.
- Good separation of API endpoints into clear functions.
- Formal, not adopted, document-based specification.

4.9 Global Change Information System

The US Global Change Research Program (USGCRP) established the Global Change Information System (GCIS) to integrate any information relating to changes in the global environment. This includes significant datasets, reports, findings, people, models, images etc. It has been developed following RESTful principles, as well as those of Linked Data.

While its scope is huge, it has made practical first steps towards useful, accessible and linked datasets. A summary of the key resources within its RESTful API is shown in Figure 11. Each bubble below the top level (marked black) is a RESTful resource endpoint that provides access to a JSON encoding of the resource, with links provided to other relevant resources (e.g. links from publications, to authors and related datasets).

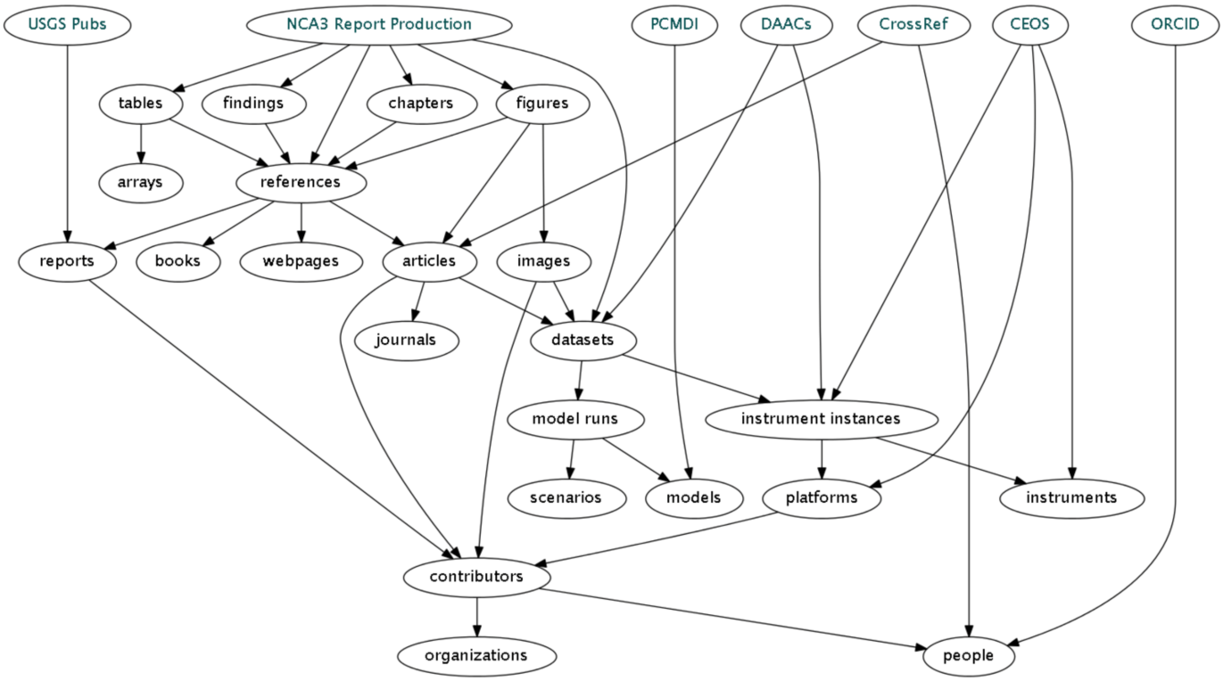


Figure 11 - Overview of the GCIS resource data model

The RESTful API is documented using Swagger (see section 5.1.1) as shown in Figure 12. The potential information available through is API is vast. Faceted search is available that supports spatial and thematic drill-down style discovery[12] of resources. A SPARQL endpoint is also available[13] for Linked Data clients. Multiple response formats are available to maximise data availability. For example, an organisation may be viewed as images of linked resources (e.g. CSIRO[14]). This facilitates exploration of the related resources, such as exploring reports published from people within an organisation.

[12] https://gcis-search.jpl.net/search

[13] http://data.globalchange.gov/sparql

[14] http://data.globalchange.gov/organization/commonwealth-scientific-industrial-research-organisation.svg

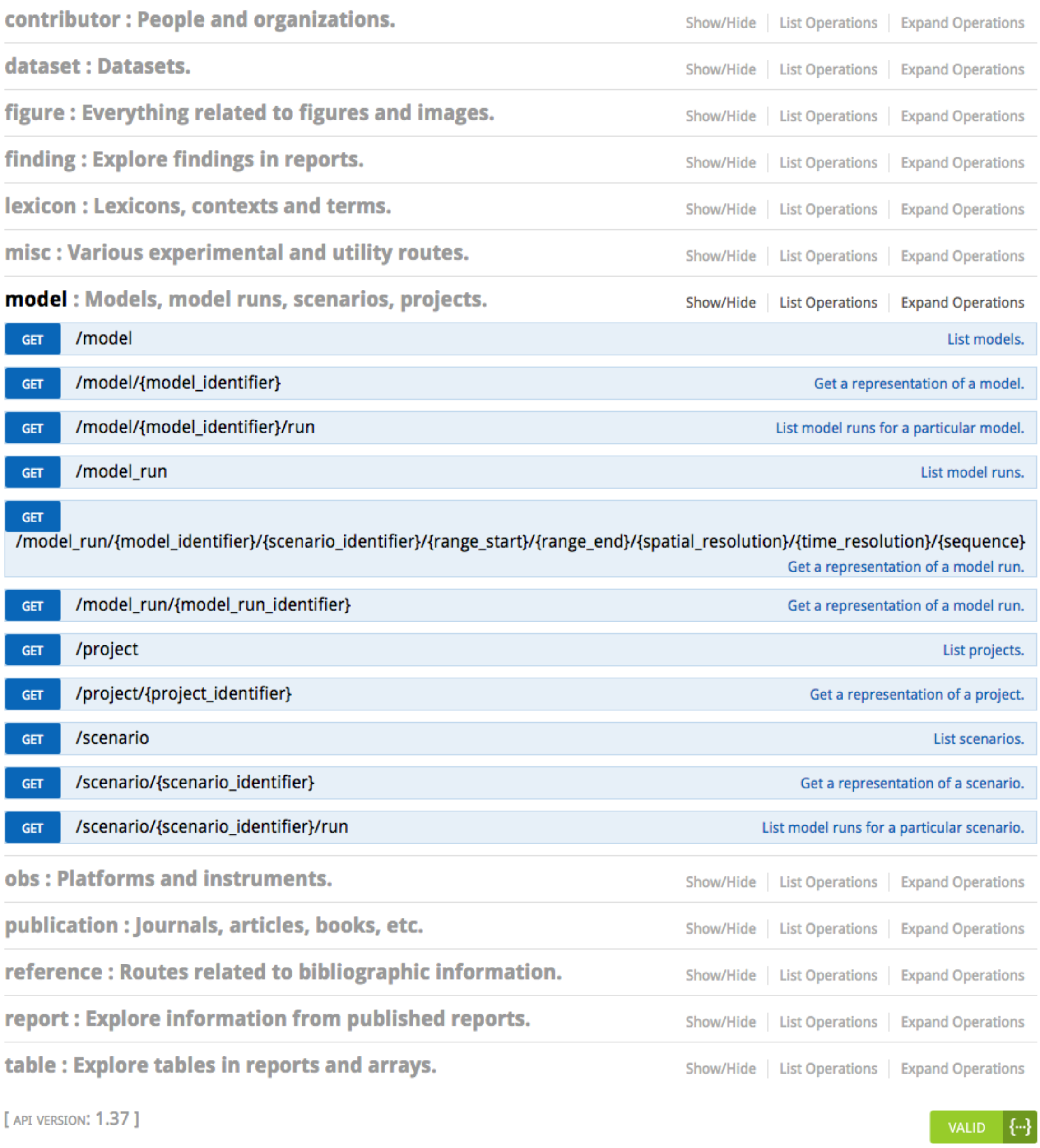


Figure 12 - GCIS RESTful API documentation

Key characteristics:

- Expansive scope.
- Swagger-based documentation.
- Lots of use of linking.
- Support for Linked Data encodings and access.

4.10 UK Environmental Agency – Bathing Water Quality API

The Bathing Water Quality provides access to the UK's water quality sampling data as well as the resulting assessments (compliance, swimming status etc.). The API has been developed as a Linked

Data API[15], which is a specification of how to provide simple RESTful APIs over linked data. A summary of the scope and structure of the API is shown in Figure 13.

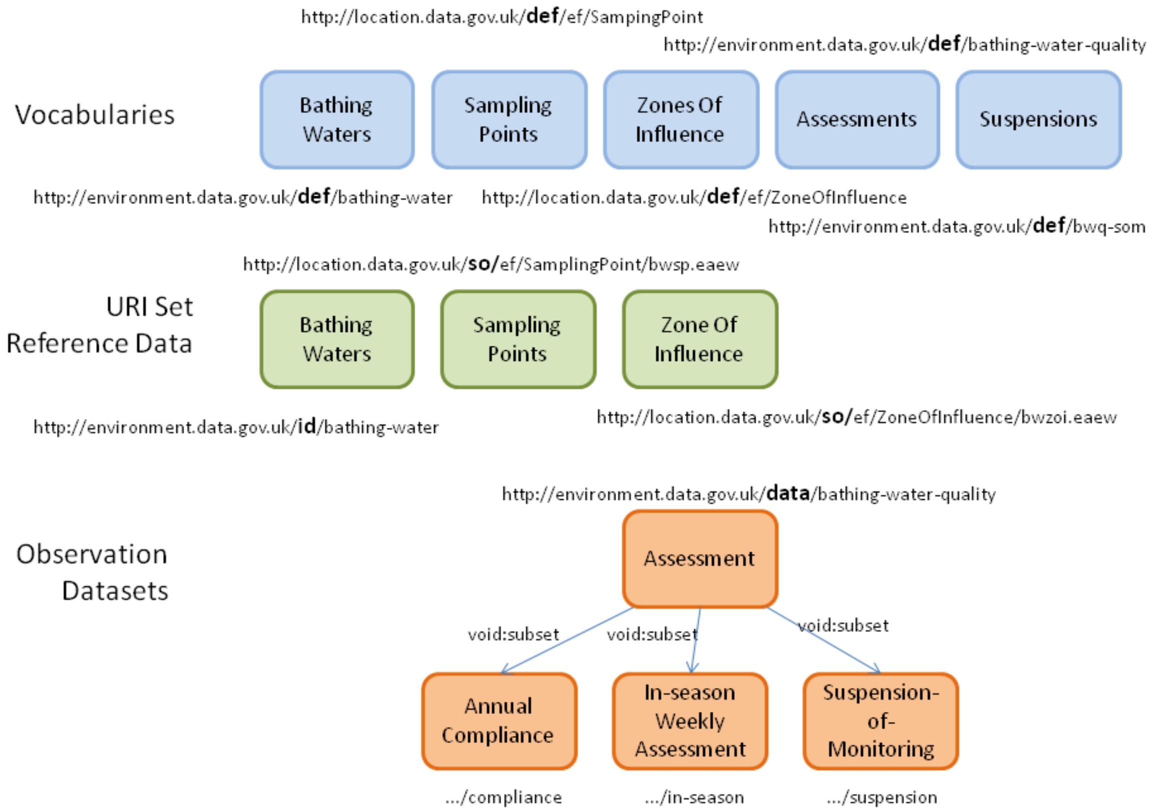


Figure 13 - UK Bathing Water Quality API

The Agency also provides a Bathing Water Profile[16] as a way of grouping information relating to specific bathing locations, such as the beach, rivers streams and pollution sources.

Key characteristics:

- Detailed documentation.
- Re-use of standard/common vocabularies.
- 'Native' Linked Data – supports multiple formats.
- Clear URI guidelines and adherence to LDA guidelines[17].

4.11 CKAN

The Comprehensive Knowledge Archive Network (CKAN) is an open source data management system that supports registration of data assets by third parties. It is used by a range of open data

[15] https://code.google.com/p/linked-data-api/wiki/Specification

[16] http://environment.data.gov.uk/bwq/doc/api-bwp-reference-v0.1.html#BathingWaterProfile

[17] https://github.com/UKGovLD/linked-data-api/blob/wiki/Specification.md

web sites, including http://datahub.io, http://data.gov.uk, http://data.gov.au/ and http://publicdata.eu/.

The functionality of the base CKAN web application includes search facilities, metadata retrieval and entry, geospatial functions, social extensions, visualisation and auditing. The RESTful API provides access to all of these functions.

Key characteristics:

- Generic data management and publication technology
- RESTful, JSON-based API for creating, updating and deleting records

4.12 Data Access Protocol (DAP), OpenDAP, Hyrax

OPeNDAP is the "Open-source Project for a Network Data Access Protocol", which provides open source implementations of the Data Access Protocol (DAP). The primary implementation is called Hyrax. DAP is an HTTP protocol that provides access to 'science' data without any dependence on how the format in which the data may be stored. It supports four generic data types: grids, sequences, arrays, and structures.

The DAP protocol uses HTTP GET requests to convey structural metadata (dataset descriptor), 'semantic' (dataset attribute) metadata and the data itself. A summary of these message types is shown in Table 4.

Table 4 - DAP messages

Request	Response	Content-type
Dataset Descriptor Structure (DDS)	DDS or Error	text/plain
Dataset Attribute Structure (DAS)	DAS or Error	text/plain
Data Dataset Descriptor Structure (DataDDS)	DataDDS or Error	application/octet
Server version	Version information as text	text/plain
Help	Help text describing all request-response pairs	text/plain

These messages allow clients to interrogate data sources and retrieve data in a binary encoding. Additionally, a constraint expression URL syntax allows name/variable selection (called projections) and variable filtering against relational expressions (called selections).

An example projection would be accessing an array using a hyperslab (array index) expression, e.g. [0:4]. An example selection would be a regular expression search across station names.

In terms of Web APIs, the DAP is a lower-level protocol which provides access to common data structures. For less sophisticated users or client requirements, it is likely that a layer would be provided on top of DAP that hides some of its complexity. An example of this is ERDDAP (see section 4.14).

4.13 THREDDS

Thematic Real-time Environmental Distributed Data Services (THREDDS) is a data catalog/server that provides web-based access to binary data sources, such as NetCDF, HDF, OPeNDAP and NEXRAD. It publishes a range of service interfaces including Web Coverages Service (WCS) Web Mapping Service (WMS), OPeNDAP and NetCDF subsetting service. It uses Unidata's Common Data Model (CDM)[18] as the key abstraction, which is summarised in Figure 14. This model is very similar to the abstractions used by the DAP.

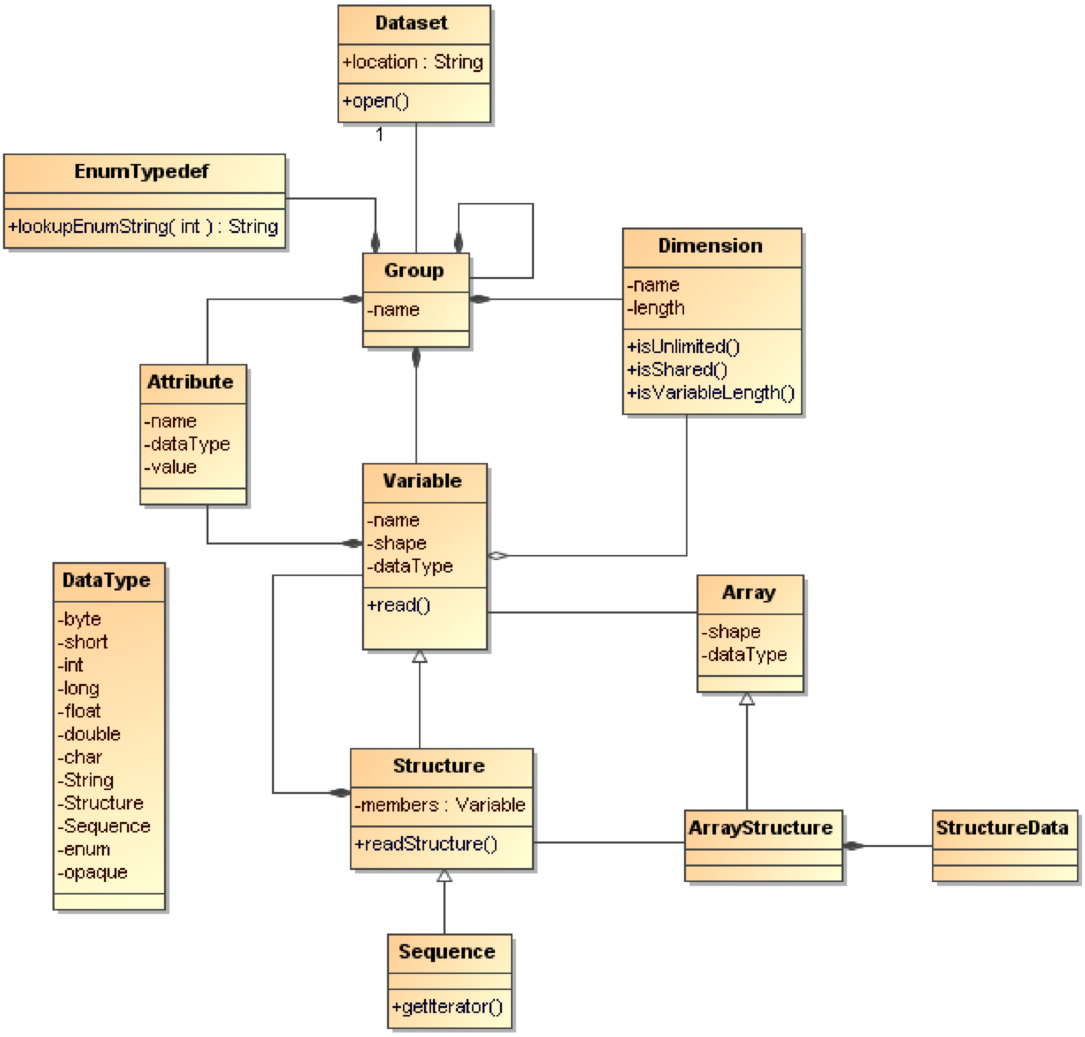


Figure 14 - Overview of Unidata's Common Data Model (CDM)

THREDDS is not a Web API in the sense of the definition within this report, however it does provide some REST-like interfaces and is a component within larger aggregated environmental data system discussed below.

[18] http://www.unidata.ucar.edu/software/thredds/current/netcdf-java/CDM/index.html

4.14 ERDDAP

ERDAPP is a data server that provides access to a range of data sources, using a range of formats and service interfaces. It can be viewed as a middleware data aggregator that supports integration of common scientific data formats and services. It does this using two abstractions for data structures: grids and tabular data. It makes use of DAP (see section 4.12) to do this.

ERDAPP provides a RESTful interface for searching and accessing data from the available services. It works well with THREDDS services and can be used to provide more simplified, consolidated access points to multiple THREDDS services.

The basic resources defined in the ERDAPP RESTful API are shown in Table 5.

Table 5 - summary of ERDDAP resources

Resource	URL
info	http://coastwatch.pfeg.noaa.gov/erddap/info/index.htmlTable?page=1&itemsPerPage=1000
search	http://coastwatch.pfeg.noaa.gov/erddap/search/index.htmlTable?page=1&itemsPerPage=1000&searchFor=
categorize	http://coastwatch.pfeg.noaa.gov/erddap/categorize/index.htmlTable?page=1&itemsPerPage=1000
griddap	http://coastwatch.pfeg.noaa.gov/erddap/griddap/index.htmlTable?page=1&itemsPerPage=1000
tabledap	http://coastwatch.pfeg.noaa.gov/erddap/tabledap/index.htmlTable?page=1&itemsPerPage=1000
wms	http://coastwatch.pfeg.noaa.gov/erddap/wms/index.htmlTable?page=1&itemsPerPage=1000

Using its key abstraction, ERDAPP can source data from SOS, WCS, WFS, ArcGIS, text, DAP, NetCDF among others, and republish using KML, JSON, R, WFS, SOS, WCS and so on. In this sense it is a powerful data server that can operate as a broker between a users' desired output and the original data source.

Key characteristics:

- Aggregator of common data access protocols and standards
- RESTful API using grid and table simplification pattern
- Supports multiple service interfaces and response formats

5 Web API practices

5.1 Describing service interfaces

Having machine-readable descriptions of APIs is a useful feature to allow automated documentation, testing, discoverability, generated client bindings, and so on. The Web Service Definition Language (WSDL) is the W3C standard for service descriptions associated with more traditional Web Services (as defined in section 2), typically using XML.

There has been recent growth in specifications describing RESTful APIs. These arose from the need to automatically produce API documentation. However, there are other benefits to having machine-readable description of an API endpoint, including discoverability and support for automated testing frameworks.

Some of the more popular frameworks and technologies are identified below. These are largely bottom-up specifications that arose from implementations. It would be desirable to have an open standard in this space, since there is already strong similarity between these specifications. This section does not attempt to pick a winner, but provides short summaries of their background and basic functions.

5.1.1 Swagger specifications

Swagger is a set of tools and services that help API developers generate documentation for APIs. The Swagger UI[19] renders a documentation page that provides full description of endpoints, with in-line forms allowing test calls to be made to services. The UI can be generated from an API specification document, which could be handwritten or generated from a service.

The specification of a service is done using JSON or YAML that follows the Swagger specification schema[20]. A JSON Schema is available to validate specification documents[21]. The specification defines all the standard parts of a RESTful API: the resource endpoints, supported functions (GET, POST, DELETE, OPTIONS etc.), media types, parameters, and responses.

Swagger was used in the WaterML2.0 part 2 Interoperability Experiment to document the CSIRO RESTful API[22], as shown in Figure 15. The addition of inline parameter description and testing provides an interactive document that is useful for developers. The service description is also rendered as a RESTful API, which shows and example of the service description document[23].

[19] http://petstore.swagger.io/

[20] http://swagger.io/specification/

[21] https://github.com/swagger-api/validator-badge

[22] http://waterml2.csiro.au/rgs-api/v1/docs

[23] http://waterml2.csiro.au/rgs-api/v1/docs/api-docs/conversion

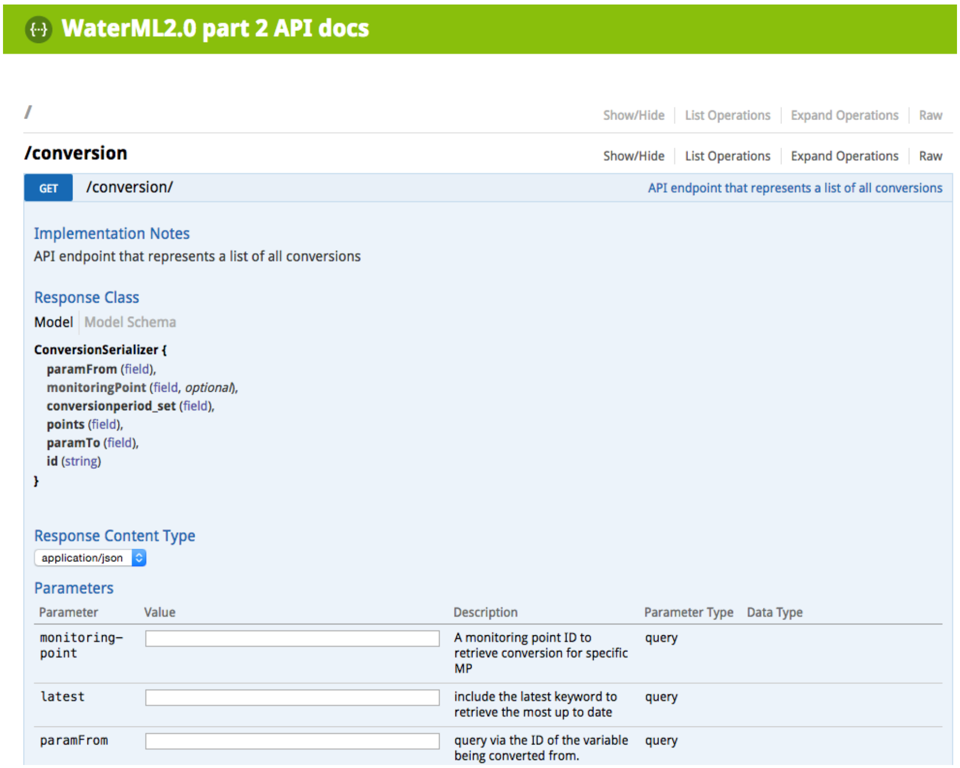


Figure 15 - Swagger UI of the WaterML2.0 part 2 RESTful API

5.1.2 RESTful API Modeling Language (RAML)

The RESTful API Modeling Language (RAML) is a non-proprietary open specification that is similar to the Swagger specification in terms of its scope and implementation. It provides a JSON and YAML description of RESTful APIs[24].

5.1.3 RESTDesc

RESTDesc[25] uses semantic approaches to define the functionality of hypermedia APIs. It uses the RDF Turtle encoding to describe the resources and hypermedia links using existing semantic vocabularies. In this sense it is simply a set of patterns for how to use existing vocabularies to describe RESTful APIs.

5.1.4 Web Application Description Language (WADL)

The Web Application Description Language (WADL)[26] is a W3C member submission (not a standard), which provides an XML schema to describe Web API interfaces. It does this through defining the API's resources, the supported operations on each resource, and available query

[24] http://raml.org/spec.html

[25] http://restdesc.org/

[26] http://www.w3.org/Submission/wadl/

parameters. The specification and underlying schema[27] are not overly complicated and focus on the core contracts for clients communicating with a Web API.

5.1.5 API Blueprint

Blueprint[28] is an API description based on Markdown, which generates a JSON specification format. The aim is to provide a more human friendly editing language (i.e. Markdown). Because it is Markdown based, the specification is defined in the form of specific keywords combined with section headers.

5.1.6 Mashery I/O Docs

Mashery[29] has a suite of API offerings including services for API management, development, documentation, security, and testing. The documentation engine is very similar to that of Swagger. It offers generated HTML pages with forms to interact with an API. Its specification does not appear to be open.

5.1.7 RESTful Service Description Language (RSDL)

The RESTful Service Description Language (RSDL)[30] is an XML format for describing RESTful services, including resources, media types, methods, links, authentication, parameters etc.

5.2 Specifications for API responses

The previous section covers how to describe a specific RESTful API, while this section covers emerging approaches to structure Web API responses in a common way. The service description approach would be used to describe an existing API; this section covers practices that may be followed when designing an API from scratch. Generally the purpose of these specifications is to support the REST concept of hypermedia as the engine of application state (HATEOAS).

The general idea behind HATEOAS is that any client should be able to navigate through available application states by traversing links provided by the server. These links are dynamically provided by the server and should not be hard-coded into clients. Thereby the server can drive application behaviour without change to the client.

5.2.1 I-JSON

Internet JSON (I-JSON) is a specification that adds additional constraints to the IETF JSON[31] to aid interoperability and software compatibility for JSON on the Web. It is not a specific definition of

---

[27] http://www.w3.org/Submission/wadl/wadl.xsd

[28] https://apiblueprint.org/

[29] http://www.mashery.com/api/io-docs

[30] http://www.balisage.net/Proceedings/vol10/html/Robie01/BalisageVol10-Robie01.html

[31] https://www.ietf.org/rfc/rfc4627.txt

response formatting, but a JSON subset that reduces variation and may be used by all Web APIs that use JSON encodings on the Web.

The main requirements introduced in iJSON include:

- **Encoding**: I-JSON Messages must be encoded using UTF-8.
- **Floating point numbers**: numbers must be exactly representable as IEEE 754:2008 64-bit binary floating point numbers
- **No duplicate property names**: JSON objects should not have properties with duplicate names.
- **Strict compliance**: software receiving a JSON document that does not meet the iJSON requirements should not trust or act on any content of the message.

There are also some recommendations that include: date time strings using ISO8601, binary data encoded using base64, and use of objects or arrays as top-level constructs.

The OGC is considering recommending I-JSON for use within JSON encoded OGC standards.

5.2.2 Hypertext Application Language (HAL)

HAL[32] provides a standardised response structuring for JSON and XML responses for Web APIs. The basis for the structure is through the separation of resources, links, and embedded resources, as shown in Figure 16.

[32] http://stateless.co/hal_specification.html

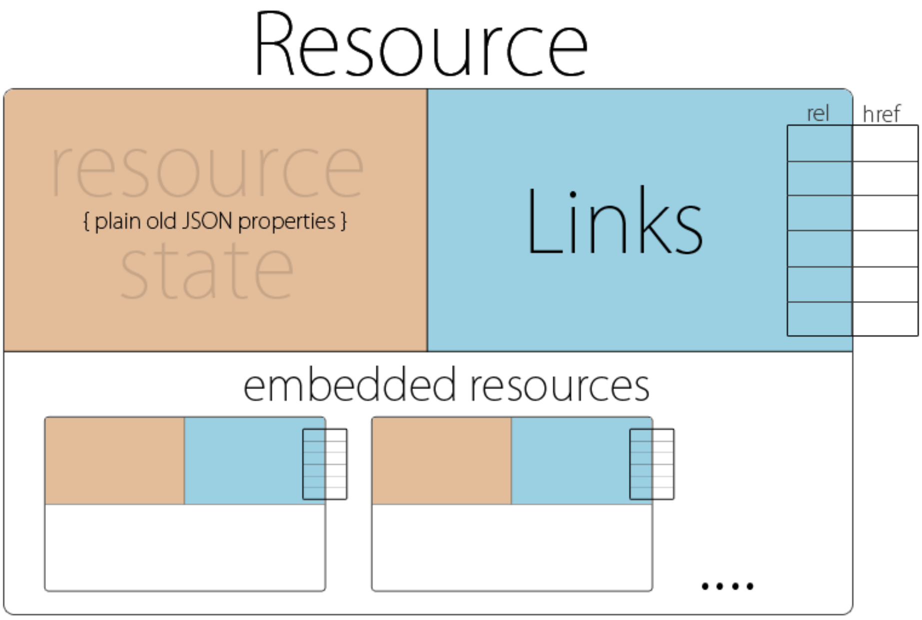


Figure 16 - HAL structuring (source http://stateless.co/hal_specification.html)

HAL makes strong use of the hypermedia aspect of RESTful APIs through the use of link encodings. It also supports URL templates[33] in its link definitions. An example response is shown below.


```
{
    "_links": {
        "self": { "href": "/orders" },
        "curies": [{ "name": "ea", "href": "http://example.com/docs/rels/{rel}", "templated": true }],
        "next": { "href": "/orders?page=2" },
        "ea:find": {
            "href": "/orders{?id}",
            "templated": true
        },
        "ea:admin": [{
            "href": "/admins/2",
            "title": "Fred"
        }, {
            "href": "/admins/5",
            "title": "Kate"
        }]
    },
    "currentlyProcessing": 14,
    "shippedToday": 20,
    "_embedded": {
        "ea:order": [{
            "_links": {
                "self": { "href": "/orders/123" },
                "ea:basket": { "href": "/baskets/98712" },
                "ea:customer": { "href": "/customers/7809" }
            },
            "total": 30.00,
            "currency": "USD",
            "status": "shipped"
        }, {
            "_links": {
                "self": { "href": "/orders/124" },
```


[33] https://tools.ietf.org/html/rfc6570

```
                "ea:basket": { "href": "/baskets/97213" },
                "ea:customer": { "href": "/customers/12369" }
            },
            "total": 20.00,
            "currency": "USD",
            "status": "processing"
        }]
    }
}
```

HAL has been drafted as an IETF standard[34].

5.2.3 Collection + JSON

Collection + JSON[35] specifies a common structuring for retrieving, updating and deleting collections of resources through an API. It does this through specifying patterns for using HTTP Link Headers and providing a common structuring of collections with embedded objects. It has similar goals to that of HAL.

A collection + JSON media type (application/vnd.collection+json) has been registered with IANA as a common media type[36]. However the actual structuring has no status as a standard.

5.2.4 SIREN

SIREN[37] is a structuring specification for JSON responses in Web APIs. Its scope is very similar to that of HAL and Collection + JSON. It separates three key aspects: entities (resources), links and actions. There is structure for embedding entities in the same way HAL does.

An example response is shown below.

```
{
  "class": [ "order" ],
  "properties": {
      "orderNumber": 42,
      "itemCount": 3,
      "status": "pending"
  },
  "entities": [
    {
      "class": [ "items", "collection" ],
      "rel": [ "http://x.io/rels/order-items" ],
      "href": "http://api.x.io/orders/42/items"
    },
    {
      "class": [ "info", "customer" ],
      "rel": [ "http://x.io/rels/customer" ],
      "properties": {
        "customerId": "pj123",
        "name": "Peter Joseph"
      },
      "links": [
        { "rel": [ "self" ], "href": "http://api.x.io/customers/pj123" }
      ]
```

[34] https://tools.ietf.org/html/draft-kelly-json-hal-07

[35] http://amundsen.com/media-types/collection/format/

[36] http://www.iana.org/assignments/media-types/application/vnd.collection+json

[37] https://github.com/kevinswiber/siren/blob/master/README.md

```
    }
  ],
  "actions": [
    {
      "name": "add-item",
      "title": "Add Item",
      "method": "POST",
      "href": "http://api.x.io/orders/42/items",
      "type": "application/x-www-form-urlencoded",
      "fields": [
        { "name": "orderNumber", "type": "hidden", "value": "42" },
        { "name": "productCode", "type": "text" },
        { "name": "quantity", "type": "number" }
      ]
    }
  ],
  "links": [
    { "rel": [ "self" ], "href": "http://api.x.io/orders/42" },
    { "rel": [ "previous" ], "href": "http://api.x.io/orders/41" },
    { "rel": [ "next" ], "href": "http://api.x.io/orders/43" }
  ]
}
```

5.2.5 Uniform Basis for Exchanging Representations (UBER)

UBER[38] is a document format that supports state transfers (hypermedia) and object encoding. It is designed to be protocol-agnostic, with guidance on how to implement it using HTTP with XML or JSON. It achieves this through use of reserved string properties that carry specific semantics in terms of state transitions (actions), templates, and embedded or linked resources.

5.2.6 JSON API

JSON API[39] is a specification of conventions for structuring JSON responses from Web APIs. It addresses encoding issues such as content negotiation, document structure (links, compound documents, metadata), sorting, pagination, and creating/updating/deleting resources. The implementations appear to growing[40] and the Github repository is very active. It addresses a number of common issues across APIs, some of which have been identified in OGC REST activities.

5.2.7 Hydra

Hydra[41] uses JSON-LD with a vocabulary[42] to allow a server to advertise its state transitions to clients, again to enable HATEOAS properties of Web APIs. The vocabulary defines the common concepts within a RESTful API: Resources, Operations, Status Codes, Paged Collections, and Links. A summary of the vocabulary is shown in Figure 17.

[38] https://rawgit.com/uber-hypermedia/specification/master/uber-hypermedia.html

[39] http://jsonapi.org/

[40] http://jsonapi.org/implementations/

[41] http://www.markus-lanthaler.com/hydra/

[42] http://www.hydra-cg.com/spec/latest/core/

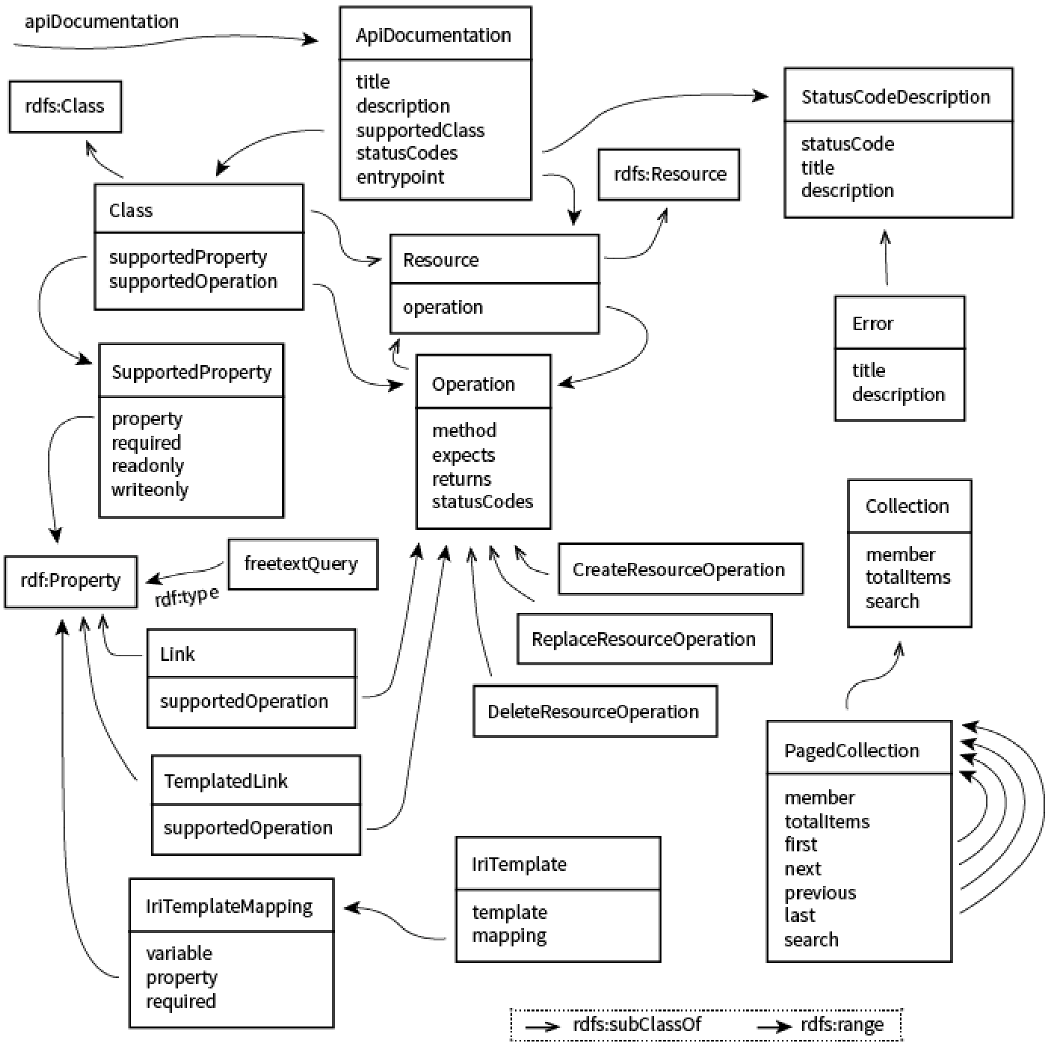


Figure 17 - Hydra core vocabulary for describing Web APIs

This vocabulary is then used to embed JSON-LD descriptions in JSON documents that describe a Web API. There is an active W3C Community Group[43] supporting the Hydra specification. The Hydra vocabulary is used within the Linked Data Fragments[44] implementations, which is a generalisation on access mechanisms for Linked Data.

5.3 API management

API management is a broad term used to describe a number of features required for operating and managing APIs through their lifecycle. These include the following types of functions:

- **Analytics**: report on usage, failed and slow requests, common queries etc.
- **API portals**: provides harmonised developer documentation, login, key management etc.
- **Traffic and performance management**: set request throttling, rate limits, quotas etc.

[43] https://www.w3.org/community/hydra/

[44] http://www.hydra-cg.com/spec/latest/linked-data-fragments/

- **Service catalogs**: discovery of services, status reporting.
- **Security**: authorisation and authentication, OAuth, single sign on, access control, key creation etc.
- **Deployment processes**: version management, test/sandbox environments, deprecation, etc.

These features are becoming increasingly important as organisations deploy more APIs, and look to manage and monetise their APIs. There are many API management frameworks available, with products focussing on specific features or supporting the suite of API management tools. An in depth review of these is out of scope for this report, however a summary is provided in Table 6.

The table represents a very superficial review: no installations or testing of features was performed. There are components that focus on particular features and some that have richer support within a feature category. For example, ‘security’ can range from basic HTTP authentication to OAuth2.0, single sign-on, and encryption etc. The categories are only to indicate some level of support. A more in depth analysis would be required to make a well-informed decision.

Table 6 - Basic comparison of API manager frameworks

Name	Analytics	API portal	Traffic	Service Catalog	Security	Deployment	License	Comments
Tyk	✓	✓	✓	✓	✓	✓	Open Source, cloud or supported	
WSO2 API Manager	✓	✓	✓	✓ [45]	✓	✓	Open Source	Part of the WSO2 suite of Web Services.
Gluu	×	×	×	×	✓✓	×	Open Source (multiple packages)	
API Axle	✓	×	✓	×	✓	×	Open Source	Focussed on key management.
Kong	✓[46]	×[47]	✓	✓[48]	✓	×[49]		

[45] With other WSO2 products

[46] Via integration with Galileo

[47] See https://github.com/Mashape/kong/issues/391, also available via integration with https://gelato.io/.

[48] Via https://market.mashape.com/

3Scale API Management	✓	✓	✓	×	✓	?	Closed source, cloud-hosted freemium model	
apigee Volos	×	×	✓	×	✓	×	Open Source	Supports cache management
API Umbrella	✓	✓	✓	×	✓	×	Open Source	
Repose	✓	×	✓	×	✓	×	Open Source	Analytics is logging only
StrongLoop (uses Loopback)	✓	✓	✓	✓	✓	✓	Commercial with community (very limited) edition.	Focused on creation and deployment of NodeJS APIs.
API Man	✓	✓	✓	✓	✓	✓	Open Source	Development roadmap available here

49 Via https://market.mashape.com

6 Conclusions

Web APIs hold great potential for agencies to provide on-demand, tailored and robust environmental data. Accordingly, the design and implementation of Web APIs should be done with careful consideration of existing practices. It is beneficial to be consistent with existing practices, while being wary not to adopt practices that gain no traction.

It is obvious that most organisations are tackling the API design challenges. Many services have evolved from the ground up and organisations are attempting to consolidate into more simple, cohesive Web APIs. All these APIs have made simplifications and trade-offs when designing their APIs to suit their intended audience. This report summarises a broad range of existing practices with the aim of informing the design of future environmental data Web APIs. To complement the analysis within this report, we provide some general observations and learnings based on our experiences using and developing Web APIs.

6.1 General

Recommendation 1 – **focus on core requirements**

When designing APIs it is important to understand your users, while keeping in mind they may not yet exist. Try to identify core requirements as a starting point, and build out as your user's requirements and needs grow.

Recommendation 2 – **clearly separate functionality into separate APIs**

Highly abstracted APIs can be difficult to understand and may be a barrier for new users. Trying to do too many things in a single API or Web Service can result in an overly complex, hard to understand API. Partitioning APIs into concrete API endpoints, each having clearly stated functions help to build understanding.

RESTful APIs have a natural separation into 'sub APIs' through the use of the resource end points. This can be a useful way of separating clear logical parts of the API that reflect nouns. For example the GitHub API resources use very concrete nouns that reflect the nature of the API interaction – see Figure 18.

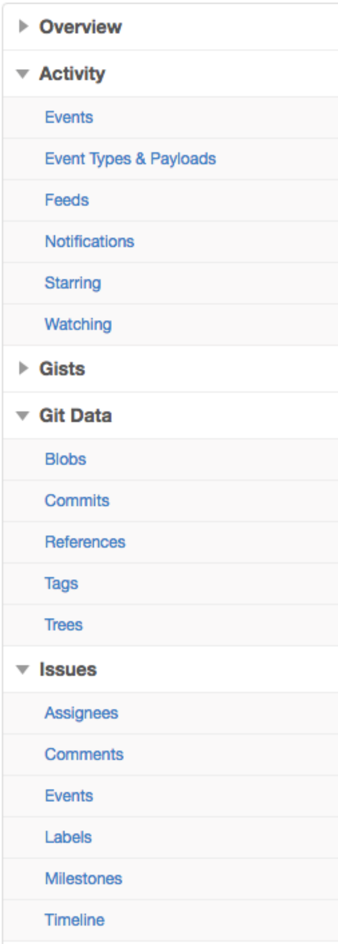


Figure 18 - Example GitHub API resources

Recommendation 3 – productise your Web APIs

It appears to be beneficial to give Web APIs a product feel. They should be branded consistently and behave consistently where there are multiple APIs, or endpoints provided.

Most of the popular Web APIs provide an initial documentation block that defines all the common behaviour across all its resources/APIs. For example Figure 19 shows Stripe and GitHub's base topics, which cover many of the common API components described in this report.

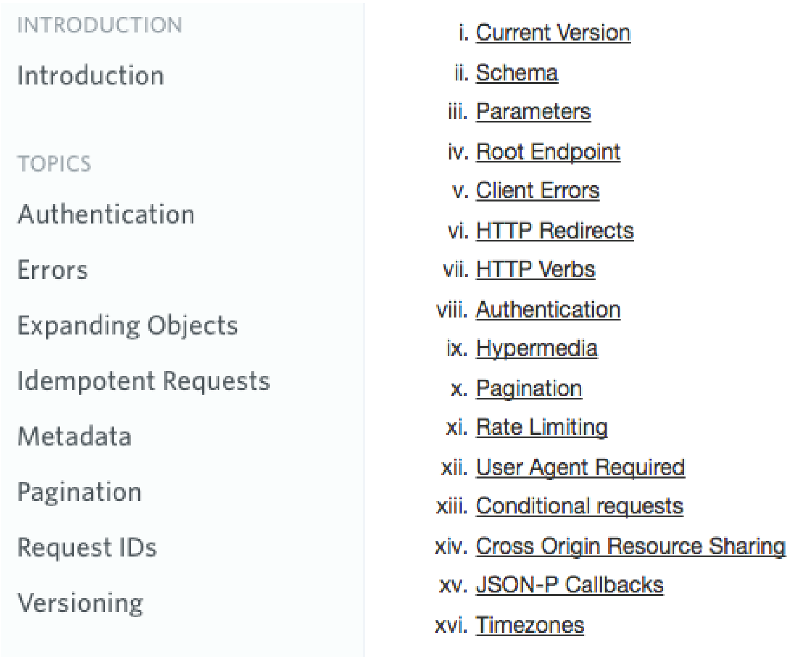


Figure 19 - Stripe API (left) GitHub API (right)

Often successful APIs reflect the product of the company very closely. It is even sometimes the case that a company's product uses its own API, which can be very beneficial.

Recommendation 4 - publish targeted documentation

Informal, tutorial style, documentation appeals to developers and is the most commonly read documentation. Avoid publishing only example URLs.

There are some excellent documentation engines available on the web. These provide navigation structures, with narrative sections, and example code sections. For example, the Stripe API (a web-based payment system) structures its documentation page into three parts: resources, description and code examples; as shown in Figure 20.

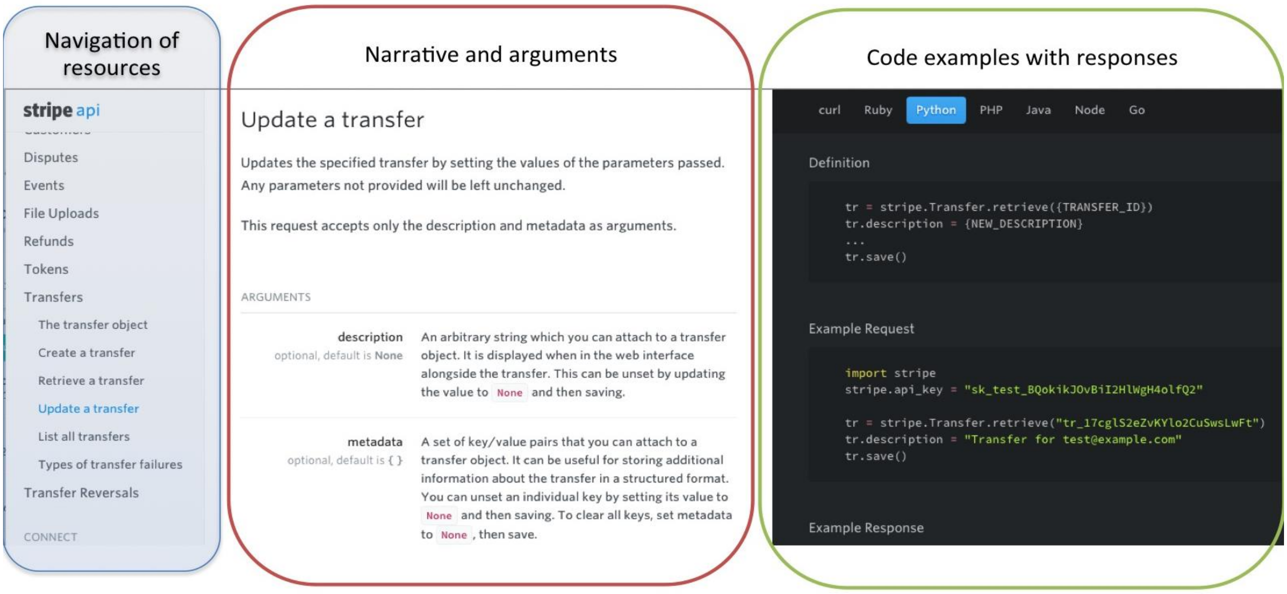


Figure 20 - Stripe API documentation

This type of documentation gives developers everything they need to interact with an API and rapidly get feedback on their interactions.

Many Web APIs have at a minimum the following core sections within their documentation:

- **Quickstart**: shortest possible path to getting something done. Developers thrive on getting some quick results. Make it easy.
- **Tutorials**: longer walkthroughs that tackle specific goals, such as 'discovering the temporal range of available observations for a site'.
- **API reference**: all the gory details of your API. Often will be supported by a tool such as Swagger or other doc-generators.

SDKS (software development kits): Links to client implementation of the API in different languages or different platforms (e.g. mobile). Typically these point to Github pages with supporting tools. Often consist of combination of organisation-developed and community-developed clients.

Recommendation 5 – foster a community

Provide community resources, such as forums and easy feedback mechanisms, to allow users and developers to share experiences and issues. APIs are successful when they are used. Having an organisational representative that is tasked with engaging with the community can be hugely valuable.

Recommendation 6 – use domain abstractions and document them

Identify key simplifications and domain abstractions to reduce complexity. *Clearly document these to the users.* This will avoid confusion.

Recommendation 7 – include your Web APIs in your enterprise architecture

Including your Web APIs in your enterprise architecture appears to be a practice that can bring many benefits. They should not be 'bolted on', but rather may become a point of convergence for multiple systems: *a platform*.

Recommendation 8 – document using common use cases

Document common usages of your APIs that achieve certain goals, especially when they require multi-step requests to the API. For example, 'I want to discover all observations available at a particular station'.

6.2 Technical

Recommendation 9 – Use API documentation frameworks

Make use of automated API documentation where possible. These can often be synched directly with an implementation version, which helps to minimise divergence. Some also provide interactive (e.g. Swagger) documentation that allows inline requests to be made. This helps to lower the barrier of entry for developers and quickly builds understanding.

Recommendation 10 – be cautious when selecting API description languages

Avoid cutting edge/emerging API description languages and response patterns. As discussed in 5.1 and 5.2, there has been an explosion in the number of these. Picking a winner is difficult. Using an overly complex, non-supported service description framework and/or response structure can be an impediment to developers. Ensure there is at least some uptake and solid usage before adopting a practice.

Recommendation 11 – Consider using an API manager

There is a range of these available. Some may require customisation, or even coupling with other technologies. A more in depth analysis of these is recommended if selecting one for use; along with alignment of organisational IT practices (e.g. supported OS, languages, open source licencing etc.). Generally they offer solutions to some very common problems in deployment and management of APIs.

Recommendation 12 – Consider conforming to the I-JSON specification

The I-JSON Message Format (Bray, 2016) defines some precise restrictions (see 5.2.1) on the use of JSON to aid interoperability, without adding implementation burden.

Recommendation 13 – Don't do anything strange

There are many Web API best practices (see references), many of which agree on some common patterns, some of which are already part of the HTTP standard. Always strive to use HTTP consistently. The use of best practices requires consideration in each case. HTTP related practices include:

- **Structuring URIs** (Masse, 2011a): use of slashes/hashes, prefer all lowercase, hyphens not underscores, use of sub-domains.
- **Use of HTTP request methods**: use of GET, POST, PUT, DELETE and OPTIONS. Some convergence is occurring on the use of these within Web APIs (Masse, 2011b).
- **Use HTTP response status codes:** these are generally well documented and their proper use appears to be growing.
- **Use HTTP headers,** including (but not limited to):
  - Use content-types and media-type syntax for specifying message content type,
  - Use last-modified, cache-control, expires and date headers for proper cache interactions,
  - Use Location headers for URIs of newly created resources.

Recommendation 14 – Use consistent link representation and error messages

These are two areas where there is no widely agreed approach (see section 5.2). Even so, it is important to choose an approach, clearly document it and be consistent across all Web APIs being implemented.

Recommendation 15 – Define your versioning practices

There's an extensive online debate (Dierking, 2016)(Zazueta, 2015)(Overflow, 2016) (Sahni, 2016) about versioning RESTful APIs. The two prevailing options are to include a version in the URI (the 'pragmatic approach') or use HTTP headers (the 'academic' approach). Understand the trade-offs, choose one, and stick with it.

6.3 Modelling the domain

Recommendation 16 – use concrete resource names

Using concrete concepts for resources with clear definitions aids understanding of APIs. Avoid vague, internal concepts, which only have meaning in certain sub-domains.

Recommendation 17 – use design patterns for modelling the domain

Consider making use of domain-driven design patterns (Evans, 2003) when tackling domain complexity. Designing Web APIs will inevitably involve interaction with many existing systems. Patterns such as Anticorruption Layer, Ubiquitous Language, Domain Vision Statement and Bounded Context (Evans, 2003) all provide useful starting points for solving domain modelling and integration problems.

Recommendation 18 – involve the experts

Involve domain specialists in resource design. In organisations as complex as multi-disciplinary environmental agencies, there won't be a single information model or Web API to rule them all. Rather there exists multiple domain contexts (Bounded Contexts - (Fowler, 2014) and (Evans, 2003)) and words with different meaning (polysemes), e.g. what is a monitoring station? Any domain-level simplifications should be reviewed within these contexts to ensure they make sense.

Recommendation 19 – use abstractions for common sampling regimes

Consider point, grid and trajectory abstractions. This has been used effectively in existing APIs (e.g. DAP, NetCDF) as a way of delineating different data types.

Recommendation 20 – Seek help from active groups

Monitor the outcomes of the joint W3C/OGC 'Spatial Data on the Web' working group[50]. It is working on general recommendations for publishing spatial data on the web in a consistent manner. These should be very useful in guiding convergence in some of the issues covered in this report.

[50] http://www.w3.org/2015/spatial/wiki/Main_Page

Appendix A

Summary of example data products from the Australian Bureau of Meteorology

id	title	url	Blurb	active	Type of service
7	Australia rainfall and river conditions	http://www.bom.gov.au/australia/flood/	The rainfall and river conditions website provides rainfall and river data collected for flood warning purposes from a network of stations across Australia.	Y	Web application;
8	Australian Baseline Sea Level Monitoring Project	http://www.bom.gov.au/oceanography/projects/abslmp/abslmp.shtml	This project is designed to monitor sea level around the coastline of Australia, with an emphasis on the enhanced greenhouse effect. Selected hourly, monthly and yearly reports are currently available.	Y	PDF reports; CSV download
13	Australian climate influences	http://www.bom.gov.au/watl/about-weather-and-climate/australian-climate-influences.shtml	This online schematic diagram presents the main influences on the Australian climate.	Y	Information; educational content.
14	Australian Climate Observations Reference Network - Surface Air Temperature (ACORN-SAT) dataset	http://www.bom.gov.au/climate/change/acorn-sat/	The ACORN-SAT dataset provides a record of daily temperatures over the last 100 years.	Y	PDF reports; CSV download; web application
19	Australian Hydrological Geospatial Fabric (Geofabric)	http://www.bom.gov.au/water/geofabric/index.shtml	Geofabric is a specialised Geographic Information System. It registers the spatial relationships between important hydrological features such as rivers, water bodies, aquifers and monitoring points.	Y	Spatial data download via FTP; web application; OGC web

					services.
21	Australian Meteorological and Oceanographic Journal	http://www.bom.gov.au/amoj/index.shtml	The Australian Meteorological and Oceanographic Journal presents articles for the atmospheric, oceanic and related sciences on topics relevant to the Southern Hemisphere.	Y	Journal; information.
23	Australian Network of Hydrologic Reference Stations web portal	http://www.bom.gov.au/water/hrs/	This interactive tool allows users to interrogate, display and download data about streamflow and streamflow variability in Australia.	Y	Web application; CSV download; analysis
27	Australian Water Resources Assessments	http://www.bom.gov.au/water/awra/	Australian Water Resources Assessment reports highlight patterns, trends and variability in water quantity at regional to national scales and over time from months to decades.	Y	PDF reports; CSV download
34	Australian climate change projections	http://www.climatechangeinaustralia.gov.au	Climate Change in Australia shows how Australia's climate may change in the future. Climate change projections are available for low, mid-range and high greenhouse gas scenarios - drawing on scenarios developed for the Intergovernmental Panel on Climate Change (IPCC) Fourth Assessment Report.	Y	Information; educational content; data download (CSV, JSON, NetCDF); analysis
35	Climate Data Online	http://www.bom.gov.au/climate/data/index.shtml	Climate Data Online provides access to a range of statistics, historical weather observations, climatology maps, and other Australian climate data.	Y	CSV download; PDF tables.
36	Climate Model Summary	http://www.bom.gov.au/climate/ahead/model-summary.shtml	The Climate Model Summary provides Pacific and Indian Ocean temperature outlooks for the next six months based on a survey of international climate models.	Y	Information; educational; model exploration.
37	Climate statements - monthly, seasonal, annual	http://www.bom.gov.au/climate/current/index.shtml	Climate statements provide links to monthly, seasonal and annual climate summaries routinely issued by the Bureau of Meteorology.	Y	Maps (images); data summaries (tables); CSV summaries; PDFs
42	Drought	http://www.bom.gov	The Drought Statement provides a monthly summary of rainfall	Y	Maps (images).

	Statement	.au/climate/drought/drought.shtml	deficiencies in Australia.		
46	El Niño-Southern Oscillation Wrap-Up	http://www.bom.gov.au/climate/enso/	The ENSO Wrap-Up provides information on El Niño/La Niña weather patterns for Australia.	Y	Information (text); graphs (images);
49	enGauge newsletter	http://www.bom.gov.au/water/about/publications/index.shtml	enGauge is an online newsletter about all things associated with water, such as dam water levels, new policies and regulations and catchment information.	Y	Newsletters
54	Fire weather warnings	http://www.bom.gov.au/weather-services/bushfire/	Fire weather warnings are issued when the weather conditions are conducive to the spread of dangerous bushfires.	Y	Warnings; RSS, XML, AMOC
55	Forecast Explorer (RETIRED AT 14 AUGUST)	http://www.bom.gov.au/australia/index.shtml	Forecast Explorer provides an interactive web display of forecasts out to seven days for Victoria, Tasmania, South Australia and New South Wales. Other states and territories will be added 2013-2015.	N	Retired.
57	Frost potential	http://www.bom.gov.au/jsp/watl/weather/frost.jsp	Frost potential maps show forecast low temperature thresholds for various Australian weather station locations. This is a trial service.	Y	Web application;
60	High seas forecasts	http://www.bom.gov.au/marine/high-seas.shtml	High seas forecasts are issued twice daily and include information on wind, waves, temperature and rain for areas beyond the coastal waters surrounding Australia.	Y	Web application; text data.
62	Joint Australian Tsunami Warning Centre	http://www.bom.gov.au/tsunami/index.shtml	The Joint Australian Tsunami Warning Centre is operated 24 hours a day and detects, monitors, verifies and warns of any tsunami threat to the coastline of Australia and its offshore territories.	Y	Web application;
67	Madden-Julian Oscillation monitoring	http://www.bom.gov.au/climate/mjo/	Madden-Julian Oscillation (MJO) monitoring provides data on major fluctuations in tropical weather, attributed to this weather pattern.	Y	Web application; images; PDF
68	Marine wind forecast	http://www.bom.gov.au/marine/wind.shtml	Marine wind forecasts are issued twice daily and provide graphical information on wind direction and speed for different marine areas across Australia.	Y	Web application;

72	Monthly Weather Review	http://www.bom.gov.au/climate/mwr/	A series of documents provide monthly overviews of temperatures, rainfall and significant weather events in Australia.	Y	PDF reports
75	National Atlas of Groundwater Dependent Ecosystems	http://www.bom.gov.au/water/groundwater/gde/index.shtml	The National Atlas of Groundwater Dependent Ecosystems displays ecological and hydrogeological information of known groundwater dependent ecosystems and ecosystems that potentially use groundwater.	Y	Web application;
82	National seasonal temperature outlook	http://www.bom.gov.au/climate/ahead/temps_ahead.shtml	National seasonal temperature outlooks are general statements about the probability or risk of wetter or drier than average weather over a three-month period.	Y	Web application; spatial data download (shape file, KMZ)
85	National Water Account	http://www.bom.gov.au/water/nwa/	The National Water Account provides standardised annual information about the management and use of Australia's water resources, including water stores and flows, water rights and water use.	Y	PDF reports
88	Ocean forecasts: sea temperatures and currents	http://www.bom.gov.au/oceanography/forecasts/	Modelled ocean forecasts provide information on sea temperatures and currents across Australian waters.	Y	Web application;
90	Pacific Climate Change Data Portal	http://www.bom.gov.au/climate/pacific/about-pacific-data-portal.shtml	The Pacific Climate Change Data Portal provides access to raw and homogenised temperature and rainfall data for several countries and islands in the South Pacific region.	Y	Web application; CSV download;
91	Predictive Ocean Atmosphere Model for Australia	http://poama.bom.gov.au/	The Predictive Ocean Atmosphere Model for Australia (POAMA) is a seasonal to inter-annual seasonal forecast modelling system that provides a number of experimental operational products.	Y	Seasonal model; NetCDF/THREDDS access.
93	Rainfall and temperature records	http://www.bom.gov.au/climate/extreme/records.shtml	Rainfall and temperature records provide the highest and lowest temperatures and rainfall levels for each Australian state and territory.	Y	HTML tables; PDF reports
94	Rainfall forecast	http://www.bom.gov.au/jsp/watl/rainfall/pme.jsp	Rainfall forecasts, over a five-day period, are produced through computer modelling and published graphically, for districts around Australia.	Y	Web application;

97	Seasonal climatic outlooks	http://www.bom.gov.au/climate/ahead/	Seasonal climatic outlooks are produced on a three-monthly basis and provide information on rainfall, temperature, El Niño/La Niña status, streamflow, tropical cyclones and climate models.	Y	Information; analysis; links to web applications;
98	Seasonal streamflow forecasts	http://www.bom.gov.au/water/ssf/index.shtml	Streamflow forecasts are presented graphically for north-eastern Australia, providing the likelihood of low, near median or high streamflows over a three-month period.	Y	Web application; CSV download; analysis
102	Southern Oscillation Index	http://www.bom.gov.au/climate/current/soi2.shtml	The Southern Oscillation Index indicates the development and intensity of El Niño or La Niña events in the Pacific Ocean.	Y	Graphs (images); CSV download
103	Spatial sea surface temperature anomaly forecast	http://www.bom.gov.au/oceanography/oceantemp/GBR_SST.shtml	Spatial sea surface temperature anomaly forecasts are plotted for the Great Barrier Reef using the Bureau of Meteorology's seasonal prediction system.	Y	Graphs (images); information
104	Special climate statements	http://www.bom.gov.au/climate/current/statements/	Special climate statements provide a detailed summary of significant (and unusual) weather and climatic events in Australia.	Y	PDF reports
110	Tide predictions for Australia, South Pacific and Antarctica	http://www.bom.gov.au/australia/tides/	This tide predictions website provides estimated low and high tide levels and times for ports within Australia, the South Pacific and Antarctica.	Y	Web application; PDF reports
112	Tropical Cyclone Outlook	http://www.bom.gov.au/nt/forecasts/tcoutlook.shtml	The Tropical Cyclone Outlook is a regular report on existing and potential cyclones in Australia.	Y	Not running (summer only)
114	Ultraviolet index forecast	http://www.bom.gov.au/australia/uv/index.shtml	The National UV Forecast Graph shows the maximum clear sky UV Index at noon for the whole of Australia. The State UV Forecast Map shows UV Index forecasts for specific locations in states and territories.	Y	Web application; images
118	Water storage	http://water.bom.gov.au/waterstorage/awris/	The water storage website (and associated iPhone app) allows users to compare water storage levels and volumes for 250 publically-owned lakes, reservoirs and weirs in states and territories and to see how much water is available in Australia.	Y	Web application;

120	Weather analysis chart archive	http://www.bom.gov.au/australia/charts/archive/index.shtml	The analysis chart archive provides access to weather maps for the Australian, Southeast Asian/Western Pacific and Southern Hemisphere regions.	Y	Images
121	Weather forecasts, warnings and observations	http://www.bom.gov.au/australia/index.shtml	These services encompass a wide range of forecasts, warnings, weather observations, radar images and information services for the Australian continent, Antarctica and the South Ocean.	Y	Landing page.
123	Weekly rainfall update	http://www.bom.gov.au/climate/current/weeklyrain.shtml	The weekly rainfall update provides a detailed analysis and map of rainfall recorded across Australia during the past week.	Y	Information; images
124	Weekly Tropical Climate Note	http://www.bom.gov.au/climate/tropnote/tropnote.shtml	The Weekly Tropical Climate Note is a summary report and forecast of Australia's tropical climate, produced on a weekly basis.	Y	Information (text)
129	Australian Water Availability Project	http://www.eoc.csiro.au/awap/	The Australian Water Availability Project (AWAP) monitors the state and trend of the terrestrial water balance of the Australian continent. The AWAP website provides water balance maps, reports and data files.	Y	Web application; images; NetCDF/Thredds to registered users.
132	MetEye	http://www.bom.gov.au/australia/meteye/	MetEye is an online mapping tool used to visualise weather observations and forecasts for Australia. It presents information on temperature, humidity, rainfall, wind, sea surface temperature, river conditions and cloud cover.	Y	Web application;
138	Climate and past weather information and data services	http://www.bom.gov.au/climate/	This website provides links to information about weather conditions in Australia–past, future and present. It includes links to seasonal outlooks, reports, summaries and maps. Data services provide past weather and climate information from the Bureau's Australian climate data archive.	Y	Landing page.
142	Severe Storms Archive	http://www.bom.gov.au/australia/stormarchive/	The Severe Storms Archive contains data relating to recorded severe thunderstorm and related events in Australia dating back to the 18th Century.	Y	Web application; CSV download;
145	Marine Water Quality Dashboard	http://www.bom.gov.au/marinewaterquality/	The Marine Water Quality Dashboard is a tool to access and visualise a range of water quality indicators for the Great Barrier Reef. It enables access to near real-time data, and over	Y	Web application; CSV, NetCDF, KMZ, GeoTIFF

			ten years of information, on sea surface temperatures, chlorophyll levels, suspended sediments, and coloured dissolved organic matter.		
152	eXchange	http://www.bom.gov.au/environment/publications.shtml	eXchange is an electronic newsletter about environmental information activity with a focus on Australian Government developments, including strategic activities, new and existing products and services and major events.	Y	Newsletters
153	Climate change and variability	http://www.bom.gov.au/climate/change/	This website provides information on the science of climate change and climate variability. It includes climate updates, trend maps and information about relevant datasets.	Y	Information (text, images); educational; analysis
155	Climate maps - Australian Water Availability Project	http://www.bom.gov.au/jsp/awap/	Meteorological analyses for Australia were generated as a contribution to the Australian Water Availability Project. They include rainfall, temperature, vapour pressure, solar exposure and the Normalised Difference Vegetation Index (NDVI) over time periods ranging from daily, weekly, monthly to 3-yearly.	Y	Web application; images; PDF; GRID download.
156	SunSmart app	http://www.bom.gov.au/uv/iphoneapp.shtml	This mobile phone application provides weather, temperature, UV level and sun protection times for the day anywhere in Australia.	Y	Mobile app.
157	Indigenous weather knowledge	http://www.bom.gov.au/iwk/	The indigenous weather knowledge website displays seasonal weather calendars, developed over thousands of years by Indigenous communities.	Y	Information (text); educational;
159	Radio and space weather services	http://www.ips.gov.au/	This website provides the latest space weather observations and forecasts. Users can also subscribe to reports and alerts.	Y	Text, images,
160	Weather Observations Website (WOW)	http://bom-wow.metoffice.gov.uk/	This website provide an online weather community where Australians can share weather observations, information and photos.	Y	Web application; community data.
161	Weather Watch radars	http://www.bom.gov.au/australia/radar/?ref=ftr	The Bureau of Meteorology operates a nationwide network of over 60 Weather Watch radars. This website provides radar locations, images, news and education.	Y	Web application; images
163	Water restrictions	http://www.bom.gov.au/water/restriction	The water restrictions website provides information about current water restrictions around Australia. It is searchable by	Y	Web application; text data.

		s/	State or Territory, water agency and restriction name.		
164	Weather Station Directory	http://www.bom.gov.au/climate/data/stations/	This website provides information about the Bureau of Meteorology weather stations, which can help users to select the type of data they require.	Y	Web application; CSV download;
165	National flood forecasting and warning service	http://www.bom.gov.au/water/floods/	The Bureau of Meteorology flood forecasting and warning service uses rainfall and streamflow observations, numerical weather predictions and hydrologic models to forecast and warn for possible flood events across Australia.	Y	Information; PDFs
166	Maps of average conditions	http://www.bom.gov.au/climate/averages/maps.shtml	These maps show the variability of the Australian climate by providing information about average as well as more extreme conditions.	Y	Maps (images).
167	Current tropical cyclones	http://www.bom.gov.au/cyclone/?ref=dropdown	This website provides information and warnings about current tropical cyclones in Australia and its waters.	Y	Information; images (maps)
168	Pilot heatwave forecast for Australia	http://www.bom.gov.au/australia/heatwave/	This website provides a graphical map of heatwaves, severe heatwaves and extreme heatwaves for the current day extending out for the next four days.	Y	N/A

Summary of Web APIs

Name	**Encodings**	**Hypermedia**	**HTTP verbs**	**Data types**	**Interface description language**	**URL**
NOAA Climate Data API v2	JSON	No	GET	Observations, stations	Text-based	https://www.ncdc.noaa.gov/cdo-web/webservices/v2
UK Met Office - DataPoint	JSON, XML, Images	No	GET	Observations, forecasts, stations, imagery, text	Text-based	http://www.metoffice.gov.uk/datapoint
Atlas of Living Australia	JSON	Yes	GET, POST, DELETE	Observations, species/taxon data, spatial data,	Dynamic HTML	http://api.ala.org.au/
UK Bathing	Linked Data*	Yes	GET	Obserations, spatial	Dynamic linked-data	http://www.epimorphics.com/web/wiki/

water quality				(zones, sampling points), assessments	HTML	bathing-water-quality-structure-published-linked-data
UK National Biodiversity Network	JSON, XML	Unknown	GET, PUT, DELETE	Observations, taxon	WADL	https://data.nbn.org.uk/Documentation/Web_Services/Web_Services-REST/
GeoServices API	JSON	No	GET, POST	Spatial, images	JSON schema + specification	http://portal.opengeospatial.org
Global Change Information System	JSON	Yes	GET	Observations, model outputs, reports, tables..	Swagger (dynamic HTML)	http://data.globalchange.gov/api/
CKAN	JSON (API) + various	Yes	GET, POST, PUT, DELETE, PATCH	Range (generic)	Text-based	http://docs.ckan.org/en/latest/
Data Access Protocol / Hyrax	HTTP Text + binary (octet)	Some	GET	Spatial (grids), observations (tables)	Text-based (specification)	http://www.opendap.org/pdf/ESE-RFC-004v1.2.pdf
THREDDS (multiple services)	XML, NetCDf	Yes	GET	Spatial (grids), observations (tables)	Multiple services (standards: WMS, WCS, NC subsetting etc).	http://www.unidata.ucar.edu/software/thredds/current/tds/
ERDDAP (multiple services)	JSON, XML, NetCDF, CSV, matlab, xhtml	Yes	GET	Spatial (grids), observations (tables)	Text-based	http://coastwatch.pfeg.noaa.gov/erddap/index.html
Forecast.io	JSON	No	GET	Observations, forecasts	Text-based	https://developer.forecast.io/docs/v2
USGS Water Data Services	JSON, CSV, XML	No	GET	Observations, statistics	Text-based	http://waterservices.usgs.gov/

*Linked Data encodings	HTML, CSV, JSON, RDF, text, TTL, XHTML, XML

References

Bechhofer, S., J. Ainsworth, J. Bhagat, I. Buchan, P. Couch, D. Cruickshank, D. De Roure, et al. 2010. “Why Linked Data Is Not Enough for Scientists.” In , 300–307. doi:10.1109/eScience.2010.21.

Bray, Tim. 2016. “The I-JSON Message Format.” https://tools.ietf.org/html/rfc7493.

Brooks, F. P. 1987. *No Silver Bullet*. April. https://books.google.com.au/books?hl=en&lr=&id=eT7uBwAAQBAJ&oi=fnd&pg=PA11&dq=no+silver+bullet&ots=bUfoIuR0Oz&sig=XGUpG1-_9A-XjdaG4X8tqDuoa4k.

Bureau of Meteorology. 2015. “Bureau of Meteorology Strategic Plan 2015-2020.” http://www.bom.gov.au/info/leaflets/strategic-plan-2015-20.pdf.

Cartwright, John, Jesse Varner, and Susan McLean. 2015. “Data Stewardship: How NOAA Delivers Environmental Information for Today and Tomorrow.” *Marine Technology Society Journal* 49 (2): 107–11.

Cornillon, P., J. Gallagher, and T. Sgouros. 2003. “OPeNDAP: Accessing Data in a Distributed, Heterogeneous Environment.” *Data Science Journal* 2: 164–74. doi:10.2481/dsj.2.164.

Cox, Simon JD, Jonathan Yu, and Terry Rankine. 2014. “SISSVoc: A Linked Data API for Access to SKOS Vocabularies.” *Semantic Web* 7 (1): 9–24.

Cox, Simon, and Peter Taylor. 2015. “Observations & Measurements JSON Encoding - Discussion Paper. OGC 15-100r1.” 15-100.

D’Amore, F., S. Cinnirella, and N. Pirrone. 2013. “Data and Metadata Management Automation for an Effective Approach to Sharing Environmental Data.” Edited by N. Pirrone. *Proceedings of the 16th International Conference on Heavy Metals in the Environment* 1: 18003. doi:10.1051/e3sconf/20130118003.

Dierking, Howard. 2016. “Versioning RESTful Services | Howard Dierking.” http://codebetter.com/howarddierking/2012/11/09/versioning-restful-services/.

DiGiuseppe, Nicholas, Line C. Pouchard, and Natalya F. Noy. 2014. “SWEET Ontology Coverage for Earth System Sciences.” *Earth Science Informatics* 7 (4): 249–64.

Dow, Andrew K., Eli M. Dow, Thomas D. Fitzsimmons, and Maurice M. Materise. 2015. “HARNESSING THE ENVIRONMENTAL DATA FLOOD A Comparative Analysis of Hydrologic, Oceanographic, and Meteorological Informatics Platforms.” *Bulletin of the American Meteorological Society* 96 (5): 725–36. doi:10.1175/BAMS-D-13-00178.1.

Elkhatib, Y., G.S. Blair, and B. Surajbali. 2013. “Experiences of Using a Hybrid Cloud to Construct an Environmental Virtual Observatory.” In , 13–18.

Evans, Eric. 2003a. “BOUNDED CONTEXT.” In *Domain-Driven Design: Tackling Complexity in the Heart of Software*. Addison-Wesley Professional. http://proquest.safaribooksonline.com/book/project-management/0321125215/fourteendot-maintaining-model-integrity/ch14lev1sec1.

Evans, Eric. 2003. *Domain-Driven Design: Tackling Complexity in the Heart of Software*. Addison-Wesley Professional. http://proquest.safaribooksonline.com/book/project-management/0321125215.

Fowler, Marting. 2003. “MultipleCanonicalModels.” *Martinfowler.com*. http://martinfowler.com/bliki/MultipleCanonicalModels.html.

Fowler, Marting. 2014. “BoundedContext.” *Martinfowler.com*. http://martinfowler.com/bliki/BoundedContext.html.

Gong, Jianya, Jing Geng, and Zeqiang Chen. 2015. “Real-Time GIS Data Model and Sensor Web Service Platform for Environmental Data Management.” *International Journal of Health Geographics* 14 (1): 2. doi:10.1186/1476-072X-14-2.

Heritage, Laura. 18:07:26 UTC. “API Description Languages.” http://www.slideshare.net/SOA_Software/api-description-languages.

Hilbring, Desiree, Anastasia Moumtzidou, Juergen Mossgraber, and Stefanos Vrochidis. 2014. “Semantically Enriching an Open Source Sensor Observation Service Implementation for Accessing Heterogeneous Environmental Data Sources.” *Transactions in Gis* 18 (4): 480–95. doi:10.1111/tgis.12055.

H, Jeremy. 2011. “There Is No Right Way: Versioning and Types in REST/HTTP API Resources.” *There Is No Right Way*. February 24. http://thereisnorightway.blogspot.com.au/2011/02/versioning-and-types-in-resthttp-api.html.

Horsburgh, Jeffery S., David G. Tarboton, Michael Piasecki, David R. Maidment, Ilya Zaslavsky, David Valentine, and Thomas Whitenack. 2009. “An Integrated System for Publishing Environmental Observations Data.” *Environmental Modelling & Software* 24 (8): 879–88. doi:10.1016/j.envsoft.2009.01.002.

Houbie, Frédéric. 2015. “Testbed 11 REST Interface Engineering Report.” OGC Engineering Report OGC 15-052.

Jacobson, Daniel, Greg Brail, and Dan Woods. 2011a. *APIs: A Strategy Guide*. O’Reilly Media, Inc. http://proquest.safaribooksonline.com/book/information-technology-and-software-development/9781449321628.

Jacobson, Daniel, Greg Brail, and Dan Woods. 2011. “Defining the Value Chain: Ask Key Questions.” In *APIs: A Strategy Guide*. O’Reilly Media, Inc. http://proquest.safaribooksonline.com/book/information-technology-and-software-development/9781449321628/5dot-key-design-principles-for-apis/id2809367.

Jacobson, Daniel, Greg Brail, and Dan Woods. 2011. “Why You Might Need an API.” In *APIs: A Strategy Guide*. O’Reilly Media, Inc. http://proquest.safaribooksonline.com/book/information-technology-and-software-development/9781449321628/2dot-apis-as-a-business-strategy/id2743825.

Jazayeri, Mohammad Ali, Steve H. L. Liang, and Chih-Yuan Huang. 2015. “Implementation and Evaluation of Four Interoperable Open Standards for the Internet of Things.” *Sensors* 15 (9): 24343–73. doi:10.3390/s150924343.

Kelly, Mike. 2015. “JSON Hypertext Application Language.” Accessed December 1. https://tools.ietf.org/html/draft-kelly-json-hal-07.

Klabnik, Steve. 2015. “UBER: A New Hypermedia Format for APIs.” *InfoQ*. Accessed December 1. http://www.infoq.com/news/2014/05/uber-hypermedia.

Kotsev, Alexander, Francesco Pantisano, Sven Schade, and Simon Jirka. 2015. “Architecture of a Service-Enabled Sensing Platform for the Environment.” *Sensors* 15 (2): 4470–95. doi:10.3390/s150204470.

Lanthaler, Markus. 2013. “Creating 3rd Generation Web APIs with Hydra.” In *Proceedings of the 22nd International Conference on World Wide Web Companion*, 35–38. International World Wide Web Conferences Steering Committee. http://dl.acm.org/citation.cfm?id=2487799.

Lanthaler, Markus. 2014. “Leveraging Linked Data to Build Hypermedia-Driven Web APIs.” In *REST: Advanced Research Topics and Practical Applications*, 107–23. Springer. http://link.springer.com/chapter/10.1007/978-1-4614-9299-3_7.

Lanthaler, Markus, and Christian Gütl. 2012. “On Using JSON-LD to Create Evolvable RESTful Services.” In *Proceedings of the Third International Workshop on RESTful Design*, 25–32. WS-REST ’12. New York, NY, USA: ACM. doi:10.1145/2307819.2307827.

Lanthaler, Markus, and Christian Gütl. 2013. “Hydra: A Vocabulary for Hypermedia-Driven Web APIs.” *LDOW* 996. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.362.4758&rep=rep1&type=pdf.

Lanthaler, Markus, and Christian Gütl. 2013. “Model Your Application Domain, Not Your JSON Structures.” In *Proceedings of the 22nd International Conference on World Wide Web Companion*, 1415–20. International World Wide Web Conferences Steering Committee. http://dl.acm.org/citation.cfm?id=2488184.

Lensmar, Ole. 2013. “An Overview of REST Metadata Formats.” *API UX*. http://apiux.com/2013/04/09/rest-metadata-formats/.

Liang, Steve, Chih-Yuan Huang, and Tania Khalafbeigi. 2015. “OGC® SensorThings API - Part 1: Sensing.” Candidate standard 15-078r4. OGC.

Masse, Mark. 2011a. “Identifier Design with URIs.” In *REST API Design Rulebook*. O’Reilly Media, Inc. http://proquest.safaribooksonline.com/book/web-development/9781449317904/2dot-identifier-design-with-uris/id2814545.

O’Reilly. 2011b. “Request Methods.” In *REST API Design Rulebook*. O’Reilly Media, Inc. http://proquest.safaribooksonline.com/book/web-development/9781449317904/2dot-identifier-design-with-uris/id2814545.

O’Reilly. 2011c. “URI Authority Design.” In *REST API Design Rulebook*. O’Reilly Media, Inc. http://proquest.safaribooksonline.com/book/web-development/9781449317904/2dot-identifier-design-with-uris/id2814545.

Ma, Xiaogang, Peter Fox, Curt Tilmes, Katharine Jacobs, and Anne Waple. 2014. “Capturing Provenance of Global Change Information.” *Nature Climate Change* 4 (6): 409–13. doi:10.1038/nclimate2141.

Michener, William K. 2015. “Ecological Data Sharing.” *Ecological Informatics* 29, Part 1 (September): 33–44. doi:10.1016/j.ecoinf.2015.06.010.

National Water Commission. 2006. “Australian Water Resources Information System: User Requirements.”

Nativi, Stefano, Paolo Mazzetti, Mattia Santoro, Fabrizio Papeschi, Max Craglia, and Osamu Ochiai. 2015. “Big Data Challenges in Building the Global Earth Observation System of Systems.” *Environmental Modelling & Software* 68 (June): 1–26. doi:10.1016/j.envsoft.2015.01.017.

Nottingham, Mark, Marc Hadley, Joe Gregorio, David Orchard, and Roy T. Fielding. 2015. “URI Template.” Accessed December 1. https://tools.ietf.org/html/rfc6570.

Overflow, Stack. 2016. “Best Practices for API Versioning?” http://stackoverflow.com/questions/389169/best-practices-for-api-versioning.

Overpeck, Jonathan T., Gerald A. Meehl, Sandrine Bony, and David R. Easterling. 2011. “Climate Data Challenges in the 21st Century.” *Science* 331 (6018): 700–702. doi:10.1126/science.1197869.

Pääkkönen, Pekka, and Daniel Pakkala. 2015. “Reference Architecture and Classification of Technologies, Products and Services for Big Data Systems.” *Big Data Research*. Accessed August 5. doi:10.1016/j.bdr.2015.01.001.

Pautasso, Cesare. 2014. “RESTful Web Services: Principles, Patterns, Emerging Technologies.” In *Web Services Foundations*, edited by Athman Bouguettaya, Quan Z. Sheng, and Florian Daniel, 31–51. Springer New York. http://link.springer.com/chapter/10.1007/978-1-4614-7518-7_2.

Pautasso, Cesare, Olaf Zimmermann, and Frank Leymann. 2008. “Restful Web Services vs. Big’web Services: Making the Right Architectural Decision.” In *Proceedings of the 17th International Conference on World Wide Web*, 805–14. ACM. http://dl.acm.org/citation.cfm?id=1367606.

“PharkMillups/beautiful-Docs.” 2016a. *GitHub*. Accessed February 10. https://github.com/PharkMillups/beautiful-docs.

Pouchard, Line C., Marcia L. Branstetter, Robert B. Cook, Ranjeet Devarakonda, Jim Green, Giri Palanisamy, Paul Alexander, and Natalya F. Noy. 2013. “A Linked Science Investigation: Enhancing Climate Change Data Discovery with Semantic Technologies.” *Earth Science Informatics* 6 (3): 175–85.

Reis, Stefan, Edmund Seto, Amanda Northcross, Nigel W. T. Quinn, Matteo Convertino, Rod L. Jones, Holger R. Maier, et al. 2015. “Integrating Modelling and Smart Sensors for Environmental and Human Health.” *Environmental Modelling & Software*. Accessed July 21. doi:10.1016/j.envsoft.2015.06.003.

Rettig, Andrew J., Sumit Khanna, and Richard A. Beck. 2015. “Open Source REST Services for Environmental Sensor Networking.” *Applied Geography* 60 (June): 294–300. doi:10.1016/j.apgeog.2014.11.003.

Romañach, Stephanie S., Mark McKelvy, Kevin Suir, and Craig Conzelmann. 2015. “EverVIEW: A Visualization Platform for Hydrologic and Earth Science Gridded Data.” *Computers & Geosciences* 76 (March): 88–95. doi:10.1016/j.cageo.2014.12.004.

Sahni, Vinay. 2016. “Best Practices for Designing a Pragmatic RESTful API.” *Vinay Sahni*. http://www.vinaysahni.com/best-practices-for-a-pragmatic-restful-api.

Schaaf, Hylke van der, and Reinhard Herzog. 2015. “Mapping the OGC SensorThings API onto the OpenIoT Middleware.” In *Interoperability and Open-Source Solutions for the Internet of Things*, edited by Ivana Podnar Žarko, Krešimir Pripužić, and Martin Serrano, 62–70. Lecture Notes in Computer Science 9001. Springer International Publishing. http://link.springer.com/chapter/10.1007/978-3-319-16546-2_6.

Schandl, Thomas, and Andreas Blumauer. 2010. “PoolParty: SKOS Thesaurus Management Utilizing Linked Data.” In *The Semantic Web: Research and Applications*, 421–25. Springer. http://link.springer.com/10.1007%2F978-3-642-13489-0_36.

Swain, Nathan R., Kilisimasi Latu, Scott D. Christensen, Norman L. Jones, E. James Nelson, Daniel P. Ames, and Gustavious P. Williams. 2015. “A Review of Open Source Software Solutions for Developing Water Resources Web Applications.” *Environmental Modelling & Software* 67 (May): 108–17. doi:10.1016/j.envsoft.2015.01.014.

Taylor, Peter, Paul Sheahan, David Briar, Stuart Hamilton, Brian Gouge, Peter Heweston, and Matthew Fry. 2015. “WaterML2.0 Part 2 - RESTful API and JSON Encoding.” OGC Discussion Paper 15-033. https://portal.opengeospatial.org/files/?artifact_id=63302&version=1.

Usländer, Thomas. 2005. “Trends of Environmental Information Systems in the Context of the European Water Framework Directive.” *Environmental Modelling & Software*, Environmental Knowledge and Information SystemsEnvironmental Knowledge and Information Systems, 20 (12): 1532–42. doi:10.1016/j.envsoft.2004.09.029.

Usländer, Thomas, Patrick Jacques, Ingo Simonis, and Kym Watson. 2010. “Designing Environmental Software Applications Based upon an Open Sensor Service Architecture.” *Environmental Modelling & Software*, Thematic issue on Sensors and the Environment – Modelling & ICT challenges, 25 (9): 977–87. doi:10.1016/j.envsoft.2010.03.013.

Verborgh, Ruben, Thomas Steiner, Rik Van de Walle, and Joaquim Gabarro. 2012. “Linked Data and Linked APIs: Similarities, Differences, and Challenges.” In *The Semantic Web: ESWC 2012 Satellite Events*, edited by Elena Simperl, Barry Norton, Dunja Mladenic, Emanuele Della Valle, Irini Fundulaki, Alexandre Passant, and Raphaël Troncy, 272–84. Lecture Notes in Computer Science 7540. Springer Berlin Heidelberg. http://link.springer.com/chapter/10.1007/978-3-662-46641-4_20.

Vitolo, C., Y. Elkhatib, D. Reusser, C.J.A. Macleod, and W. Buytaert. 2015. “Web Technologies for Environmental Big Data.” *Environmental Modelling and Software* 63: 185–98. doi:10.1016/j.envsoft.2014.10.007.

Wanner, Leo, Harald Bosch, Nadjet Bouayad-Agha, Gerard Casamayor, Thomas Ertl, Desiree Hilbring, Lasse Johansson, Kostas Karatzas, Ari Karppinen, Ioannis Kompatsiaris, Tarja Koskentalo, Simon Mille, Juergen Mossgraber, et al. 2015. “Getting the Environmental

Information across: From the Web to the User.” *Expert Systems* 32 (3): 405–32. doi:10.1111/exsy.12100.

Wanner, Leo, Harald Bosch, Nadjet Bouayad-Agha, Gerard Casamayor, Thomas Ertl, Désirée Hilbring, Lasse Johansson, Kostas Karatzas, Ari Karppinen, Ioannis Kompatsiaris, Tarja Koskentalo, Simon Mille, Jürgen Moßgraber, et al. 2015. “Getting the Environmental Information across: From the Web to the User.” *Expert Systems* 32 (3): 405–32. doi:10.1111/exsy.12100.

Wilson, Anne, Robert R. Downs, W. Christopher Lenhardt, Carol Meyer, William Michener, Hampapuram Ramapriyan, and Erin Robinson. 2014. “Realizing the Value of a National Asset: Scientific Data.” *Eos, Transactions American Geophysical Union* 95 (50): 477–78. doi:10.1002/2014EO500006.

Wood, Tim. 2012. “How Are REST APIs Versioned?” *Lexical Scope*. http://www.lexicalscope.com/blog/2012/03/12/how-are-rest-apis-versioned/.

Yu, Jonathan, Simon Cox, Gavin Walker, Paul J. Box, and Paul Sheahan. 2011. “Use of Standard Vocabulary Services in Validation of Water Resources Data Described in XML.” *Earth Science Informatics* 4 (3): 125–37.

Yu, Liang, and Yong Liu. 2015. “Using Linked Data in a Heterogeneous Sensor Web: Challenges, Experiments and Lessons Learned.” *International Journal of Digital Earth* 8 (1): 15–35. doi:10.1080/17538947.2013.839007.

Zazueta, Rob. 2015. “The Ultimate Solution to Versioning REST APIs: Content Negotiation.” *The RESTed NARWHL*. June. http://www.narwhl.com/2015/03/the-ultimate-solution-to-versioning-rest-apis-content-negotiation/.

2016. Accessed April 14. http://landing.deloitte.com.au/rs/761-IBL-328/images/TechTrends2016.pdf.

CONTACT US

t 1300 363 400
+61 3 9545 2176
e csiroenquiries@csiro.au
w www.data61.csiro.au

AT CSIRO WE SHAPE THE FUTURE

We do this by using science and technology to solve real issues. Our research makes a difference to industry, people and the planet.

FOR FURTHER INFORMATION

Peter Taylor
Senior Software Engineer
t +61 3 62375617
e peter.taylor@csiro.au
w www.data61.csiro.au