PAS 212:2016

Incorporating Corrigendum No. 1

Automatic resource discovery for the Internet of Things – Specification

HYPERCAT

bsi.

Publishing and copyright information

ISBN 978 0 580 96280 6

ICS 35.080, 35.240.01

Publication history

First published May 2016

Amendments issued since publication

Date	Text affected
30 Nov 2016	Corrigendum No. 1 to add missing] and } brackets to Figure D.4

Contents

List of tables

Foreword

This PAS was sponsored by Flexeye with funding from Innovate UK. Its development was facilitated by BSI Standards Limited and it was published under licence from The British Standards Institution. It came into effect on 31 May 2016.

Acknowledgement is given to Pilgrim Beart of DevicePilot as the technical author of this PAS and to Toby Jaffey and John Davies as authors of the original specification on which this PAS is based. Acknowledgement is also given to the following organizations that were involved in the development of this PAS as members of the steering group:

- British Telecommunications plc
- BSI Consumer & Public Interest Network
- Centre for the Protection of National Infrastructure (CPNI)
- DevicePilot
- Flexeye
- Greater London Authority
- Hypercat
- IBM UK Ltd
- In Touch Ltd
- King's College London
- Knowledge Transfer Network
- RedBite Solutions Ltd
- Thingful
- Co-opted member

Acknowledgement is also given to the members of a wider review panel who were consulted in the development of this PAS.

This PAS is not to be regarded as a British Standard. It will be withdrawn upon publication of its content in, or as, a British Standard.

The PAS process enables a specification to be rapidly developed in order to fulfil an immediate need in industry. A PAS can be considered for further development as a British Standard, or constitute part of the UK input into the development of a European or International Standard.

Relationship with other publications

This PAS is based on a specification, known as Hypercat, which was originally developed by the Hypercat working group.

Use of this document

It has been assumed in the preparation of this PAS that the execution of its provisions will be entrusted to appropriately qualified and experienced people, for whose use it has been produced.

Presentational conventions

The provisions of this PAS are presented in roman (i.e. upright) type. Its requirements are expressed in sentences in which the principal auxiliary verb is "shall". The word "should" is used to express recommendations, the word "may" is used to express permissibility and the word "can" is used to express possibility, e.g. the consequence of an action of an event.

Commentary, explanation and general informative material is presented in italic type, and does not constitute a normative element.

Spelling conforms to The Shorter Oxford English Dictionary. If a word has more than one spelling, the first spelling in the dictionary is used.

Contractual and legal considerations

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

Compliance with a PAS cannot confer immunity from legal obligations.

0 Introduction

0.1 Motivation

There is an increasing need to share data easily, driven by innovations such as the Internet of Things (IoT), Smart Cities and big data, within specific sectors of the economy such as healthcare, automotive and energy, and across the World Wide Web in general. Some of this data can be made fully public, whilst access to other data needs to be controlled.

If there is a need for data-consuming clients to be hand-coded to access each specific data server, then software engineering becomes a resource constraint which inhibits exponential growth in the number and combination of such clients and servers. PAS 212 aims to address this issue by specifying a means to automate the discovery of such data resources, without either the client or server having to be written to be explicitly compatible with each other.

PAS 212 specifies a common catalogue format that clients can use to discover data in servers that they can use. It describes an open, lightweight JSON-based hypermedia catalogue format for exposing collections of uniform resource identifiers (URIs). Each catalogue may expose any number of URIs, each with any number of resource description framework (RDF)-like triple statements about it. This allows developers to publish linked-data descriptions of resources.

This PAS therefore allows a server to provide a set of resources to a client, each with a set of semantic annotations (metadata). Implementers are free to choose or invent any set of annotations to suit their needs. Where implementers choose similar or overlapping semantics, the possibilities for interoperability are increased.

This PAS intentionally does not set out to solve all the challenges of data interoperability, but only to address the problem of resource discovery and provide a framework in which other interoperability challenges, such as ontologies, monetization and privacy, may be worked out.

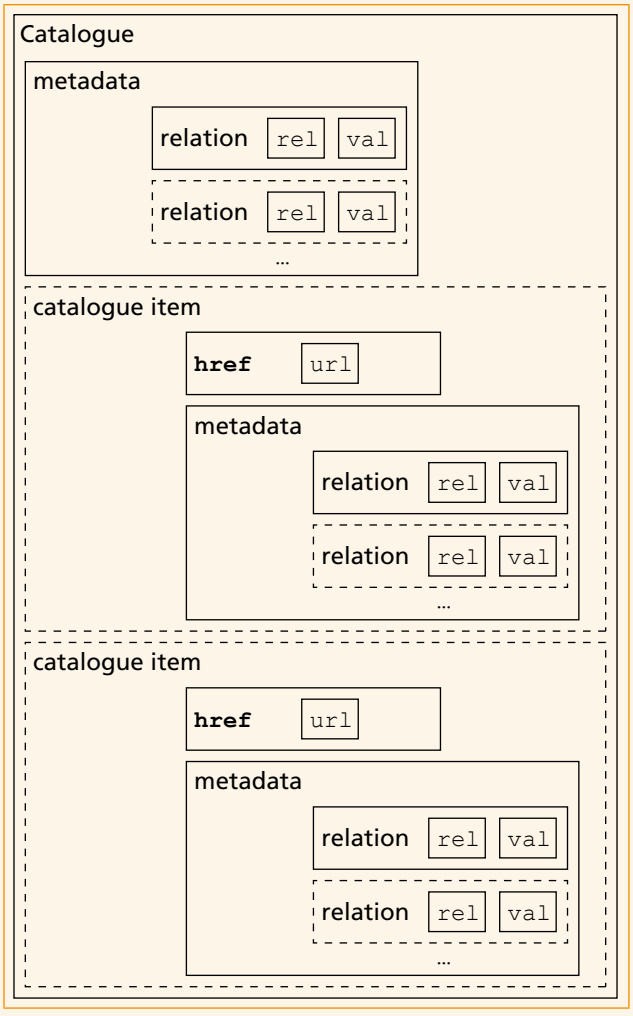
0.2 Structure of a catalogue

Figure 1 shows the structure of the catalogue format described in this PAS. Dotted lines indicate zero or more optional items. Ellipses (…) indicate optional repetition.

A catalogue contains its metadata, and also zero or more catalogue items. Each catalogue item consists of an `href` pointer to the item and metadata about the item.

Metadata consists of a list of relations. Each relation is a `rel-val` pair.

Figure 1 – The structure of the catalogue format



1 Scope

This PAS specifies a protocol whereby any compliant software client can automatically discover data that is stored within any compliant software server, without either the client or server having to be written to have been explicitly compatible with each other.

It applies to the design of services for IoT and the World Wide Web in general, and in particular to the design of applications intended to operate within broad ecosystems such as smart cities, as well as specific industry sectors. It aims to break down the vertically-integrated software silos that have previously existed within the IoT industry.

More specifically it covers the format for representing a catalogue of linked-data resources, annotated with metadata.

It also provides conditional requirements for catalogue access in the following areas:

- catalogue transport;
- security mechanisms to protect access and to prove provenance;
- search functions;
- subscription mechanisms;
- well-known entry-points;
- machine-readable hints to ease usability.

NOTE 1 *Clauses **5** to **8** provide conditional requirements, which means that they might not be relevant, but if they are, then they need to be implemented as specified in these clauses.*

It does not cover implementation of the linked-data resource services themselves.

This PAS is for use by software engineers for IoT (or web services more generally), who are seeking to:

- write a software interface for a client that does not need to be re-written every time it is used with a new server; and
- write a software interface for a server that does not need to be re-written every time it is used with a new client.

Use of this PAS therefore seeks to solve the current problem that lack of interoperability is preventing the exponential growth in the number and combination of such clients and servers.

This PAS is also for use by commissioners of software projects, who, by recommending compliance to this specification, can promote open interoperability between the project parts, and thus avoid vendor lock-in.

This PAS allows and encourages the use of existing ontologies, schemas, etc. by reference.

NOTE 2 *Attention is drawn to the importance of adopting a security-minded approach, further details of which can be found in Annex A.*

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

[NR1] FIELDING, R. *Relative Uniform Resource Locators.* RFC 1808, June 1995. Available from: https://tools.ietf.org/html/rfc1808 [viewed May 2016]

[NR2] FREED, N., BORENSTEIN, N. *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*. RFC 2046, November 1996. Available from: https://www.ietf.org/rfc/rfc2046.txt [viewed May 2016]

[NR3] FRANKS, J., HALLAM-BAKER, P., HOSTETLER, J., LAWRENCE, S., LEACH, P., LUOTONEN, A., STEWART, L. *HTTP Authentication: Basic and Digest Access Authentication*. RFC 2617, June 1999. Available from: https://www.ietf.org/rfc/rfc2617.txt [viewed May 2016]

[NR4] BRICKLEY, D., GUHA, R.V. *RDF Schema 1.1, W3C Recommendation 25 February 2014*. W3C, 2014. Available from: http://www.w3.org/TR/rdf-schema/ [viewed May 2016]

3 Terms, definitions and abbreviations

3.1 Terms and definitions

For the purposes of this PAS, the following terms and definitions apply.

3.1.1 body

content part of a message

3.1.2 catalogue

data structure in the format described by this PAS

3.1.3 client

online computer system which originates requests (**3.1.10**) to a server (**3.1.12**)

3.1.4 `href`

uniform resource identifier (URI)

NOTE *An* `href` *may be a URL (commonly known as a "web link") or a URN.*

3.1.5 metadata

3.1.5.1 catalogue metadata

information pertaining to a catalogue (**3.1.2**)

NOTE *Metadata in this PAS is held in structured form and is a list of relations.*

3.1.5.2 item metadata

information pertaining to a catalogue item

NOTE *Metadata in this PAS is held in structured form and is a list of relations.*

3.1.6 parameter

argument of a query (**3.1.8**)

3.1.7 prefix

string that comes at the beginning of another string

3.1.8 query

`GET` request which is qualified by some parameters in the URL to modify the response

3.1.9 relation

instance of metadata (**3.1.5**), a single key/value pair

3.1.10 request

API call made to a catalogue server

3.1.11 resource URI

value of the `href` parameter in a catalogue item

3.1.12 server

online computer system which responds to requests (**3.1.10**) from clients (**3.1.3**)

3.2 Abbreviations

For the purposes of this PAS the following abbreviations apply:

API	application programming interface
ASCII	American standard code for information interchange
CoAP	constrained application protocol
CORS	cross-origin resource sharing
HTTP	hypertext transfer protocol
HTTP(S)	either HTTP or secure HTTP
IoT	internet of things
JSON	JavaScript object notation
JOSE	JavaScript object signing and encryption
JWS	JSON web signature
MIME	multipurpose internet mail extensions
MQTT	OASIS messaging standard, formerly known as message queuing telemetry transport
RDF	resource description framework
RDFS	RDF schema (a language for representing simple RDF vocabularies)
REST	representational state transfer
RFC	request for comments
RSS	really simple syndication
SSL	secure sockets layer
TCP	transmission control protocol
TLS	transport layer security
UDP	user datagram protocol
URI	uniform resource identifier
URL	uniform resource locator
URN	uniform resource name
UTC	coordinated universal time
XML	extensible markup language
XMPP	extensible messaging and presence protocol

4 Catalogue format

*COMMENTARY ON **4***

General

A catalogue represents an unordered collection of resources on the web. Each item in a catalogue refers to a single resource by its URI, which might itself be a further catalogue. All URNs relating to this PAS are in the `X-hypercat` *namespace and therefore begin with the string* `"urn:X-hypercat"`.

MIME Type

A catalogue is represented by a JSON document of multipurpose internet mail extensions (MIME) type `application/vnd.hypercat.catalogue+json` *containing a single catalogue JSON object.*

Metadata

Metadata about catalogues or catalogue items is defined in the catalogue using a set containing zero or more `rel-val` *pairs. For example, the following relation is interpreted as saying that the parent object is red in colour:* `{"rel":"urn:X-hypercat:rels:hasColour","val":"red"}`.

Extensibility

New forms of metadata may be added by defining new metadata relations. A client encountering an unknown metadata relation should ignore it. Servers and clients are free to support different sets of metadata relations.

For example, indication of a new search method supported by a catalogue server may be added, discovered and identified by defining a new value for the relation `urn:X-hypercat:rels:supportsSearch`.

A new language or style for human-readable descriptions may be added, discovered and identified by defining a new `urn:X-hypercat:rels:hasDescription` *variant (e.g. German* `urn:X-hypercat:rels:hasDescription:de`*).*

A complete replacement catalogue format may be implemented by declaring a new MIME type for `urn:X-hypercat:rels:isContentType`. *Catalogues conforming to this PAS may therefore point to other catalogues conforming to a future version and vice versa, without version ambiguity.*

*In addition to the properties and object structures specified in Clause **4**, a catalogue may also contain any number of other properties and objects as creators see fit.*

It is recommended that all metadata extensions be confined to defining new valid `rel-val` *data pairs.*

4.1 Catalogues

4.1.1 A catalogue shall provide metadata describing itself.

4.1.2 A catalogue shall provide metadata describing each of the catalogue items contained within it.

4.1.3 All resource URIs in a catalogue shall be unique within the catalogue.

***NOTE** This enables each resource URI to be referred to individually.*

4.1.4 The same resource shall not appear more than once in a single catalogue's items.

***NOTE** This is the case even if the entries have different metadata.*

4.2 Catalogue contents

A catalogue shall contain all of the properties specified in Table 1.

***NOTE** A catalogue is a JSON object.*

Table 1 – Required catalogue properties

Property name	Meaning	Property value
`items`	List of items	JSON array of zero or more catalogue items
`catalogue-metadata`	An array of relations describing the catalogue	JSON array of relations

4.3 Catalogue item

4.3.1 A catalogue item shall contain all of the properties specified in Table 2.

NOTE 1 *A catalogue item (from the catalogue's* `items` *array) is a JSON object.*

NOTE 2 *See BS ISO/IEC 11179-3:2013 for further information on metadata registries.*

Table 2 – Required item properties

Property name	Meaning	Property value
`href`	Identifier for the resource item	URI as a JSON string
`item-metadata`	An array of relations describing the resource item	JSON array of relations

4.3.2 The `catalogue-metadata` and `item-metadata` arrays shall contain at least one relation for each of the required relations set out in Tables 4 and 5.

NOTE *The* `catalogue-metadata` *and* `item-metadata` *arrays can contain multiple relations with the same* `rel` *(and* `val`*) properties (they are bags/multisets of features).*

4.4 Relation

All relations shall include all of the properties specified in Table 3.

Table 3 – Required metadata relations

`rel` value	Meaning	`val` value
`urn:X-hypercat:rels:hasDescription:en`	Resource has a human-readable description, e.g. in English	Description as a JSON string
NOTE *In the relation above, ":en" is used to denote that the description is in English. For other languages, the two-letter country code set out in BS EN ISO 3166-1:2014 should be used.*		

Table 4 – Required metadata relations

`rel` value	Meaning	`val` value
`urn:X-hypercat:rels:isContentType`	Data provided by resource is of given type	`application/vnd.hypercat.catalogue+json`

Table 5 – Required metadata properties

Property name	Meaning	Property value
`rel`	A relation between the parent object and a target noun, expressed as a predicate (verb)	URI of a relation as a JSON string
`val`	Value of the `rel` property	JSON string (optionally URI of concept or entity). Relative links shall be interpreted as in RFC 1808 [NR1]

4.5 Metadata

4.5.1 All `catalogue-metadata` and `item-metadata` objects shall include all of the relations specified in Table 4.

NOTE *An example of the minimum valid catalogue is given in Annex B.*

4.5.2 All `catalogue-metadata` objects describing the catalogue shall include all of the relations specified in Table 5.

4.5.3 Where a catalogue publisher wishes to include further metadata, the relations, if used, shall be as specified in Tables 6 and 7.

Table 6 – Optional catalogue relations

`rel` value	Meaning	`val` value
`urn:X-hypercat:rels:hasHomepage`	A reference to a human-readable web page concerning the resource	URL as a JSON string
`urn:X-hypercat:rels:containsContentType`	This catalogue contains resources of a given content type. Only meaningful for relations contained by or pointing to catalogues	RFC 2046 [NR2] MIME type as JSON string (e.g. `"text/csv"`)
`urn:X-hypercat:rels:supportsSearch`	This catalogue's server supports a search mechanism. Only meaningful for relations contained by or pointing to catalogues	URI of a defined search mechanism as a JSON string (see Clause **6**)

Table 7 – Optional item relations

`rel` value	Meaning	`val` value
`urn:X-hypercat:rels:isContentType`	Data provided by resource is of given MIME type	RFC 2046 [NR2] MIME type as JSON string (e.g. `"text/csv"`)
`urn:X-hypercat:rels:hasHomepage`	A reference to a human-readable web page concerning the resource	URL as a JSON string
`urn:X-hypercat:rels:containsContentType`	This catalogue contains resources of a given content type. Only meaningful for relations contained by or pointing to catalogues	RFC 2046 [NR2] MIME type as JSON string (e.g. `"text/csv"`)
`urn:X-hypercat:rels:supportsSearch`	This catalogue's server supports a search mechanism. Only meaningful for relations contained by or pointing to catalogues	URI of a defined search mechanism as a JSON string (see Clause **6**)

5 Server API

5.1 General

Where an HTTP(S) application programming interface (API) is provided to allow clients to interact with catalogues, the server shall conform to the requirements set out in **5.2** to **5.7**.

5.2 Endpoint

Every server shall provide a publicly readable `/cat` endpoint serving a catalogue.

NOTE 1 *All other API operations are optional.*

NOTE 2 *This PAS specifies operations to create, read, update and delete individual items within a catalogue. Operations to create, update and delete entire catalogues are not covered in this PAS and are considered out of scope.*

5.3 Read catalogue

5.3.1 Where a client wishes to read an entire catalogue, the client shall `GET` the catalogue URL.

5.3.2 Where a server successfully serves an entire catalogue, the server shall respond with:

- an HTTP 200 status code; and
- the catalogue.

NOTE *Reading only part of a catalogue is accomplished using one of the defined search mechanisms such as simplesearch, geosearch, lexsearch, etc. (see Clause **6**).*

5.4 Create catalogue item

5.4.1 Where a client wishes to add a new item to a catalogue, the client shall `POST` a catalogue item (JSON) to a catalogue URL.

NOTE *The server may place the new catalogue item in any catalogue it chooses, allowing it to organize "uploaded" resource catalogues in any pattern it wishes.*

5.4.2 On successful creation of a new catalogue item, the server shall respond with:

- an HTTP 201 status code; and
- an HTTP location header with the URL of the catalogue to which the item was added.

NOTE *Creating an entire new catalogue is out of scope of this PAS.*

5.4.3 Where an item already exists in the catalogue, the server shall replace the existing item with the new item and respond with an HTTP 200 status code.

5.5 Update catalogue item

5.5.1 Where a client wishes to replace an existing item in a catalogue, the client shall:

- `PUT` or `POST` a catalogue item to a catalogue URL; and
- specify which existing item is to be replaced by providing a query parameter of `href` with a value corresponding to a resource URI held in the catalogue (URL-encoded).

5.5.2 On successful replacement of the catalogue item, the server shall respond with an HTTP 200 status code.

NOTE 1 *On successful replacement of the catalogue item, the server may also respond with the catalogue item.*

NOTE 2 *Updating multiple catalogue items or an entire catalogue object in a single operation are out of scope of this PAS.*

NOTE 3 *A `POST` operation can create or update a catalogue item, whereas a `PUT` only updates.*

5.6 Delete catalogue item

5.6.1 Where a client wishes to delete an item from a catalogue, the client shall `DELETE` that item from a catalogue URL.

5.6.2 To identify the item to be deleted, the client shall use a query parameter of `href` with a value corresponding to a resource URI held in the catalogue.

NOTE *All URIs passed in by the client as query parameters should be URL-encoded.*

5.6.3 On successful deletion of a catalogue item, the server shall respond with an HTTP 200 status code.

NOTE *Deleting an entire catalogue, as opposed to individual items within a catalogue, is out of scope of this PAS.*

5.7 HTTP status codes

In response to all HTTP requests from a client, a server shall return a valid response, including an appropriate status code, as specified in Table 8.

Table 8 – Status codes

Code	Meaning	Notes
200	No error	
201	Successful creation	
204	No response	
400	Bad request	For example, malformed input
401	Unauthorized	
404	Not found	
409	Conflict	For example, insert existing `href`
501	Not implemented	

6 Search extensions

6.1 Catalogue simple search

*COMMENTARY ON **6.1***

Catalogues may be searched (filtered) to find items matching a specified set of metadata.

If a catalogue provides a search capability, it can advertise this to a client by defining a `"urn:X-hypercat:rels:supportsSearch"` metadata relation.

Simple search is a catalogue extension allowing the search or filtering of a catalogue according to specified parameters.

6.1.1 Where a catalogue server advertises that it supports the simple search mechanism, it shall provide the metadata relation `{"rel":"urn:X-hypercat:rels:supportsSearch", "val":"urn:X-hypercat:search:simple"}` in a catalogue's `catalogue-metadata` array.

***NOTE 1** A simple search only searches a single catalogue resource. It does not include other linked or nested catalogues.*

***NOTE 2** An example catalogue containing two items is given in Annex C.*

6.1.2 Where query parameters are included, a client shall URL-encode them.

***NOTE** The inclusion of query parameters is optional. Optional query parameters are given in Table 9.*

Table 9 – Simple search parameters

Parameter	Meaning	Allowed value
`rel`	Any metadata relation	URI as a JSON string
`val`	Any metadata value	JSON string
`href`	A resource URI	URI as a JSON string

6.1.3 If multiple search parameters are supplied, the server shall respond with the intersection of items where the search parameters match in a single item, combining the parameters with boolean AND.

***NOTE** A simple search is performed by providing a query string [1] to a catalogue.*

6.1.4 The server shall respond with a catalogue containing zero or more items.

6.2 Catalogue prefix match search

*COMMENTARY ON **6.2***

***6.2** describes a catalogue search extension allowing for the search and retrieval of catalogue items with metadata `rel`, `val` or `href` matching a specified prefix.*

6.2.1 Metadata

Where a served catalogue supports prefix match search, the `catalogue-metadata` section shall include:`{ "rel": "urn:X-hypercat:rels:supportsSearch", "val": "urn:X-hypercat:search:prefix" }`.

***NOTE** On seeing this relation defined with this value, a client may then provide the query parameters set out in Table 10 to execute a prefix match search.*

6.2.2 Query parameters

6.2.2.1 A prefix match search is performed by providing a query string to a catalogue. Where multiple search parameters are supplied, the server shall respond with the intersection of matched items, combining the parameters with a boolean AND.

***NOTE** A prefix match search only searches a single catalogue resource. It does not include other linked or nested catalogues.*

6.2.2.2 Where a client includes a query parameter, the query parameters specified in Table 10 shall be URL-encoded.

***NOTE 1** The inclusion of all query parameters is optional.*

***NOTE 2** The examples set out in Table 11 illustrate the operation of the string matcher.*

Table 10 – Prefix search

Parameter	Meaning	Allowed value
`prefix-rel`	Any metadata relation	JSON string
`prefix-val`	Any metadata value	JSON string
`prefix-href`	A resource URI	JSON string

Table 11 – Prefix search examples

Needle (in query)	Haystack (in catalogue)	Matches
`foo`	`foobarbaz`	Yes
`bar`	`foobarbaz`	No
`foobar`	`foobarbaz`	Yes
`foobarbaz`	`foobarbaz`	Yes
`xfoo`	`foobarbaz`	No

6.3 Catalogue lexicographic range search

*COMMENTARY ON **6.3***

General

***6.3** describes a catalogue search extension allowing for the search and retrieval of catalogue items with metadata values within a given lexicographic range. The mechanism is designed to be general purpose, but is inspired by the need to respond with catalogue items within a given time or date range. An example time-based `rel` is set out in Table 12 for use with the search mechanism.*

Lexicographic range search

A lexicographic range search allows searching for items which, when sorted lexicographically, fall between a minimum and maximum value.

The primary intended use for this search mechanism is to respond with items with `rel`s in a particular time range. Catalogue `val`s are always strings, so time-based metadata is always represented as a string. When sorting and searching, it is critical that time be represented with a lexicographically-orderable time string (see BS ISO 8601) with coordinated universal time (UTC) and 'Z' suffix for the timezone.

6.3.1 Catalogue metadata for lexicographic range search

Where a served catalogue supports lexicographic range search, the `catalogue-metadata` section shall include the following: `{ "rel": "urn:X-hypercat:rels:supportsSearch", "val": "urn:X-hypercat:search:lexrange" }`.

***NOTE** On seeing this relation defined with this value in a catalogue, a client may then provide the query parameters set out in Table 12 to execute a lexicographic search.*

6.3.2 Example item metadata for lexicographic range search

Where a catalogue creator wishes it to be possible to use lexicographic search with dates and times in a catalogue, the creator shall represent them using a format which can be sorted as a string.

6.3.3 Query parameters for lexicographic range search

Where a client requests a lexicographic range search, the client shall include the query parameters specified in Table 13.

***NOTE 1** For example: ?lexrange-rel=urn:X-hypercat:rels:lastUpdated&lexrange-min=2007-03-01T13:00:00Z&lexrange-max=2007-04-02T12:07:41Z.*

***NOTE 2** As with other search mechanisms, the server responds with a complete valid catalogue containing only items matching the search criteria.*

Table 12 – Examples of item relations to enable lexicographic search

`rel`	Meaning	Example `val`
`urn:X-hypercat:rels:lastUpdated`	BS ISO 8601 UTC timedate string of last known update to resource	`2007-03-01T13:00:00Z`
`urn:X-space:rels:launchDate`	Just a date (note that in order for lexicographic search to work, the most-significant part of the date needs to be the first characters in the string)	`2014/10/29`

Table 13 – Lexicographic range search parameters

Parameter name	Meaning	Example query value
`lexrange-rel`	Specifies the `rel` to search on	`urn:X-hypercat:rels:lastUpdated`
`lexrange-min`	Lower bound of range to respond with (inclusive)	`2007-03-01T13:00:00Z`
`lexrange-max`	Upper bound of range to respond with (non-inclusive)	`2007-04-02T12:07:41Z`

6.4 Catalogue geographic bounding box search

*COMMENTARY ON **6.4***

***6.4** describes a catalogue search extension allowing for the search and retrieval of catalogue items with longitude and latitude within a 2D bounding box. The mechanism relies on position data being encoded with the geo-coding format WGS84 [1].*

See BS EN ISO 19115-1:2014 for an alternative approach to geographic search.

6.4.1 Geographic search

*COMMENTARY ON **6.4.1***

A geographic search allows for the filtering of items which fall within a geographic region, defined by a bounding box. While a simple bounding box is insufficient for some applications, clients of the API are free to then further refine the search results by other geographic criteria (for example a bounding circle).

Where the creator of a catalogue wishes it to be possible to search catalogue items using the method set out in the Commentary to **6.4**, longitude and latitude shall be encoded with the metadata specified in Table 14.

6.4.2 Catalogue metadata for geographic search

Where a served catalogue supports geographic bounding box search, the catalogue metadata section shall include the following: `{ "rel": "urn:X-hypercat:rels:supportsSearch", "val": "urn:X-hypercat:search:geobound" }`.

***NOTE** On seeing this relation defined with this value in a catalogue, a client may then provide the query parameters set out in **6.5** to execute a geographic bounding box search.*

Table 14 – Item relations to allow geographic search

`rel`	Meaning	Example `val`
`http://www.w3.org/2003/01/geo/wgs84_pos#lat`	WGS84 Latitude	`51.508775`
`http://www.w3.org/2003/01/geo/wgs84_pos#long`	WGS84 Longitude	`-0.116993`

Table 15 – Geographic search query parameters

Parameter name	Meaning
`geobound-minlat`	Inclusive lower bound of latitude of bounding box
`geobound-maxlat`	Inclusive upper bound of latitude of bounding box
`geobound-minlong`	Inclusive lower bound of longitude of bounding box
`geobound-maxlong`	Inclusive upper bound of longitude of bounding box

[1] See https://web.archive.org/web/20120402143802/https://www1.nga.mil/ProductsServices/GeodesyandGeophysics/WorldGeodeticSystem/Pages/default.aspx

6.5 Query parameters for geographic search

Where a client requests a geographic search, the client shall include the query parameters set out in Table 15.

NOTE As with other search mechanisms, the server responds with a complete valid catalogue containing only the items that match the search criteria.

6.6 Catalogue multi-search

COMMENTARY ON ***6.6***

General

This PAS supports several different search extensions covering domains such as time, location and string matching. However, these search extensions each allow only simple interactions with a catalogue. Clients may wish to search according to multiple criteria, for example by "geographic bounding box" and "item creation date". A client can submit two independent queries, then produce the intersection of the queries itself, but this is inefficient, requiring more data to be transmitted to the client than it needs. For large datasets, it may be impractical.

Multi-search allows a client to combine single or multiple search mechanisms supported by a server to produce a catalogue containing only the items of interest.

Multi-search JSON object

Multi-search accepts a single JSON object. The simplest search contains a single `query` *parameter, specifying the URL-encoded parameters for the query (see Table 16).*

As shown in Annex D, searches may be nested to allow complex mixing of union and intersection.

As with other search mechanisms, the server responds with a complete valid catalogue containing only items matching the search criteria.

Where a served catalogue supports multi-search, the `catalogue-metadata` section shall include the following: `{ "rel": "urn:X-hypercat:rels:supportsSearch", "val": "urn:X-hypercat:search:multi"}`.

NOTE 1 *On seeing this relation defined with this value, a client may then execute a search by* `GET`*ting a JSON object from the catalogue URL with the URL parameter* `multi`*. For example, the catalogue http://example.org/cat is multi-searched by* `GET`*ting from http://example.org/cat?multi.*

For example, to request the result of the multi-search query {"query":"?rel=A"}, the following request would be made: `GET` *http://example.org/cat?multi=%7B%22query%22%3A%22%3Frel%3DA%22%7D.*

Where the multi-search object is too large to be passed as a URL query parameter, the `POST` *mechanism may be used. To avoid very long URLs for complex queries, it is optional for a client to submit the query as a* `POST` *instead of the required* `GET`*. It is recommended that servers support such* `POST` *queries.*

NOTE 2 *Examples of search queries are given in Annex D.*

Table 16 – Multi-search parameters

Property name	Property value	Example
`query`	JSON string, holding URL-encoded query string as passed to underlying search mechanism	`"?rel=A"` (for use of simpleSearch extension)
`intersection`	JSON array of objects, containing `query`, `intersection` or `union`.	`"intersection": [ {"query": "?rel=A"}, {"query": "?rel=B"} ]`
`union`	JSON array of objects containing `query`, `intersection` or `union`.	`"union": [ {"query": "?rel=C"}, { "query": "?val=D"} ]`
NOTE *"Intersection" means logical AND, and "union" means logical OR.*		

7 Security extensions

*COMMENTARY ON **7***

This PAS allows users to make use of web technologies to follow best practice for security.

The contents of a catalogue provided by a server to individual clients may vary according to with business (commercial), privacy and security needs of the catalogue custodian/publisher, and the data owner/originator. This may include discovering the existence of the catalogue itself.

7.1 Authentication

*COMMENTARY ON **7.1***

All HTTP(S) requests may be authenticated with a key.

Where keys are presented, a client shall present them in one of two ways. Either:

- basic-auth (RFC 2617 [NR3]) is used, with the key as username and no password provided. Such a header is constructed as follows:
  `"Authorization: " + base64(KEY + ':' + '');` or
- a key is passed in an x-api-key header, constructed as follows:
  `"x-api-key: " + KEY`

where + denotes string concatenation.

***NOTE** How keys are generated, distributed and mapped to permissions in a server is implementation-specific and out of scope of this PAS.*

7.2 Catalogue signing

*COMMENTARY ON **7.2***

General

***7.2** describes a secure signing system for catalogues, using JSON Web Signature (JWS) (RFC 7515 [2] with a custom serialization of the catalogue. This may be used, for example, to establish provenance or authenticate changes.*

Signing catalogues

Digital signatures may be used to sign either individual items, the whole catalogue, or both:

- *for individual items contained in a catalogue (`items` array) a signature may be added to the `item-metadata` for the item. In this case, the digest should cover all of the items (`href` and `item-metadata`).*
- *for an entire catalogue, the signature may be added to the `catalogue-metadata`, with the digest covering both `catalogue-metadata` and all of `item-metadata`.*

Other signing techniques may also be used.

JSON Web signature (JWS)

Where a client chooses to sign a catalogue, it is recommended that a JWS is used, although any other signing mechanism is valid.

JWS allows for the digital signing of content using JSON data structures. A signed document comprises three parts: a JavaScript object signing and encryption (JOSE) header, a JWS payload and a JWS signature.

JOSE header

The JOSE header declares the algorithm used to sign the payload in JSON.

JWS payload

The JWS payload to be signed is an ordered list of octets, typically a string. When generating and comparing the signed digest hash of a document, ordering is critical. JSON specifies a string serialization (`JSON.stringify`), however this does not guarantee the ordering of properties within objects.

In order to produce a stable string representation of a catalogue item, a deterministic `JSON.stringify` is needed by producers and verifiers [2)].

JWS signature

The JWS signed hash is a base64 encoded string [3] representing the hash digest signed with the chosen algorithm. A per-item, catalogue `rel` is used for this purpose.

7.2.1 Where a server uses JWS to sign a catalogue, it shall use the `rel`s specified in Table 17.

7.2.2 All keys in JSON objects shall be sorted ascendingly according to their American standard code for information interchange (ASCII) value as performed by `json-stable-stringify`.

***NOTE** Examples of an entire catalogue signature and a per-item signature are given in Annex E.*

7.2.3 In order to verify a signature, a public key is required. This key shall be specified in PEM (Privacy Enhanced Mail) format in a `val`, as follows:

```
{ rel: "urn:X-hypercat:rels:jws:key", val:
"-----BEGIN RSA PUBLIC KEY-----…" }
```

***NOTE** PEM is a base64 translation of the x509 ASN.1 keys (see RFCs 1421 [4], RFC 1422 [5], RFC 1423 [6] and RFC 1424 [7]).*

[2)] For example, https://github.com/substack/json-stable-stringify

Table 17 – JWS relations

Part	rel	val
JOSE header	urn:X-hypercat:rels:jws:alg	RS256
JWS signature	urn:X-hypercat:rels:jws:signature	YOWPexyGHKu4T_1M_ vtlEnNlqmFOclqp4Hy6hVHfFT4

7.2.4 To verify a catalogue, the signature is generated and compared. When generating the signature, the `rel`s related to signatures shall not be included in the digest.

7.3 Catalogue security access hints

*COMMENTARY ON **7.3***

In order to build a successful ecosystem, systems implementing this PAS should provide open data with traversable links whenever possible. However, many systems wish to provide resources or catalogues only to authenticated clients. Sometimes, these need to be entirely obscured from view. However, where they are discoverable (but not accessible) without authentication, clients require information about how to authenticate for access to the linked resource.

An example of providing an access hint is given in Annex F.

7.3.1 Access metadata

7.3.1.1 Where a client wishes to authenticate for access to a linked resource, the client shall check for the existence of a `rel` in the per-item `item-metadata` object, as specified in Table 18.

Table 18 – Security access relation

rel	val
urn:X-hypercat:rels:accessHint	URI

7.3.1.2 Where the val is a URL, it shall point at a machine- or human-readable description of the authentication method required.

***NOTE** Some catalogues may choose to use well-defined uniform resource names (URNs) to signal a key, or protocol required for authentication.*

7.3.1.3 Where multiple `accessHint` declarations are present, the client shall attempt access using any or all of the hinted mechanisms.

7.4 Catalogue security credential acquisition

*COMMENTARY ON **7.4***

General

*Access to a catalogue may be controlled by a key, as described in **7.1**.*

Catalogue key provisioning

*Many methods of acquiring access credentials are possible. These could involve a simple request, an exchange of further credentials or even a financial contract. **7.4** specifies a catalogue `rel` which signals to a client where a key may be sought for access to a resource or catalogue. The nature of the method referred to is out of scope for this PAS – although it could be, for example, a website requiring a human to sign-up manually to obtain a key, or some mechanism which can be used automatically by a client.*

7.4.1 Credential metadata

7.4.1.1 Where a server wishes to provide a hint to a client about how the client can acquire access credentials to either the catalogue or a resource in the catalogue, the catalogue shall contain a `catalogue-metadata` object (catalogue) or `item-metadata` object (resource) as specified in Table 19.

Table 19 – Credential acquisition relation

rel	val
urn:X-hypercat:rels:acquireCredential	URL

7.4.1.2 Where a server uses the `rel` set out in Table 19 the corresponding `val` shall always be a URL, resolving to a self-describing web page or resource helping the client acquire credentials.

***NOTE** In its simplest form, the URL may be a human-readable web page explaining what to do. More advanced systems may wish to use machine-readable documents to enable automatic negotiation of credentials.*

Where multiple `acquireCredential` declarations are present, the client should assume that credentials can be acquired in multiple ways.

8 Other extensions

8.1 Catalogue subscription

*COMMENTARY ON **8.1***

General

***8.1** describes a simple subscription system for catalogues using server-sent events. Further information on subscription can be found in Annex G.*

Server-sent events

The MIME media type `text/event-stream` describes a streaming text format for passing notifications to a listening client. Each message is delivered discretely over a streamed channel.

When used with HTTP(S), `text/event-stream` is supported in modern browsers using the HTML5 EventSource API [8]. Provided cross-origin resource sharing (CORS) requirements are met, client-side JavaScript applications can stream from servers and react to events.

Server-side, supporting `text/event-stream` is as simple as writing event text to a held-open HTTP(S) connection.

Model

A client subscribed to a catalogue server receives a stream of events. Every event contains an event ID, event name and event date. By first fetching a catalogue with HTTP(S), then accumulating these events, a client may keep a synchronized local copy of the catalogue being watched.

8.1.1 Event endpoints

*COMMENTARY ON **8.1.1***

Authentication may be required to access the event endpoint. Clients should assume that the same authentication systems used to fetch the catalogue over HTTP(S) can be used to access the event endpoint.

The following examples are given in Annex G:

- *a client fetching a catalogue, seeing the available subscription method and beginning to stream events from the provided endpoint;*
- *a client receiving an item update event for item http://example.org/item; and*
- *a client receiving an item deletion event for item http://example.org/item.*

Mechanisms for catalogue subscription are given in Annex H.

Where a server accepts subscriptions to a catalogue, that catalogue shall be annotated with the metadata specified in Table 20.

***NOTE** This informs clients that the subscription method is available.*

Table 20 – Event relations

`rel` value	`val` value
`urn:X-hypercat:rels:eventsource`	URL of event endpoint

8.1.2 Event format

8.1.2.1 Where an event is sent by the server, the server shall have `event` and `data` fields.

8.1.2.2 Where an event concerns a specific catalogue item within a catalogue, the event name shall be the URL-encoded `href` of the catalogue item (see Table 21).

Table 21 – Event format

Action	Event name	Event body
Notification of item change	<item `href` >	Catalogue item
Notification of item deletion	<item `href` >	Empty string
***NOTE** <item `href`> is the URL `href` field from a catalogue item.*		

8.2 Catalogue linked data `rel`

*COMMENTARY ON **8.2***

General

***8.2** describes a catalogue `rel` for specifying that the item at hand is an instance of an RDF class.*

Metadata

A single `rel` is described in Table 22.

The `val` shall point at an RDF Schema (RDFS) class in accordance with RDF Schema 1.1 [NR4].

Table 22 – Linked data relations

rel	Meaning	val
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	The resource is an instance of an RDF class	URI

8.3 Catalogue licence

*COMMENTARY ON **8.3***

***8.3** describes a catalogue `rel` for marking up the licence conditions under which a catalogue or linked resource may be used.*

8.3.1 Catalogue licences

Where data is found in a catalogue or a linked resource, a client shall be able to determine the licence under which the data is released by reading the metadata relations provided with the `href`.

***NOTE** The chosen licence may allow or disallow dissemination, sale or other forms of use by the client, and may define obligations on the catalogue publisher.*

8.3.2 Metadata

*COMMENTARY ON **8.3.2***

In order to allow maximum flexibility, a single `rel` is described in Table 23 which acts as a signal to clients that data is provided under a set of licence conditions. While some possible `val`s are described, users should select from the wide range of licences available across jurisdictions and data types.

Table 23 – Licence relations

rel	val
urn:X-hypercat:rels:hasLicense	URI

***NOTE** The spelling of the relation follows the American convention.*

8.3.2.1 Where the `val` is a URL, it shall point at a machine- or human-readable version of a licence [3], preferably a permalink.

***NOTE 1** The `hasLicense` metadata may be used in both `catalogue-metadata` and `item-metadata` sections to mark the licence conditions of the containing catalogue and of a linked resource.*

***NOTE 2** Where multiple `hasLicense` declarations are present, a client may make use of the data under any of the given licences.*

8.3.3 Cross domain trust

*COMMENTARY ON **8.3.3***

In the case where a catalogue provides `hasLicense` for a resource, it is possible for the licence to be incorrect for the resource. For example, a rogue catalogue could mark that all `.gov.uk` datasets are for limited use, when they are not.

An example of a catalogue declaring a licence for itself is given in Annex I.

Where a client finds `hasLicense` declarations in a catalogue, the client shall only trust those declarations for resources held on the same domain as the linking catalogue.

8.4 Catalogue robots exclusion search

*COMMENTARY ON **8.4***

General

***8.4** describes a catalogue extension allowing a catalogue to reference a `robots.txt` file.*

The robots exclusion standard

The robots exclusion standard [4] (also known as `/robots.txt`) is a de-facto standard based on work by Martikn Koster [9] and is not owned by any standards body.

It is used by websites to communicate with web crawlers and other web robots. The robots exclusion standard specifies the instruction format to be used to inform the robot about which areas of the website can and/or cannot be processed or scanned.

8.4.1 Where a served catalogue has an associated `robots.txt` file, the **`catalogue-metadata`** section shall include the `rel-val` pair: `{ "rel": "urn:X-hypercat:rels:hasRobotstxt", "val": "[BASE_URL]/robots.txt" }`.

8.4.2 Where a robots exclusion file is provided, it shall be located at the BASE_URL from which a catalogue is served (see **8.4.1**).

8.4.3 Where a robots exclusion file is provided, it shall be named `robots.txt`.

[3] For example: https://creativecommons.org/licenses/by/4.0/ or https://www.gnu.org/licenses/old-licenses/gpl-2.0.en.html

[4] See http://www.w3.org/TR/html4/appendix/notes.html#h-B.4.1.1

9 Claims of conformity

9.1 The creation of a software interface claimed to be in compliance with the requirements of this PAS shall be supported by a declaration of conformity to this PAS, stating that all the requirements set out in Clause **4** have been met.

9.2 The declaration shall also state whether the conditional requirements set out in Clauses **5** to **8** have been met.

9.3 Where the requirements set out in any of all of Clauses **5** to **8** have not been met, the declaration shall:

- state which Clauses are not included in the claim of compliance; and
- state the reasons for which these Clauses are not included in the claim of compliance.

Annex A (informative)
Adopting a security-minded approach

A.1 General

NOTE This annex was developed from source material provided by the Centre for the Protection of National Infrastructure (CPNI).

While there are benefits to sharing data through innovations such as the Internet of Things (IoT), Smart Cities, and big data, it is vital that users of this PAS are aware of the range of potential security issues, applicable to both the catalogue and items referenced within it, as a consequence of publishing on the internet, or making information otherwise publicly available.

It is important to recognise that once anything has been released in this manner, it is virtually impossible to delete, destroy, remove or secure all copies of it. Therefore, when publishing data or creating catalogues, particular care should be taken to protect information that could have an impact on the safety or security of:

- individuals;
- assets and systems, particularly those that deliver safety and/or security functionality;
- the benefits which the assets or systems exist to deliver; and/or
- intellectual property, through loss, misuse or compromise of integrity.

Further, the release of aggregated data, apparently innocuous when considered individually, can result in exposure of commercial, security or otherwise sensitive information. This can arise through the combination of separate catalogues, allowing relationships to be established through the context of the individual items.

Adoption of a security-minded approach, as set out in PAS 1192-5 [5], is therefore important to address the threats and opportunities that arise from publication of catalogues employing semantic or linked data approaches.

In order to fulfil the aims of both this PAS and PAS 1192-5, the security-minded approach should be holistic, addressing security around the aspects of people, process, as well as physical and technological security. The last of these should encompass:

- safety – preventing the creation, by the catalogue, of harmful states which could lead to injury or loss of life or unintentional environmental damage;
- authenticity – ensuring that the catalogue is genuine;
- availability (including reliability) – ensuring accessibility and usability of the catalogue in an appropriate and timely fashion;
- confidentiality – ensuring control of access and prevention of unauthorized access to sensitive information;
- integrity – maintaining consistency, coherence and configuration of the catalogue;
- possession – preventing unauthorized control, manipulation or interference with systems hosting the catalogue;
- resilience – ensuring the ability of systems hosting the catalogue to transform, renew and recover in a timely fashion in response to adverse events; and
- utility – ensuring usability and usefulness of the catalogue over time.

A.2 Security principles

The need for a security-minded approach should be assessed when:

- publishing a new catalogue;
- updating an existing catalogue; or
- undertaking a review of an existing catalogue.

On the occurrence of one of these triggers, the catalogue publisher should:

a) apply the security triage process set out in the CPNI guidance on open data [10] [6];

[5] PAS 1192-5:2015, Specification for security-minded building information modelling, digital built assets and smart asset management – http://shop.bsigroup.com/forms/PASs/PAS-1192-5/

[6] For the latest guidance on adopting a security-minded approach, see http://www.cpni.gov.uk/advice/Cross-cutting-advice/Open-data/

b) adopt an appropriate and proportionate security-minded approach. As a minimum, systems used to host and manage the catalogue should not be subject to any of the common vulnerabilities (e.g. the [Open Web Application Security Project] OWASP Top Ten list [7] and should be compliant with the UK government's Cyber Essentials scheme, or local equivalent [8];

c) understand the processes used to manage the catalogue over its lifecycle and ensure that they address the technical security considerations listed above;

d) ensure that persons involved in the management and support of the catalogue are appropriately trained and aware of the need for a security-minded approach;

e) ensure that appropriate technical and operational measures are in place to detect any unauthorized changes or access to the catalogue and the systems that host it;

f) ensure that appropriate forensic readiness measures are in place, which may be used to investigate any security breach affecting the catalogue.

If there is any uncertainty about the sensitivity of the items in the catalogue, or about the risks associated with data aggregation, appropriate advice should be sought (see PAS 1192-5:2015, **5.1.2** for further information).

A.3 On-going risk management

Threats and vulnerabilities change over time so it is important that the catalogue owner adopts an appropriate and proportionate risk management approach in order that the security risks to the catalogue are periodically reviewed.

[7] The OWASP Top Ten list is accessible at: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

[8] Cyber Essentials Scheme: https://www.cyberstreetwise.com/cyberessentials/

Annex B (informative)
Minimum valid catalogue

Figure B.1 shows the canonical minimum valid catalogue.

Technically, this PAS does not preclude leaving the description field as an empty string, however, it is advised that the field is meaningful to a human. Note that the ordering of elements of a JSON set is not defined, so `catalogue-metadata` may come before or after `items`.

Figure B.1 – Canonical minimum valid catalogue

```
{
     "catalogue-metadata":[
         {
               "rel":"urn:X-hypercat:rels:isContentType",
               "val":"application/vnd.hypercat.catalogue+json"
         },
         {
              "rel":"urn:X-hypercat:rels:hasDescription:en", "val":""
         }
    ],
    "items":[
    ]
}
```

Annex C (informative)
Search queries and results

Figure C.1 is an example catalogue containing two items which is used to illustrate simple search queries and results.

Figure C.1 – Example catalogue used to illustrate simple search queries

```
{
     "catalogue-metadata":[
          {
                "rel":"urn:X-hypercat:rels:isContentType",
                "val":"application/vnd.hypercat.catalogue+json"
          },
          {
               "rel":"urn:X-hypercat:rels:hasDescription:en",
               "val":"example catalogue"
          },
          {
               "rel":"urn:X-hypercat:rels:supportsSearch",
               "val":"urn:X-hypercat:search:simple"
          }
     ],
     "items":[
          {
               "href":"http://A",
               "item-metadata":[
                    {
                         "rel":"urn:X-hypercat:rels:hasDescription:en",
                         "val":"example item A",
                    },
                    {
                         "rel":"urn:X-hypercat:rels:1",
                         "val":"1"
                    },
                    {
                         "rel":"urn:X-hypercat:rels:2",
                         "val":"2"
                    },
                    {
                         "rel":"urn:X-hypercat:rels:3",
                         "val":""
                    }
               ]
      },
          {
               "href":"http://B",
               "item-metadata":[
                    {
                         "rel":"urn:X-hypercat:rels:hasDescription:en",
                         "val":"example item B",
                    }
               ]
          }
     ]
}
```

Given the catalogue in Figure C.1, the following query strings would all cause the server to respond with the zero-item catalogue shown in Figure C.2:

```
?rel=urn:X-hypercat:rels:4
?val=3
?rel=urn:X-hypercat:rels:1&val=2
?rel=urn:X-hypercat:rels:1&val=
```

Figure C.2 – Example catalogue resulting from simple search queries set out above

```
  {
       "catalogue-metadata":[
           {
                 "rel":"urn:X-hypercat:rels:isContentType", "val":"application/vnd.
                 hypercat.catalogue+json"
           },
           {
                "rel":"urn:X-hypercat:rels:hasDescription:en", "val":"example
                catalogue"
     }],
     "items":[
     ]
}
```

Given the catalogue in Figure C.1 on the server, the following query strings would all cause the server to respond with the single-item catalogue shown in Figure C.3:

```
?rel=urn:X-hypercat:rels:1
?rel=urn:X-hypercat:rels:2
?rel=urn:X-hypercat:rels:3
?val=1
?val=2
?val=
?rel=urn:X-hypercat:rels:1&val=1
?rel=urn:X-hypercat:rels:3&val=
```

Figure C.3 – Example catalogue resulting from above further simple search queries


```
  {
       "catalogue-metadata":[
           {
                  "rel":"urn:X-hypercat:rels:isContentType",
                  "val":"application/vnd.hypercat.catalogue+json"
           },
           {
                "rel":"urn:X-hypercat:rels:hasDescription:en",
                "val":"example catalogue"
           },
           {
                "rel":"urn:X-hypercat:rels:supportsSearch",
                "val":"urn:X-hypercat:search:simple"
           }
      ],
      "items":[
            {
                "href":"http://A",
                "item-metadata":[
                    {
                           "rel":"urn:X-hypercat:rels:hasDescription:en",
                           "val":"example item A",
                     },
                     {
                          "rel":"urn:X-hypercat:rels:1",
                          "val":"1"
                     },
                     {
                          "rel":"urn:X-hypercat:rels:2",
                          "val":"2"
                     },
                     {
                          "rel":"urn:X-hypercat:rels:3",
                          "val":""
                     }
                ]
       },
      ]
}
```

Annex D (informative)
Multi-search queries

Figures D.1 to D.4 give examples of search queries.

Figure D.1 gives an example of a catalogue search using a single `simpleSearch` query.

Figure D.1 – Example multi-search query containing a single query

```
{
    "query": "?rel=A"
}
```

Figure D.2 gives an example of a catalogue search, with the server responding with the intersection of two `simpleSearch` queries.

Figure D.2 – Example multi-search query for the intersection of two queries

```
{
    "intersection": [
        { "query": "?rel=A" },
        { "query": "?rel=B" }
    ]
}
```

Figure D.3 gives an example of a catalogue search, with the server responding with the union of two `simpleSearch` queries.

Figure D.3 – Example multi-search query for the union of two queries

```
{
    "union": [
        { "query": "?rel=C" },
        { "query": "?rel=D" }
    ]
}
```

Figure D.4 gives an example of catalogue search, with the server responding with the intersection of one `simpleSearch` query with the union of two other queries.

Figure D.4 – Example complex multi-search query

```
{
    "intersection": [
        { "query": "?rel=A" },
        { "union": [
            { "query": "?rel=C" },
            { "query": "?rel=D" }
          ]
        }
    ]
}
```

Annex E (informative)
Signing catalogues

Figure E.1 gives an example of an entire catalogue signature.

Figure E.1 – Example of signed catalogue

```
{
    "catalogue-metadata":[
        {
            "rel":"urn:X-hypercat:rels:isContentType",
            "val":"application/vnd.hypercat.catalogue+json"
        },
        {
            "rel":"urn:X-hypercat:rels:hasDescription:en",
            "val":"Signing example"
        },
        {
            rel: "urn:X-hypercat:rels:jws:key",
            val: "-----BEGIN RSA PUBLIC KEY-----..."
        },
        {
            rel: "urn:X-hypercat:rels:jws:signature",
            val: "YOWPexyGHKu4T_1M_vtlEnNlqmFOclqp4Hy6hVHfFT4"
        }
        {
            rel: "urn:X-hypercat:rels:jws:alg",
            val: "RS256"
        }
    ],
    "items":[
        ...
    ]
}
```

Figure E.2 gives an example of a per-item signature.

Figure E.2 – Example of signed item

```
{
    "catalogue-metadata":[
        {
            "rel":"urn:X-hypercat:rels:isContentType",
            "val":"application/vnd.hypercat.catalogue+json"
        },
        {
            "rel":"urn:X-hypercat:rels:hasDescription:en",
            "val":"Signing example"
        }
    ],
    "items":[
        "href":"https://example.org/sensor1",
        "item-metadata": [
            {
                rel: "urn:X-hypercat:rels:jws:key",
                val: "-----BEGIN RSA PUBLIC KEY-----..."
            },
            {
                rel: "urn:X-hypercat:rels:jws:signature",
                val: "YOWPexyGHKu4T_1M_vtlEnNlqmFOclqp4Hy6hVHfFT4"
            }
            {
                rel: "urn:X-hypercat:rels:jws:alg",
                val: "RS256"
            }
            {
                "rel":"urn:X-hypercat:rels:isContentType",
                "val":"application/senml+json"
            },
            {
                "rel":"urn:X-hypercat:rels:hasDescription:en",
                "val":"Sensor number one"
            }
        ]
    ]
}
```

Annex F (informative)
Providing an access hint

Figure F.1 gives an example of providing an access hint.

Figure F.1 – Example access hint

```
{
 ...
  "items": [
    {
      "href": "http://example.org/A",
      "item-metadata": [
        {
          "rel": "urn:X-hypercat:rels:accessHint",
          "val": "urn:X-hypercat:rels:oauth2.0"
        }
      ]
    }
  ]
}
```

Annex G (informative)
Catalogue subscription examples

Figure G.1 gives an example of a client fetching a catalogue, seeing the available subscription method and beginning to stream events from the provided endpoint.

Figure G.1 – Subscribing to a stream

```
GET /cat
```

(the client receives a catalogue containing a `urn:X-hypercat:rels:eventsource` relation with a value of `/cat/events` and decides to subscribe to catalogue updates)

```
GET /cat/events
```

(the client request initiates an EventSource stream. The server will now spontaneously send catalogue updates to the client whenever they occur, as shown in the examples below)

Figure G.2 gives an example of a client receiving an item update event for item http://example.org/item.

Figure G.2 – Receiving an update

```
id: 14:23:51
event: http://example.org/item
data: {"href":"http://example.org/item","item-metadata":[{"rel":"urn:Xhypercat:rels:ha
sDescription:en","val":"thing"},{"rel":"...","val":"..."},{"rel":"...","val":"..."}]}
```

Figure G.3 gives an example of a client receiving an item deletion event for item http://example.org/item.

Figure G.3 – Receiving a delete event

```
Id: 14:23:51
event: http://example.org/item
data:
```

Annex H (informative)
Mechanisms for catalogue subscription

COMMENTARY ON ***Annex H***

Annex H covers the case of streaming every change to an item in a catalogue. However, some users might wish to only receive events pertaining to particular items, or where `rel` *and* `val` *criteria are met. This filtering might be provided by passing URL parameters describing criteria in the streamed* `GET` *request.*

The details of the search mechanism are out of scope for this PAS.

H.1 Retrofitting

The simplest catalogue server is a traditional HTTP server. Implementing this subscription requires a more capable server, able to hold open long-lived connections and update clients with events. The mechanisms in **8.1** can be implemented as a proxy for an existing catalogue server, polling from the existing server and pushing events to clients. This enables adoption of subscription and streaming without impacting existing servers.

H.2 Resource subscription

Catalogues provide the ability to link to resources by URL/URI. Often, those resources contain data required by applications which can change in real-time. There are multiple IoT use-cases where client applications require real-time data feeds from devices, hubs or other services.

H.3 Catalogues versus resources

Catalogues are JSON documents, typically served over HTTP(S). A catalogue may list any number of resources by URI of any type, accessed with any protocol. Due to the wide range of resource types which may be linked, it is difficult to imagine a single, universal, mechanism for subscribing to resource data.

One possibility is to place all data directly into catalogues; for example, a sensor may insert its current temperature value into a catalogue as item metadata. While this approach enables existing subscription methods to be used, it falls short in some real world use-cases. This PAS has a simple data model of `rel`-`val` pairs of metadata, where `val`s are always strings. While this is ideal for ensuring interoperability of high-level metadata and aiding discovery of resources, it is a poor fit for real-world IoT asset and device data which is often too complex to fit this PAS's intentionally simple metadata model. For this reason, structured, live or complex IoT data is often held in resources, referenced in catalogues by URL/URI.

H.4 Resource types

There are many media-type protocol combinations suitable for streaming data which can be linked to by a catalogue, with each providing a different specific mechanism of subscribing to the resource. Where they are used, conventions should be established for linking to the resource and, optionally, mapping the received data to best-practice ontologies.

H.5 Alternative subscription transports

A number of transports other than HTTP may be used to provide catalogue subscription.

H.5.1 MQTT

MQTT is a popular lightweight pubsub protocol where clients connect to a central broker [9].

The MQTT protocol is designed for both publishing and subscribing and has a compact on-the-wire representation to allow use by lightweight devices. MQTT runs over transmission control protocol (TCP) and, as such, every client and server needs an MQTT implementation.

Security is provided by using secure sockets layer (SSL) or transport layer security (TLS) for the TCP socket and exchanging credentials with the broker.

[9] See http://www.oasis-open.org/committees/mqtt/

Although MQTT clients exist for most languages and programming environments, there are relatively few brokers. Every catalogue server wishing to implement an event interface based on MQTT needs to either implement an MQTT broker or connect to one.

MQTT URLs may be embedded in catalogue documents. At the time of publication, no formal URL scheme is available. Research should be undertaken to establish best practices for linking to streamed data with MQTT.

At the time of publication, MQTT does not provide the client with any hints as to the data type it receives. Catalogue `rel`s may be declared to inform clients of the kind of data to expect from an MQTT stream. For example, subscribing to the provided topic could provide payloads in a particular MIME-type.

H.5.2 Constrained application protocol (CoAP)

CoAP is a lightweight RESTful protocol designed to take web-like interaction down to the smallest devices. Operating over user datagram protocol (UDP), typically in Internet Protocol version 6 (IPv6) environments, CoAP extends the concept of client-server Internet interaction to the micro-controller level. Unlike HTTP, CoAP has an in-built mechanism for observing resources, allowing clients to request server-pushed updates in real-time.

CoAP has a fully documented URI scheme which may be used in catalogues [11].

Although a CoAP server can notify clients with a content-type header, it might be desirable to define extra `rel`s as hints to clients about the data they can expect from a CoAP endpoint.

H.5.3 Really simple syndication (RSS)/Atom

RSS and Atom are XML-based media-types intended to aid in the syndication of content on the web. Although not intended for streaming, RSS and Atom provide a mechanism for clients to poll and determine if a new resource is available. Placing a link to an RSS or Atom document in a catalogue should indicate to clients that they can monitor the RSS/Atom file for links to updated resources.

H.5.4 Extensible Messaging and Presence Protocol (XMPP)

XMPP is an XML-based messaging protocol designed for a range of client-server messaging applications [12]. Being very flexible, XMPP has been adapted to many purposes including streaming data about IoT resources.

Where XMPP resources are linked to from a catalogue, there is a formal definition of a URI scheme [13]. However, as with other protocols, it might be advantageous to provide extra metadata `rel`s telling the client what data type to expect at the end of the XMPP link.

Annex I (informative)
Declaring a licence

Figure I.1 gives an example of a catalogue declaring a licence for itself, containing a catalogue item for which it also declares a licence.

Figure I.1 – Example of a catalogue declaring a licence

```
{
  "catalogue-metadata": [
    {
      "rel": "urn:X-hypercat:rels:isContentType",
      "val": "application/vnd.hypercat.catalogue+json"
    },
    {
      "rel": "urn:X-hypercat:rels:hasDescription:en",
      "val": "licence example"
    },
    {
      "rel": "urn:X-hypercat:rels:hasLicense",
      "val": "http://hypercat.io/catlicense4.3"
    }
  ],
  "items": [
    {
      "href": "http://example.org/A",
      "item-metadata": [
        {
          "rel": "urn:X-hypercat:rels:hasLicense",
          "val": "http://example.org/publicfreedata4.3"
        }
      ]
    }
  ]
}
```

Bibliography

Standards publications

For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

BS EN ISO 3166-1:2014, *Codes for the representation of names of countries and their subdivisions – Part 1: Country codes (ISO 3166-1:2013)*

BS EN ISO 19115-1:2014, *Geographic information – Metadata – Part 1: Fundamentals*

BS ISO 8601, *Data elements and interchange formats – Information interchange – Representation of dates and times*

BS ISO/IEC 11179-3:2013, *Information technology – Metadata registries (MDR) – Part 3: Registry metamodel and basic attributes*

PAS 1192-5:2015, *Specification for security-minded building information modelling, digital built environments and smart asset management*

Other publications

[1] BERNERS-LEE, T., MASINTER, T., McCAHILL, M. *Uniform Resource Locators (URL)*. RFC 1738, December 1994. Available from: https://tools.ietf.org/html/rfc1738 [viewed May 2016]

[2] JONES, M., BRADLEY, J., SAKIMURA, N. *JSON Web Signature (JWS)*. RFC 7515, Internet Engineering Task Force (IETF), May 2015. Available from: https://tools.ietf.org/html/rfc7515 [viewed May 2016]

[3] JOSEFSSON, S. *The Base16, Base32, and Base64 Data Encodings*. RFC 3548, July 2003. Available from: https://www.ietf.org/rfc/rfc3548.txt [viewed May 2016]

[4] LINN, J. *Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures*. RFC 1421, February 1993. Available from: http://www.rfc-editor.org/rfc/rfc1421.txt [viewed May 2016]

[5] KENT, S. *Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management*. RFC 1422, February 1993. Available from: http://www.rfc-editor.org/rfc/rfc1422.txt [viewed May 2016]

[6] BALENSON, D. *Privacy Enhancement for Internet Electronic Mail: Part III: Algorithms, Modes, and Identifiers*. RFC 1423, February 1993. Available from: http://www.rfc-editor.org/rfc/rfc1423.txt [viewed May 2016]

[7] KALISKI, B. *Privacy Enhancement for Internet Electronic Mail: Part IV: Key Certification and Related Services*. RFC 1424, February 1993. Available from: http://www.rfc-editor.org/rfc/rfc1424.txt [viewed May 2016]

[8] HICKSON, I. *Server-Sent Events, W3C Working Draft 29 October 2009*. W3C, October 2009. Available from: https://www.w3.org/TR/2009/WD-eventsource-20091029/ [viewed May 2016]

[9] KOSTER, M., et al. *Standard for Robot Exclusion*. 1994. Available from: http://www.robotstxt.org/orig.html#status [viewed May 2016]

[10] CPNI. *Open Data: Adopting A Security-minded Approach*. Crown Copyright. November 2015. Available from: http://www.cpni.gov.uk/documents/publications/2015/25%20november%202015%20open%20data%20the%20need%20for%20a%20security-minded%20approach%20-%20%20full%20version.pdf?epslanguage=en-gb [viewed May 2016]

[11] SHELBY, Z., ARM., HARTKE, K., BORMANN, C. *The Constrained Application Protocol (CoAP)*. RFC 7252, Internet Engineering Task Force (IETF), June 2014. Available from: https://tools.ietf.org/html/rfc7252 [viewed May 2016]

[12] WAHER, P. *XEP-0323: Internet of Things - Sensor Data. XMPP Standards Foundation*. November, 2015. Available from: http://xmpp.org/extensions/xep-0323.html [viewed May 2016]

[13] BERNERS-LEE, T., FIELDING, R., MASINTER, L. *Uniform Resource Identifier (URI): Generic Syntax*. RFC 3986, January 2005. Available from: https://www.ietf.org/rfc/rfc3986.txt [viewed May 2016]

Further reading

BS ISO 646, *ISO 7-bit coded character set for information interchange*

Websites

Open Web Application Security Project (OWASP) Top Ten Project. Available from: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project [viewed May 2016]

Cyber Essentials Scheme. Available from: https://www.cyberstreetwise.com/cyberessentials/ [viewed May 2016]

British Standards Institution (BSI)

BSI is the national body responsible for preparing British Standards and other standards-related publications, information and services.

BSI is incorporated by Royal Charter. British Standards and other standardization products are published by BSI Standards Limited.

About us

We bring together business, industry, government, consumers, innovators and others to shape their combined experience and expertise into standards -based solutions.

The knowledge embodied in our standards has been carefully assembled in a dependable format and refined through our open consultation process. Organizations of all sizes and across all sectors choose standards to help them achieve their goals.

Information on standards

We can provide you with the knowledge that your organization needs to succeed. Find out more about British Standards by visiting our website at bsigroup.com/standards or contacting our Customer Services team or Knowledge Centre.

Buying standards

You can buy and download PDF versions of BSI publications, including British and adopted European and international standards, through our website at bsigroup.com/shop, where hard copies can also be purchased.

If you need international and foreign standards from other Standards Development Organizations, hard copies can be ordered from our Customer Services team.

Subscriptions

Our range of subscription services are designed to make using standards easier for you. For further information on our subscription products go to bsigroup.com/subscriptions.

With **British Standards Online (BSOL)** you'll have instant access to over 55,000 British and adopted European and international standards from your desktop. It's available 24/7 and is refreshed daily so you'll always be up to date.

You can keep in touch with standards developments and receive substantial discounts on the purchase price of standards, both in single copy and subscription format, by becoming a **BSI Subscribing Member**.

PLUS is an updating service exclusive to BSI Subscribing Members. You will automatically receive the latest hard copy of your standards when they're revised or replaced.

To find out more about becoming a BSI Subscribing Member and the benefits of membership, please visit bsigroup.com/shop.

With a **Multi-User Network Licence (MUNL)** you are able to host standards publications on your intranet. Licences can cover as few or as many users as you wish. With updates supplied as soon as they're available, you can be sure your documentation is current. For further information, email bsmusales@bsigroup.com.

Revisions

Our British Standards and other publications are updated by amendment or revision.

We continually improve the quality of our products and services to benefit your business. If you find an inaccuracy or ambiguity within a British Standard or other BSI publication please inform the Knowledge Centre.

Copyright

Useful Contacts:

Customer Services
Tel: +44 845 086 9001
Email (orders): orders@bsigroup.com
Email (enquiries): cservices@bsigroup.com

Subscriptions
Tel: +44 845 086 9001
Email: subscriptions@bsigroup.com

Knowledge Centre
Tel: +44 20 8996 7004
Email: knowledgecentre@bsigroup.com

Copyright & Licensing
Tel: +44 20 8996 7070
Email: copyright@bsigroup.com

This page is deliberately left blank.

This page is deliberately left blank.

BSI, 389 Chiswick High Road
London W4 4AL
United Kingdom
www.bsigroup.com

ISBN 978-0-580-96280-6