JSON Implementation of Time-Frequency Radio Catalogues: TFCat

- **Authors**: Cecconi, Baptiste (1); Taylor, Mark (2); Bonnin, Xavier (1); Loh, Alan (1).
- **Affiliations**: (1) LESIA, CNRS, Observatoire de Paris-PSL, France; (2) H H Wills Physics Laboratory, University of Bristol, Bristol, UK.

- **Date**: Sep. 2022
- **Version**: 1.0
- **Publisher**: PADC
- **DOI**: 10.25935/6068-8528

TFCat (Time-Frequency Catalogue) is a data interchange format based on JSON. It defines several types of JSON objects and the manner in which they are combined to represent data about time-frequency features of a time spectrogram (a.k.a. dynamic spectrum), their properties, and their temporal and spectral extents.

This implementation is inheriting from the GeoJSON file format [RFC7946].

1. Introduction

TFCat is a format for encoding a variety of spectro-temporal data structures using JavaScript Object Notation (JSON) [RFC7159]. The representation space of the structures is the time-frequency plane. We will use “tf-plane”, “tf-space”, “tf-spatial” and “tf-spatially” terms to refer to this time-frequency space. the TFCat format is following most of the GeoJSON specification [rfc7946]. All spatial information described in the GeoJSON specification are applicable, after transposition into the time-frequency plane space.

A TFCat object may represent a region of the tf-space (a Geometry), a tf-spatially bounded entity (a Feature), or a list of Features (a FeatureCollection). TFCat supports the same geometry types as GeoJSON: Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon, and GeometryCollection. Features in TFCat contain a Geometry object and additional properties, and a FeatureCollection contains a list of Features. The TFCat object contains a list of fields defining the properties listed in the Features, and a coordinate reference system (CRS) describing the temporal and spectral axes of the geometry data.

1.1. Requirements Language

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “NOT RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

1.2. Conventions Used in This Document

The ordering of the members of any JSON object defined in this document MUST be considered irrelevant, as specified by [RFC7159].

Some examples use the combination of a JavaScript single-line comment (//) followed by an ellipsis (...) as placeholder notation for content deemed irrelevant by the authors. These placeholders must of course be deleted or otherwise replaced, before attempting to validate the corresponding JSON code example.

Whitespace is used in the examples inside this document to help illustrate the data structures, but it is not required. Unquoted whitespace is not significant in JSON.

1.3. Definitions

The definitions of section 1.3 of [rfc7946] are applicable here, replacing the "GeoJSON" string by "TFCat". We recall them below.

- JavaScript Object Notation (JSON), and the terms object, member, name, value, array, number, true, false, and null, are to be interpreted as defined in [RFC7159].
- Inside this document, the term "geometry type" refers to seven case-sensitive strings: "Point", "MultiPoint", "LineString", "MultiLineString", "Polygon", "MultiPolygon", and "GeometryCollection"
- As another shorthand notation, the term "TFCat types" refers to nine case-sensitive strings: "Feature", "FeatureCollection", and the geometry types listed above.
- The word "Collection" in "FeatureCollection" and "GeometryCollection" does not have any significance for the semantics of array members. The "features" and "geometries" members, respectively, of these objects are standard ordered JSON arrays, not unordered sets.

1.4. Example

A TFCat FeatureCollection:

```
{
    "type": "FeatureCollection",
    "features": [
        {
            "type": "Feature",
            "id": 0,
            "geometry": {
                "type": "Point",
                "coordinates": [1158051858, 24730.0]
            },
            "properties": {
                "quality": "3"
```

```
            }
        },
        {
            "type": "Feature",
            "id": 1,
            "geometry": {
                "type": "LineString",
                "coordinates": [[1158051874, 34130.0], [1158051882, 22770.0]]
            },
            "properties": {
                "quality": "2"
            }
        }
    ],
     "properties": {
        "facility_name": "Cassini",
        "instrument_name: ": "RPWS",
        "receiver_name": "HFR",
        "title": "Faraday Rotation (FR) patches",
    },
    "fields": {
      "quality": {
        "info": "Quality parameter: -1 (not set), 1 (good), 2 (medium) and 3 (bad quality).'
        "datatype": "str",
        "ucd": "meta.code.qual"
      }
    },
    "crs": {
        "type": "local",
        "properties": {
            "time_coords_id": "unix",
            "spectral_coords": {
                "type": "frequency",
                "unit": "kHz"
            },
            "ref_position_id": "cassini-orbiter"
        }
    }
}
```


2. TFCat Text

The TFCat Text is defined similarly to Section 2 of [rfc7946], replacing every instance of "GeoJSON" by "TFCat". For the sake of clarity, we reproduce the applicable definition below.

A TFCat text is a JSON text and consists of a single TFCat object.

3. TFCat Object

> The TFCat Object is defined similarly to Section 3 of [rfc7946], replacing every instance of "GeoJSON" by "TFCat". For the sake of clarity, we reproduce the applicable definition below.

A TFCat object represents a Geometry, Feature or a collection of Features.

- A TFCat object is a JSON object.
- A TFCat object has a member with the name "type". The value of the member MUST be one the TFCat types.
- A TFCat object MAY have a "crs" member, the value of which MUST be a Coordinate Reference System object (see Section 4).
- A TFCat object MAY have a "bbox" member, the value of which MUST be a bounding box array (see Section 5).
- A TFCat object MAY have other members (see Section 6).

3.1. Geometry Object

> The Geometry Object is defined similarly to Section 3.1 of [rfc7946], replacing every instance of "GeoJSON" by "TFCat". For the sake of clarity, we reproduce the applicable definition below.

A Geometry object represents points, curves, and surfaces in coordinate space. Every Geometry object is a TFCat object no matter where it occurs in a TFCat text.

- The value of a Geometry object's "type" member MUST be one of the seven geometry types (see Section 1.3).
- A TFCat Geometry object of any type other than "GeometryCollection" has a member with the name "coordinates". The value of the "coordinates" member is an array. The structure of the elements in this array is determined by the type of geometry. TFCat processors MAY interpret Geometry objects with empty "coordinates" arrays as null objects.

3.1.1. Position

> The Position is defined similarly as Section 3.1.1 of [rfc7946], with restrictions. For the sake of clarity, we reproduce the applicable definition below.

A position is the fundamental geometry construct. The "coordinates" member of a Geometry object is composed of either:

- one position in the case of a Point geometry,
- an array of positions in the case of a LineString or MultiPoint geometry,

- an array of LineString or linear ring coordinates in the case of a Polygon or MultiLineString geometry, or
- an array of Polygon coordinates in the case of a MultiPolygon geometry.

A position is an array of numbers. There MUST be two elements. The two elements are the temporal and spectral coordinates, in that order, using decimal numbers.

A line between two positions is a straight Cartesian line, the shortest line between those two points in the coordinate reference system (see Section 4).

In other words, every point (t, f) on a line between a point (t0, f0) and (t1, f1) can be calculated as

```
(t, f) = (t0 + (t1 - t0) * x, f0 + (f1 - f0) * x)
```

with `x` being a real number greater than or equal to 0 and smaller than or equal to 1.

Examples of positions and geometries are provided in Appendix A, "Geometry Examples".

3.1.2. Geometric Object Classes

For Point, MultiPoint, LineString, MultiLineString, Polygon, and MultiPolygon, the same definitions as in section 3.1.2. to 3.1.7. of [rfc7946] are applicable. The GeometryCollection is also defined as in section 3.1.8. of [rfc7946].

Section 3.1.9 of [rfc7946] is NOT applicable.

3.1.3. Uncertainty and Precision

Section 3.1.10 of [rfc7946] is applicable.

3.2. Feature Object

A Feature object represents a spectro-temporal bounded thing. Every Feature object is a TFCat object no matter where it occurs in a TFCat text.

- A Feature object has a "type" member with the value "Feature".
- A Feature object has a member with the name "geometry". The value of the geometry member SHALL be either a Geometry object as defined above or, in the case that the Feature is unlocated, a JSON null value.
- A Feature object has a member with the name "properties". The value of the properties member is a Properties Object or a JSON null value.
- If a Feature has a commonly used identifier, that identifier SHOULD be included as a member of the Feature object with the name "id", and the value of this member is either a JSON string or number.

3.2.1. Properties Object

The Properties Object contains additional metadata associated to the corresponding Feature Object. The Properties Object is a JSON Object, which members are called "property".

- Each "property" name MUST be defined in the Fields Object (see Section 3.3.1.).
- A "property" value MUST be either a JSON string or number.

3.3. FeatureCollection Object

A TFCat object with the type "FeatureCollection" is a FeatureCollection object. A FeatureCollection object has a member with the name "features". The value of "features" is a JSON array. Each element of the array is a Feature object as defined above. It is possible for this array to be empty.

- The FeatureCollection Object has a member with name "fields". The value of the "fields" member is a JSON Object.
- The names of "fields" object members MUST correspond to the "property" members of the Properties Objects of the Feature Objects listed in the "features" member.
- The members of the "fields" Object are Field Objects.

3.3.1. Field Object

The Field Object contains the metadata defining one of the "property" included in the Properties Object members of each Feature Object (see Section 3.2.1.). The Field Object is a JSON object.

- The Field Object has a member with name "info". The value of this member contain a description of the "field".
- The Field Object has a member with name "datatype". The value of this member is one of 'int', 'float' 'bool', or 'str' (TBC)
- The Field Object has a member with name "ucd". The value of this member is a valid UCD (IVOA Unified Content Description).
- The Field Object may have a member with name "unit". The value of this member is a valid unit, as defined in the VOUnit specification. If "unit" is not applicable, the member should not be present.

4. Coordinate Reference System

The coordinate reference system (CRS) for all TFCat coordinates is a time-frequency coordinate reference system, which consists in a temporal coordinate reference system and a spectral coordinate reference system. It also defines the CRS reference position.

The CRS of a TFCat object is determined by its "crs" member (referred to below as the CRS object). If a TFCat object has no "crs" member, then its parent's

CRS is considered to apply. A CRS MUST be in scope for every Geometry object. This is usually applied by providing a "crs" member in the top-level TFCat object in a TFCat Text.

The CRS object contains the description of the TimeFrame and SpectralFrame.

- A CRS object has a member with the name "type". The value of member MUST be one of "local", "link" or "name". In this version, solely the "local" value is actually implemented, so that the "link" and "name" SHOULD not be used.
- A CRS object has a member with the name "properties". The value of the properties member is an object.

4.1. Local CRS

- A Local CRS object is a CRS object with its "type" member set to "local".
- A Local CRS object has a member with the name "properties". The value of the properties member is a Local CRS Properties Object

4.1.1 Local CRS Properties Object

- A Local CRS Properties Object MAY have a member with name "time_coords". The value of this element is a TimeCoords object.
- A Local CRS Properties Object MAY have a member with name "time_coords_id". The velue of this member MUST be one of the following values: "unix", "jd", "mjd", "mjd_nasa", "mjd_cnes", "cdf_tt2000". The definitions of the "time_coords_id" values is provided in Appendix B.
- One of the "time_coords" and "time_coords_id" members MUST be present in a Local CRS Properties Object.
- A Local CRS Properties Object MUST have a member with name "spectral_coords". The value of this element is a SpectralCoords object.
- A Local CRS Properties Object MUST have a member with name "ref_position_id". The value of this member is a named reference location. The allowed values for this member SHOULD be either from the https://www.ivoa.net/rdf/refposition list of values, or the spacecraft name (free string until an Observation Facility vocabulary is available).

Example of Local CRS object:

```
"crs": {
  "type": "local",
  "properties": {
    "time_coords_id": "unix",
    "spectral_coords": {
      "type": "frequency",
      "unit": "kHz"
    },
    "ref_position_id": "cassini-orbiter"
```

```
  }
}
```

4.1.2 TimeCoords Object A TimeCoords object represents the time coordinate reference system in use for the 1st coordinate axis of the Point objects. The time coordinate reference system is defined by a frame (with time scale and a reference position), and a representation (e.g., a unit and a time origin).

- A TimeCoords object may have a member with name "name", with value set to a character string describing the temporal coordinate axis. The value of this member may be used by display tools for presenting the data.
- A TimeCoords object has a member with name "time_scale", with values in the list: GPS, TAI, TCB, TCG, TDB, TT, UT, UTC, UNKNOWN. The definition of the
  time scale terms is available from https://www.ivoa.net/rdf/timescale/
  Two additional allowed values are: SCET (SpaceCraft Event Time), SCLK (Spacecraft Clock). They refer to the time measured by the spacecraft clock, SCET being transformed into a user readable unit, and SCLK is in the internal spacecraft clock format.

- A TimeCoords object has a member with name "unit", with value set to the unit of the time axis. The unit MUST be a valid VOUnit string.
- A TimeCoords object has a member with name "time_origin", with value set to the origin of the time axis, expressed in the following ISO 8601-like format:

  `[-]YYYY-MM-DD['T'hh:mm:ss[.SSS]]`

Example of TimeCoords object:

```
"time_coords": {
    "name": "Timestamp (Unix Time)",
    "unit": "s",
    "time_origin": "1970-01-01T00:00:00",
    "time_scale": "UTC"
}
```

```
"time_coords": {
    "name": "Modified Julian Date",
    "unit": "d",
    "time_origin": "1858-11-17T00:00:00.0",
    "time_scale": "UTC"
}
```

4.1.3. SpectralCoords

A SpectralCoords object represents the spectral coordinate reference system in use for the 2nd coordinate axis of the Point objects. The spectral coordinate reference system is defined by a reference position and a unit.

- A SpectralCoords object has a member with name "type", with value set to a character string describing the spectral coordinate axis. The allowed values are: "frequency", "wavelength", "energy", "wavenumber".
- A SpectralCoords object has a member with name "unit", with value set to the unit of the spectral axis. The unit MUST be a valid VOUnit string.
- A SpectralCoords object may have a member with name "scale", with value set to ether "linear" or "log". If present, the value of this member may be used by display tools to present the data with a linear or logarithmic spectral scale.

Example of SpectralCoords object:

```
"spectral_coords": {
    "type": "frequency",
    "unit": "MHz"
}
```

```
"spectral_coords": {
    "type": "wavelength",
    "unit": "cm"
}
```

4.2. Linked CRS

- A Linked CRS object is a CRS object with its "type" member set to "link".
- A Linked CRS object has a member with the name "properties". The value of the properties member is a Linked CRS Properties Object.

4.2.1 Linked CRS Properties Object

- A Linked CRS Properties Object has a member with name "href", with value set to a URL linking to the TFCat CRS definition;
- A Linked CRS Properties Object has a member with name "type", with value set to a string defining the standard used to describe the linked TFCat CRS definition.

In this version of the TFCat specification, the allowed values for the "type" element are not defined. Hence, the Linked CRS object SHOULD NOT be used.

4.3. Named CRS

- A Named CRS object is a CRS object with its "type" member set to "name".

- A Named CRS object has a member with the name "properties". The value of the properties member is a Named CRS Properties Object.

4.3.1 Named CRS Properties Object

- A Named CRS Properties Object has a member with name "name", with value set to a string referring to a well-known TFCat CRS.

In this version of the TFCat specification, the allowed values for the "name" element are not defined. Hence, the Named CRS object SHOULD NOT be used.

5. Bounding Box

> Section 5 [rfc7946] is applicable, after replacing "GeoJSON" be "TFCat". The following text is a simplified normative version applicable to TFCat.

A TFCat object MAY have a member named "bbox" to include information on the coordinate range for its Geometries, Features, or FeatureCollections. The value of the bbox member MUST be an array of length 4 and MUST be of the form:

```
[lower_time, lower_spectral, upper_time, upper_spectral]
```

6. Extending TFCat and Versioning

> Section 6, 7 and 8 of [rfc7946] are applicable, after replacing "GeoJSON" be "TFCat".
>
> Section 9 of [rfc7946] is NOT applicable.

7. Security Considerations

TFCat shares security issues common to all JSON content types. See [RFC7159], Section 12 for additional information. TFCat does not provide executable content.

TFCat does not provide privacy or integrity services. If sensitive data requires privacy or integrity protection, those must be provided by the transport – for example, Transport Layer Security (TLS) or HTTPS. There will be cases in which stored data need protection, which is out of scope for this document.

8. Interoperability Considerations

8.1. I-JSON

TFCat texts should follow the constraints of Internet JSON (I-JSON) [RFC7493] for maximum interoperability.

8.2. IVOA

TFCat uses several IVOA elements, which improves its interoperability. The units SHOULD be specified following the VOUnits specification. The reference positions and time scale, used to define the coordinate reference systems are using terms defined in IVOA controlled vocabularies, which are, respectively, RefPosition and TimeScale.

9. References

9.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, http://www.rfc-editor.org/info/rfc2119.
- [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, http://www.rfc-editor.org/info/rfc6838.
- [RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, http://www.rfc-editor.org/info/rfc7159.
- [RFC7493] Bray, T., Ed., "The I-JSON Message Format", RFC 7493, DOI 10.17487/RFC7493, March 2015, http://www.rfc-editor.org/info/rfc7493.
- [RFC7946] Butler, H., Ed., "The GeoJSON Format", RFC 7946, https://tools.ietf.org/html/rfc7946
- [UCD1+] Preite Martinez, A., M. Louys, B. Cecconi, S. Derrière, F. Ochsenbein, S. Erard, M. Demleitner. 2021. UCD1+ controlled vocabulary - Updated List of Terms. Version 1.4. IVOA Endorsed Note. https://ivoa.net/documents/UCD1+/
- [VOUnits] Demleitner, M., S. Derriere, N. Gray, M. Louys, and F. Ochsenbein. 2014. Units in the VO. Version 1.0. IVOA Recommendation. https://ivoa.net/documents/VOUnits/

A. Geometry Examples

Each of the examples below represents a valid and complete TFCat object. In the provided examples, the time values are provided in "unixtime" (seconds since 1970-01-01), and frequency values are proided in units of Hz.

A.1. Points

Point coordinates are in t, f order (time, frequency):

```
{
    "type": "Point",
    "coordinates": [1158051858, 24730.0]
}
```

A.2. LineStrings

Coordinates of LineString are an array of positions (see Section 3.1.1):

```
{
    "type": "LineString",
    "coordinates": [
        [1158051858, 24730.0],
        [1158051858, 24735.0]
    ]
}
```

A.3. Polygons

Coordinates of a Polygon are an array of linear ring (see [rfc7946] Section 3.1.6) coordinate arrays. The first element in the array represents the exterior ring. Any subsequent elements represent interior rings (or holes).

No holes:

```
{
    "type": "Polygon",
    "coordinates": [
        [
            [1158051858, 24730.0],
            [1158051868, 24730.0],
            [1158051868, 24735.0],
            [1158051858, 24735.0],
            [1158051858, 24730.0]
        ]
    ]
}
```

With holes:

```
{
    "type": "Polygon",
    "coordinates": [
        [
            [1158051858, 24730.0],
            [1158051868, 24730.0],
            [1158051868, 24735.0],
            [1158051858, 24735.0],
            [1158051858, 24730.0]
        ],
        [
            [1158051860, 24731.0],
            [1158051860, 24734.0],
            [1158051866, 24734.0],
```

```
            [1158051866, 24731.0],
            [1158051860, 24731.0]
        ]
    ]
}
```

A.4. MultiPoints

Coordinates of a MultiPoint are an array of positions:

```
{
    "type": "MultiPoint",
    "coordinates": [
         [1158051858, 24730.0],
         [1158051868, 24735.0]
    ]
}
```

A.5. MultiLineStrings

Coordinates of a MultiLineString are an array of LineString coordinate arrays:

```
{
    "type": "MultiLineString",
    "coordinates": [
        [
         [1158051858, 24730.0],
         [1158051868, 24735.0]
        ],
        [
         [1158051878, 24740.0],
         [1158051888, 24745.0]
        ]
    ]
}
```

A.6. MultiPolygons

Coordinates of a MultiPolygon are an array of Polygon coordinate arrays:

```
{
    "type": "MultiPolygon",
    "coordinates": [
        [
            [
                [1158051858, 24730.0],
                [1158051868, 24730.0],
                [1158051868, 24735.0],
```

```
                [1158051858, 24735.0],
                [1158051858, 24730.0]
            ]
        ],
        [
            [
                [1158051878, 24730.0],
                [1158051888, 24730.0],
                [1158051888, 24735.0],
                [1158051878, 24735.0],
                [1158051878, 24730.0]
            ],
            [
                [1158051880, 24731.0],
                [1158051880, 24734.0],
                [1158051886, 24734.0],
                [1158051886, 24731.0],
                [1158051880, 24731.0]
            ]
        ]
    ]
}
```

A.7. GeometryCollections

Each element in the "geometries" array of a GeometryCollection is one of the Geometry objects described above:

```
{
    "type": "GeometryCollection",
    "geometries": [{
        "type": "Point",
        "coordinates": [1158051880, 24731.0]
    }, {
        "type": "LineString",
        "coordinates": [
            [1158051878, 24730.0],
            [1158051888, 24730.0]
        ]
    }]
}
```

B. TFCat CRS Predefined TimeCoords

time_coords_id	Description	time_origin	unit	time_scale
unix	Unix Timestamp	1970-01-01T00:00:00	s	UTC
jd	Julian Day	-4712-01-01T12:00:00	d	UTC
mjd	Modified Julian Day	1858-11-17T00:00:00	d	UTC
mjd_nasa	NASA Modified Julian Day	1968-05-24T00:00:00	d	UTC
mjd_cnes	CNES Modified Julian Day	1950-01-01T00:00:00	d	UTC
cdf_tt2000	CDF Epoch TT2000	2000-01-01T00:00:00	ns	TT