Epidemics 15 (2016) 20–26

Contents lists available at ScienceDirect

Epidemics

journal homepage: www.elsevier.com/locate/epidemics

EpiJSON: A unified data-format for epidemiology

Thomas J.R. Finnie [a,*], Andy South [b], Ana Bento [c], Ellie Sherrard-Smith [c], Thibaut Jombart [c]

[a] Emergency Response Department Science and Technology, Public Health England, Porton Down, Salisbury, United Kingdom
[b] Consultancy, Norwich, United Kingdom
[c] MRC Centre for Outbreak Analysis and Modelling, Department of Infectious Disease Epidemiology, School of Public Health, Imperial College London, London, United Kingdom

ARTICLE INFO

Article history:
Received 7 July 2015
Received in revised form
11 December 2015
Accepted 21 December 2015
Available online 29 December 2015

Keywords:
Outbreaks
Epidemics
Software
Databases
Communications standards

ABSTRACT

Epidemiology relies on data but the divergent ways data are recorded and transferred, both within and between outbreaks, and the expanding range of data-types are creating an increasingly complex problem for the discipline. There is a need for a consistent, interpretable and precise way to transfer data while maintaining its fidelity. We introduce 'EpiJSON', a new, flexible, and standards-compliant format for the interchange of epidemiological data using JavaScript Object Notation. This format is designed to enable the widest range of epidemiological data to be unambiguously held and transferred between people, software and institutions. In this paper, we provide a full description of the format and a discussion of the design decisions made. We introduce a schema enabling automatic checks of the validity of data stored as EpiJSON, which can serve as a basis for the development of additional tools. In addition, we also present the R package 'repijson' which provides conversion tools between this format, line-list data and pre-existing analysis tools. An example is given to illustrate how EpiJSON can be used to store line list data. EpiJSON, designed around modern standards for interchange of information on the internet, is simple to implement, read and check. As such, it provides an ideal new standard for epidemiological, and other, data transfer to the fast-growing open-source platform for the analysis of disease outbreaks.

1. Introduction

Infectious disease epidemiology relies on integrating increasingly diverse and complex data. This complexity comes not only from the types of data now collected (for example genetic sequence, image and digital sensor data are routinely generated during the course of a disease outbreak, together with more traditional epidemiological data) but also through multiple partners investigating different facets, from different specialities or covering different geographical areas. This has been seen in recent major epidemics including the 2009 influenza pandemic (Fraser et al., 2009), Middle-East Respiratory Syndrome outbreaks (Cauchemez et al., 2014) or the West-African Ebola epidemic (WHO Ebola Response Team, 2014, 2015). In this context, the safe storage and swift exchange of epidemiological data between collaborators and institutions is key to the successful assessment of, and response to, infectious disease epidemics. Consequently, a great deal of effort has been recently devoted to standardising platforms for the analysis of epidemiological data with software tools being constructed to permit interoperability between separate methodological approaches (Jombart et al., 2014). Similar efforts have also been made in the fields of epidemiological data-gathering and recording (Aanensen et al., 2009; ECDC, 2015).

Overall however, there is a scarcity of systematised approaches for the transfer of data. The production of such a capability would vastly improve our ability to transfer information between systems and in doing so aid the interpretation of disease dynamics and ultimately protect a greater number of individuals. Yet epidemics data are still, usually, held as a potentially confusing mass of spread-sheets, databases, text and binary files. A universal format enabling the coherent storage and transfer of these data is lacking. As a consequence, misinterpretation of the data may happen during transfer and result in errors being introduced into subsequent analyses and reports. Unfortunately, the inherent complexity of epidemiological data magnifies the risks of such errors. Fig. 1 illustrates the major systems within an epidemiology work-flow where a standard for digital epidemiology data would be of assistance. A major difficulty associated with transferring epidemiological data lies in the degree of complexity that a dataset may display. The information that is recorded may vary markedly not only between outbreaks but also within a single outbreak. In addition, the

* Corresponding author. Tel.: +44 (0) 1980 616940.
E-mail address: Thomas.Finnie@phe.gov.uk (T.J.R. Finnie).
URL: http://andysouth.co.uk (A. South).

http://dx.doi.org/10.1016/j.epidem.2015.12.002

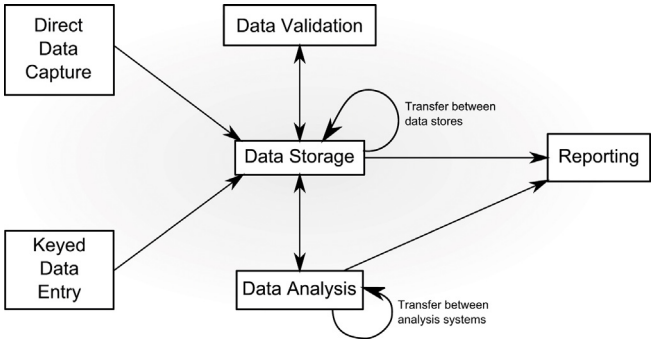

Fig. 1. The major components of an epidemiology workflow (blocks) and the places where a standard transfer format would be of assistance (arrows).

epidemic context itself makes data collection a daunting task, leading to some inevitable disparities in the data recorded.

Despite these challenges, we can identify a common structure to epidemiological datasets that can make the task of storing them easier. At the top level of this common structure is information relating to the dataset as a whole, such as the name of the infection that is causing the epidemic or the particular geographic setting of the study. This information is meta-data. At a second level, most datasets are divided into subunits (units-of-record) that hold other information. These subunits could be individuals, regions, countries or time periods. In a conventional spreadsheet these units-of-record are usually stored as rows. The information relating to these units-of-record makes up the third level and is usually stored as columns in a conventional spreadsheet. This information can either relate directly to the unit-of-record itself (such as gender for an individual) or can relate to an event happening to or at that unit-of-record (such as the onset of symptoms for an individual).

Any format for the conveyance of epidemiological data has two competing goals: consistency and flexibility. With this and the common morphology of a dataset in mind, we propose a standard for the storage and transmission of data for infectious disease epidemiology: EpiJSON (Epidemiological JavaScript Object Notation). This format is intended to be language and software agnostic, simple to implement, and leverages modern data standards whilst maintaining the flexibility to represent most epidemiological data.

While initially developed for problems within the infectious disease domain, EpiJSON is applicable to any dataset where "events" happen to "units of record". We believe that it is sufficiently flexible to accommodate other datasets such as those found in non-communicable disease and chemical hazard areas. It has been designed to draw together all relevant epidemiological data into a single place so that, for example, genetic sequences may be stored alongside image data, a patient's standard demographic information and the disease trajectory data in an unambiguous manner.

2. Material and methods

2.1. Structure of the EpiJSON format

The EpiJSON format capitalises on the common structure of most epidemiological datasets outlined above. Fundamentally, the structure of an EpiJSON file consists of three levels that we term "metadata", "records" and "events" (Fig. 2). Within each of these three levels, data are stored in collections of objects called "attributes" which are the core of data storage in EpiJSON. An "attribute" object is used for storing unambiguously a discrete piece of information, recording not only the value of the data but also its name, type and units. The "name" is a label defining what the attribute is (e.g. "age" or "gender"), "type" defines the type of data being stored (e.g. "number" or "string"), "value" is the actual data

Fig. 2. Block diagram of the structure of an EpiJSON file.

Table 1
Keys and values for the base JSON object contained in an EpiJSON file. Square brackets following a data type indicate arrays.

Key	Type	Value	Description
"metadata"	attribute[]	Zero or more attribute objects	A set of attribute objects relating to the dataset as a whole
"records"	record[]	Zero or more record objects	A set of record objects containing the dataset

value (e.g. 42 or "male"), and the optional "units" key specifies measurement units (e.g. "years"). As this data representation is very generic, it is used as a unit of data storage across the whole structure of EpiJSON. For example: an "attribute"" object could be used to store the name of an infection within "metadata", the gender of an individual within a "record", and the infection status of an individual at a test within an "event". The difference between an "event" and an "attribute" is that an "event" occurs at a defined time or place and can therefore store dates and locations using standard formats.

The advantage of the EpiJSON file structure is to clarify which data refer to the dataset, to records or to events associated with records. In contrast, conventional line list and other spreadsheet data can cause confusion as columns are often used to store all levels of data.

2.2. Outline of EpiJSON format

An EpiJSON file is essentially a text file containing a standards compliant JSON object. JSON is a widespread, language independent, human readable data format. JSON is made up of key/value pairs where keys are names and values are the data. EpiJSON files are readable by any system capable of reading JSON even if it is not directly aware of the EpiJSON structure.

An EpiJSON file consists of two parts: the metadata and the dataset (Fig. 2, Table 1). Both parts are arrays of objects. Within this manuscript, an "array" refers to a one-dimensional collection of objects of the same type. Arrays are indicated using square brackets "[]" immediately following the data type. For instance, an

Table 2
Keys and values for an attribute object.

Key	Type	Value	Description
"name"	string	string	An identifier for this attribute
"type"	string	"string" "number" "integer" "boolean" "date" "location" "base64"	Type must be one of the enumerated types listed. (See Table 3 for a fuller explanation of type definitions.)
"value"	string; numeric; boolean		The value to be recorded. It must be a valid example of "type". May be a homogeneous array of one of the permitted types
"units"	string	string	A string with the UDUNITS2 unit name. For non-numeric or non-dimensional attributes this key may be omitted. Non-standard units *may* be used but this is not recommended. If included the string must not be empty

Table 3
Possible values for the type key of an attribute object and the consequence for the data held in the value key.

Type	Value
"string"	A character string of unspecified length
"number"	A decimal number. In languages making a distinction between numeric types, it is recommended that the "value" key is a signed floating point number encoded on at least 64 bits. Numbers must not contain any whitespace. Commas and periods may only appear where they indicate the decimal place
"integer"	An integer number. In languages making a distinction between numeric types, it is recommended that the "value" key is a signed integer number encoded on at least 32 bits
"boolean"	A Boolean value, true or false. If a language supports Boolean types then the implementation may treat the value key as Boolean. Must be lower-case and unquoted
"date"	A character string representing a date conforming to RFC3339 (Newman and Klyne, 2002). E.g. 1996-12-20T00:39:57Z note the 'Z' at the end indicating zero offset from UTC. Time zones are represented as numeric offsets from UTC in hh:mm format (e.g. "1996-12-19T16:39:57-08:00" would represent the same time in Pacific Standard Time)
"location"	A GeoJSON object representing a spatial entity. Note: although attributes can hold either spatial (location) or temporal (date) information this is mostly for use in metadata. For records, the event object is the recommended form for storing this information
"base64"	A character string of binary data encoded to a text character set using base 64 encoding as per RFC4648 (Josefsson, 2006). By including a method of holding binary data within an EpiJSON file we permit abstract data to be included

array of integers will be noted "integer[]". The metadata is an array of attribute objects (attribute[]) while the dataset is an array of record objects (record[]). Both the metadata and the record keys are required but may hold zero objects. The "metadata" key is required to enable the inclusion of information describing the dataset whilst the "records" key holds the data.

2.2.1. Attributes

Attribute objects are a fundamental concept in the EpiJSON format. This construct holds data on a parent object in a form that permits a clear understanding of the data. An attribute object consists of keys: "name"", "type", "value" and "units" (Table 2). The "name" key is a character string identifying the name of this attribute. The "type" key identifies the type of data that is held under the value key. This is a character string but is limited to certain enumerated values contained in Table 3. Identifying the data type is necessary to ensure that software will correctly interpret the data when parsing an EpiJSON file. The "value" key holds the actual value of the attribute object. This may be either a character string, a number, an integer, a Boolean value, a JSON object or a homogeneous array of one of these (other types as specified in the type key are effectively sub-types of these primitives). The object held in the value key must be compatible with the value of the type key. For numeric value types the "units" key permits the units in which values have been recorded to be included in the dataset.

2.2.2. Records

An EpiJSON dataset is made up from a series of record objects detailing units-of-record. Certain characteristics of a unit-of-record are fixed (e.g. gender or region name) and may be recorded as part of the "attributes" of a given record. Alternatively, other characteristics occur in space or time and are stored as "events". The record object consists of three keys: "id", "attributes" and "events" (Table 4). The "id" key should be a string conforming to and generated according to version 4 UUID (Universally Unique IDentifier) specification as found in RFC 4122 (Leach et al., 2005), and represents the unique identifier of a record. Correctly generated, such a UUID is considered to be sufficiently individual that this identifier will be unique across all EpiJSON datasets and greatly simplifies aggregation and sub-setting operations. Should it be necessary to keep another identifier from an existing system then this should be stored as an attribute of the record. The "attributes" key holds an array of "attribute" objects (as above). These are the attributes of this specific record object. The "events" key holds an array of "event" objects (noted "event[]") relating to this record. Events are usually distinguished from attributes by occurring in time or space while attributes tend to have no spatio-temporal component. For example a date of occurrence is an event, it has a temporal dimension (and spatial-but this is rarely required) while the sex of an individual is an attribute as it lacks either a spatial or temporal dimension.

2.2.3. Events

The "event" object records observations made on its parent object. These might be events directly related to disease such as infection, symptom onset, hospitalisation or they might be more generic such as date of birth, the locations from an individual's travel history, their place of work, or their home. Events must have a time, a place or both; for recording of time RFC3339 (Newman and Klyne, 2002) is used (e.g. "2014-12-12T00:00:00Z") and we use GeoJSON for recording of location (Butler et al., 2008). For datasets where the unit-of-record is not the individual, events could be data such as census dates and the resultant population counts, dates of ward cleaning etc.

An event object has up to five keys: "id", "name", "date", "location" and "attributes" (Table 5). The keys "id", and "name" are mandatory. In addition, at least "date" or "location" must be provided. The key "attributes" is optional. The value of the "id" key, as for the record object, should be a string consisting of a version 4 UUID conforming to and generated as specified in RFC 4122 (Leach et al., 2005). "Name" is a string that names the event. The value of "name" need not be unique, indeed it is suggested that a standard set of names are used to identify events across the dataset (and across multiple datasets). The "date" key records the time at which an event took place; it should be a valid string representation of a RFC3339 (Newman and Klyne, 2002) conformant date, this is a

Table 4
keys and values for the record object. Square brackets following a data type indicate arrays.

Key	Type	Value	Description
"id"	string	String representing a UUID	A string conforming to RFC 4122 acting as a unique identifier for this record
"attributes"	attribute[]	Zero or more attribute objects	Attributes relating to this record. Examples might be gender, county name etc.
"events"	event[]	Zero or more event objects	Events relating to this record. Examples include infection, symptom onset or travel history etc.

Table 5
Keys and values for the event object.

Key	Type	Value	Description
"id"	string	String representing a UUID	A string conforming to RFC 4122 acting as a unique identifier for this event
"name"	string	String	The name of this event
"date"	string	A string representation of a date	This string must conform to RFC3339 (as above for the attribute object)
"location"	object	A GeoJSON object	This object represents the spatial occurrence of the event as valid GeoJSON object
"attributes"	attribute[]	Zero or more attribute objects	Attributes relating to this event

subset of the ISO 8601 extended format. If a single date point is insufficient to record an event, for instance to record a period of exposure, then we suggest that multiple events are recorded; in this example, one event object is coded to record the start of exposure and one to record the end.

The "location" key stores a geospatial object in GeoJSON format (Butler et al., 2008) representing the location of an event. By allowing different events to have different geospatial types it is possible to choose the most appropriate geospatial object for an event without being restricted by the choice within other events (i.e. it would be valid to record the only vaguely known location of infection as a large polygon, while simultaneously having a point to represent where an individual was admitted to hospital). This scheme also permits the use of different projections for different events (e.g. a national/local grid for some while using WGS 84, the GPS coordinate system, for others). The "attributes" key holds an array of attributes defined in the same way as above. Here the attributes relate to the event and may hold data such as number of colony-forming units from a swab, genetic sequence data or the recorded population values from a census.

2.3. Example

In this example, an individual record is presented (Fig. 3). First there is the metadata; here simply who created this dataset. Then there comes the actual records, in this case only one person is recorded. This person has an age, and an ID from the field recording system. A single event, the onset of disease, is also recorded together with the patient's temperature at that point. The source data could have been presented as a comma delimited file (csv) or line list format but by using EpiJSON we can assign various parts of the line list data to objects in the JSON format to reduce ambiguity and provide information to other users of this information: in this case, in what units we should interpret the temperature reading. We also draw the reader's attention to the more fully worked example of a data storage and an analytical system communicating via EpiJSON in the supplementary material.


```
{
  "metadata": [
    {
      "name": "creator", "type": "string", "value": "Hackout2 team"
    }
  ],
  "records": [
    {
      "id": "22D15BC2-6C2C-11E5-BAAE-3D122D9BAEF2",
      "attributes": [
        {
          "name": "age", "type": "integer", "value": 10
        },
        {
          "name": "fieldID", "type": "string", "value": "A3"
        },
        {
          "name":"eye-colour", "type":"string", "value":"blue"
        }
      ],
      "events": [
        {
          "id": "31309E38-6C2E-11E5-A91A-585A358CC448",
          "name": "disease onset",
          "date": "2014-12-12",
          "attributes": [
            {
              "name": "temperature", "type": "number", "value": 37.7, "units": "degC"
            }
          ]
        }
      ]
    }
  ]
}
```


Fig. 3. Example of data in EpiJSON format.

2.4. The repijson package

The repijson package has been developed as a demonstration implementation and will facilitate data transfer to and from the EpiJSON format within the statistical and programming software R (R Core Team, 2014). It provides a variety of functions that can convert data to each of the levels within EpiJSON (metadata, attributes, records, events and objects). It also implements conversion tools for the data structures used to store outbreak line lists in OutbreakTools (Jombart et al., 2014). Fig. 4 provides a brief overview of the conversions and translations provided by the package. The package is released with a fully documented manual and a vignette tutorial which includes the above examples to help users implement the formatting system. It also illustrates how information from EpiJSON files can easily be extracted for further analysis.

The software package, repijson, is distributed under GNU Public Licence (version 2 or greater), and developed on github (https://github.com/Hackout2/repijson), where instructions on installation and contributions can be found. The stable version of the package is distributed on the Comprehensive R Archive Network (CRAN: http://cran.r-project.org/).

2.5. Technical details of EpiJSON

JSON, JavaScript Object Notation, ECMA standard 404 (Bray, 2014), was chosen as the base technology for a universal

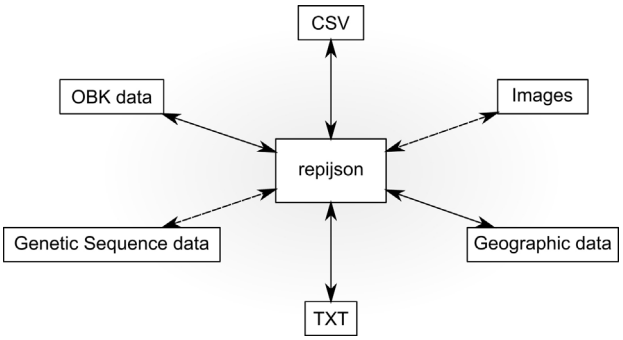

Fig. 4. Schematic of the common types of data that the repijson package may convert between or include into an EpiJSON file. Solid lines indicate conversion of data dashed lines indicated direct inclusion.

epidemiology interchange format because: (i) it is lightweight with no complex formatting rules; (ii) it is based on plain text; (iii) it is easy for both humans and machines to read; (iv) it is in widespread use; and, (v) it has a large number of high quality libraries available for all common computer systems and languages. All of these properties combine to make JSON an ideal data-interchange medium.

As EpiJSON is standards-compliant JSON, a file is made up from a series of key value pairs. Keys are always double quoted character strings while the "value" may be a character string, a number, a Boolean value or another JSON object. A pictorial overview of this structure for EpiJSON is presented in Fig. 2.

As a detail for implementations, where possible, numeric types should be equivalent to C's long type (i.e. in the range −2147,483,647 to +2147,483,647) for integers and double precision should be used for floating point values. For clarity these have been referred to using JSON terminology as integer and number types, respectively, throughout the manuscript.

Within an attribute object the value of the units key where possible should be a unit name taken from the name parameter of the matching unit definition in the UDUNITS2 unit database (Unidata, 2015a). When the type key is "integer" or "number" and the units key is omitted, the value should be considered to be non-dimensional.

3. Discussion

The major benefits of the EpiJSON format are its flexibility and simplicity. EpiJSON has broad-scale application to data transfer across multiple disciplines as we reach an era of rapid data assimilation. EpiJSON has been designed to take advantage of existing standards for the data that it represents. This has two major advantages: the first is that existing domain expertise, for example in the representation of spatial data, is implicitly incorporated into this standard; the second is that the implementation of EpiJSON parsers and filters for existing languages and software is greatly simplified.

From the outset, EpiJSON has been designed to reduce ambiguity and permit greater ease of transmission for epidemiological datasets. Key to this ambition is ensuring that core information held by the dataset may be simply understood and can be transferred with fidelity. To provide documented data of the greatest clarity and concision within EpiJSON we offer the attribute object. This object not only holds much of the fundamental data of the dataset but also important metadata that enables the interpretation of those data. The repetition of identically constructed attribute objects throughout the data structure allows clear documentation of information at all levels of a dataset without the added confusion of a different structure at each level.

The first level within EpiJSON is the dataset. Information on the dataset itself is stored as metadata. The use of metadata is standard in many epidemiological settings and providing a method for this to be included within EpiJSON is essential and best practice. Indeed, including metadata allows some of the most important pieces of information relating to a dataset to be stored alongside the data, such as why, when and by whom the dataset was created or collated. Similar to the epidemiological data, the type and detail of metadata can be very broad.

The next level is that of the record. A great asset of our format is that EpiJSON makes no assumptions about the unit-of-record (that is person or region, country, etc.) nor even forces record objects to be uniform. Although not recommended, it is perfectly possible to mix individuals with regions within a single dataset.

The final level is that of the event. So that there is no ambiguity within the data, EpiJSON requires the use of a standard for the recording of time and place. Our decision to enter location data using GeoJSON format (Butler et al., 2008) permits recording of event data to any of the standard geographical objects (point, line or polygon) and simultaneously solves potential issues caused by using different Coordinate Reference Systems (CRS). An additional benefit is that different events within the same record may be recorded to a different geographical object, even one in a different CRS.

Naturally, no standard epidemiological software currently supports the EpiJSON standard. The first step in more widespread support is to provide a mechanism by which the validity of a EpiJSON file and hence software implementation can be checked. To this end we provide a schema for EpiJSON files (see links below). In the short term the availability of an R package permits not only EpiJSON files to be used within R but also for them to be converted to and from a wide range of file formats including the ubiquitous csv spreadsheet format read by most systems. The next step is for library functions for common languages to be written, easing the developer effort for high level packages familiar to the epidemiological user base. We believe that, because the standard is well defined, open, based on existing standards and easily validated, this effort in including parsers within existing packages should be small.

The transfer of epidemiological data may be particularly sensitive either because of its personal or political nature. However, we believe that encryption is a different problem to that of high fidelity transfer of scientific information. Performing encryption well is difficult, as evidenced by the many security breaches in even extensively tested systems over recent years. We believe that the user is better served by well tested external encryption libraries or tools than by providing a mechanism within the EpiJSON standard that would become rapidly obsolescent. However we envisage that much as HTML (HyperText Mark-up Language, the text format of most Web pages) may be encrypted using HTTPS, the text based EpiJSON could be encrypted using a system such as ssl to provide an encrypted version of the format. Such a development would also provide the user with the choice of symmetric, password-based encryption as used in later versions of well-known spreadsheet programs or the more secure key-based asymmetric encryption commonly in use on the internet in banking and encrypted email transactions.

Representing data as EpiJSON means that increased storage space is required in comparison to equivalent, terse, spread-sheets. Yet the additional detail of EpiJSON, the reduction in ambiguity and the direct readability by machine outweigh the disadvantage of increased use of storage. Further, it is possible to transfer genomic data, images and location data within a single EpiJSON file ensuring that a wide variety of data, relating to a single unit-of-record, can be collated. As with encryption, it is not within the scope of a format for epidemiological data to specify a data compression standard but note that, as a structured text document with a good deal of repetition, EpiJSON files compress well using commonly available compression tools. EpiJSON's requirement for additional space is not considered an impairment to its applicability.

Finally, the EpiJSON format is deliberately broad to permit the capture of the widest range of epidemiological datasets. However, we recognise that there are sub-types of epidemiological data. It is envisaged that standard attribute sets will be agreed to address specific types of epidemiological data, much as conventions have emerged for NetCDF datasets (Unidata, 2015b) e.g. the CF convention for climatology (Eaton et al., 2011). The adoption of well-defined data standards makes both the contribution of data to communal efforts and the development of data tools much easier. This has been the case for public transport data (Google, 2015), where a standard allows companies to submit their timetables in a form that can be used by Google Maps to provide route information. This definition of an open data structure for epidemiology is the first step in allowing other developers and collaborators to modify their software and work-flows to utilise the standard. The development of useful tools and practices will, by necessity, be an iterative process with input from a wide range of users.

4. Conclusion

In EpiJSON we provide a well-understood file structure with a verifiable format for storing and exchanging epidemiological data. The EpiJSON format is highly flexible to enable the widest range of datasets to be conveyed while being sufficiently rigid to remove many of the common causes of ambiguity and error.

While EpiJSON is not intended to replace existing databases or surveillance systems, it should prove useful for transferring information between these types of systems, collections and analysis tools. In the resources below we provide links to the current EpiJSON schema (permitting the automatic verification of EpiJSON files), tools to convert between common formats and EpiJSON and example datasets in EpiJSON form.

We appreciate that full implementation of the EpiJSON standard across all of the software and systems used within epidemiology is a large undertaking, but by providing an open blueprint for data interchange between these systems, together with validation tools and a library for one of the more popular data analysis environments we hope to greatly simplify the process by which epidemiology systems interact. Researchers and institutions should find that collaboration becomes easier by reducing data compatibility problems and additional capabilities will become simpler to add to existing workflows with the adoption of EpiJSON as a medium for information interchange.

5. Resources

5.1. Licence

GNU General Public Licence (GPL) ≥ 2.

5.2. Website

https://github.com/Hackout2/EpiJSON.

5.3. JSON schema

https://raw.githubusercontent.com/Hackout2/EpiJSON/master/schema/epijson.json.

5.4. Tools

R library in CRAN: https://cran.r-project.org/web/packages/repijson/.

5.5. Development version

https://github.com/Hackout2/repijson.

Acknowledgements

EpiJSON was created during "*Hackout 2: Graphical Resources for Infectious Disease Epidemiology in R*" (https://sites.google.com/site/hackout2/), an event funded by the Medical Research Council (MRC) Centre for Outbreak Analysis and Modelling and the National Institute for Health Research (NIHR) Health Protection Research Unit (HPRU) for Modelling Methodology (NIHR-HPRU-2012-100-80). We thank Neil Ferguson, Zak Kadrou and Susannah Fisher for their invaluable help hosting and organising this event. TJ is funded by MRC and NIHR. AS is partly funded by the Liverpool School of Tropical Medicine.

This work has been partially funded by the EC Research Executive Agency in the 7th Framework Programme, (SEC-2013.4.1-4: Development of decision support tools for improving preparedness and response of Health Services involved in emergency situations) under grant number FP7-SEC-2013–608078—IMproving Preparedness and Response of HEalth Services in major criseS (IMPRESS).

TF would like to thank Tomasz Zatorski and Matthew Bull for their technical advice and sensible suggestions and Ian Hall and Steve Leach for their support of this work.

We would also like to thank the two anonymous reviewers whose constructive comments helped to strengthen this manuscript.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at doi:10.1016/j.epidem.2015.12.002.

References

Aanensen, D.M., Huntley, D.M., Feil, E.J., Al-Own, F.'a, Spratt, B.G., 2009. EpiCollect: linking smartphones to web applications for epidemiology. Ecology and community data collection. PLoS ONE 4, e6968, http://dx.doi.org/10.1371/journal.pone.0006968.

Bray, T. (Ed.), 2014. The JavaScript Object Notation (JSON) Data Interchange Format. RFC 7159. The JavaScript Object Notation (JSON), http://dx.doi.org/10.17487/RFC7159.

Butler, H., Daly, M., Doyle, A., Gillies, S., Schaub, T., Schmidt, C., 2008. GeoJSON Specification [WWW Document]. GeoJSON Format Specif, ⟨http://geojson.org/geojson-spec.html⟩ (accessed 4.17.15).

Cauchemez, S., Fraser, C., Van Kerkhove, M.D., Donnelly, C.A., Riley, S., Rambaut, A., Enouf, V., van der Werf, S., Ferguson, N.M., 2014. Middle East respiratory syndrome *coronavirus*: quantification of the extent of the epidemic, surveillance biases, and transmissibility. Lancet Infect. Dis. 14, 50–56, http://dx.doi.org/10.1016/S1473-3099(13)70304-9.

Eaton, B., Gregory, J., Drach, B., Taylor, K., Hankin, S., Caron, J., Signell, R., Bentley, P., Rappa, G., Höck, H., Pamment, A., Juckes, M., 2011. NetCDF Climate and Forecast (CF) Metadata Conventions (http://cfconventions.org/latest.html) (accessed 04.17.15).

ECDC, 2015. Legionella Outbreak Toolbox [WWW Document]. Legion. Dis. Outbreak Investig. Toolbox, ⟨http://legionnaires.ecdc.europa.eu/⟩ (accessed 4.17.15).

Fraser, C., Donnelly, C.A., Cauchemez, S., Hanage, W.P., Van Kerkhove, M.D., Hollingsworth, T.D., Griffin, J., Baggaley, R.F., Jenkins, H.E., Lyons, E.J., et al., 2009. Pandemic potential of a strain of influenza A (H1N1): early findings. Science 324, 1557.

Google, 2015. General Transit Feed Specification Reference [WWW Document]. Google Developers, ⟨https://developers.google.com/transit/gtfs/reference⟩ (accessed 6.4.15).

Jombart, T., Aanensen, D.M., Baguelin, M., Birrell, P., Cauchemez, S., Camacho, A., Colijn, C., Collins, C., Cori, A., Didelot, X., Fraser, C., Frost, S., Hens, N., Hugues, J., Höhle, M., Opatowski, L., Rambaut, A., Ratmann, O., Soubeyrand, S., Suchard, M.A., Wallinga, J., Ypma, R., Ferguson, N., 2014. OutbreakTools: a new platform for disease outbreak analysis using the R software. Epidemics 7, 28–34, http://dx.doi.org/10.1016/j.epidem.2014.04.003.

Josefsson, S., 2006. The Base16, Base32 and Base64 Data Encodings. RFC 4648., http://dx.doi.org/10.17487/RFC4648.

Leach, P., Mealling, M., Salz, R., 2005. A Universally Unique IDentifier (UUID) URN Namespace. RFC 4122., http://dx.doi.org/10.17487/RFC4122.

Newman, C., Klyne, G., 2002. Date and Time on the Internet: Timestamps. RFC 3339., http://dx.doi.org/10.17487/RFC3339.

R Core Team, 2014. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria.
Unidata, 2015a. Unidata | UDUNITS [WWW Document], ⟨http://www.unidata.ucar.edu/software/udunits/⟩ (accessed 4.17.15).
Unidata, 2015b. NetCDF Conventions [WWW Document], ⟨http://www.unidata.ucar.edu/software/netcdf/conventions.html⟩ (accessed 4.17.15).
WHO Ebola Response Team, 2015. West African Ebola epidemic after one year—slowing but not yet under control. N. Engl. J. Med. 372, 584–587, http://dx.doi.org/10.1056/NEJMc1414992.
WHO Ebola Response Team, 2014. Ebola virus disease in West Africa—the first 9 months of the epidemic and forward projections. N. Engl. J. Med. 371, 1481–1495, http://dx.doi.org/10.1056/NEJMoa1411100.