Open Geospatial Consortium

Date: 2013-03-02

Reference number of this document: OGC 12-164

Category: Technical Note

Editor: Clemens Portele, Satish Sankaran

GeoServices REST API – RFC comments

Warning

This document is not an OGC Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard.

Contents

Page

1Overview.....
1.1Scope.....
1.2General remarks.....
2Discussion of key topics.....
1.3Overview.....
1.4Drop GeoServices REST API and concentrate on developing new RESTful profiles of the existing OGC Web Service standards.....
1.5Using GeoJSON.....
1.6RESTfulness of the GeoServices REST API.....
1.7The title GeoServices REST API of the series and the titles of each part.....
1.8Phased Approach.....
1.9Spatial Reference.....
1.10Handling cross-domain issues.....
3Comments.....

GeoServices REST API – RFC comments

1 Overview

1.1 Scope

This document includes all comments received on the GeoServices REST API family of candidate standards during the public comment period and documents how the GeoServices REST SWG has resolved the comments.

1.2 General remarks

The OGC Policies and Procedures state the following:

"Once the RFC comment period closes, the RFC submission team "collects" the comments and integrates them into a single RFC comment document. The SWG reviews the comments and makes a decision as to the fate of the RFC. If the SWG decides that comments received are sufficient to halt the RFC, then the RFC "fails" and adoption of the proposal halts. The submitter(s) may then make changes and resubmit the RFC proposal.

Document type:	Technical Note
Document subtype:	n/a
Document stage:	Proposed
Document language:	English

If, however, the comments received do not cause the SWG to halt the RFC, then the SWG edits the document based on the comments received during the comment period."

I.e., as the first step, the SWG had to decide about the general way forward. The following motion was approved in an electronic vote (Yes: 7, No: 0, Abstain: 0, Quorum: 5 votes):

> MOTION: Comments that have been submitted during the public comment period proposing changes to the candidate standard that would break backward compatibility with current implementations of the OWF GeoServices REST Specification v1.0 are out of scope for this version. Such comments will be retained for consideration for later versions of the standard.
>
> NOTES:
>
> - This motion is not strictly necessary as it simply reiterates a statement from the SWG charter, but the SWG decided in the SWG Telecon on Oct 2, 2012 to explicitly confirm the handling of comments that break backward compatibility in a motion.
> - The document with all the RFC comments will contain these comments and a discussion of key issues raised in these comments, so that TC members can fully consider these comments in their adoption votes.
> - Some comments fall partly in this category, but contain also aspects that can be addressed in this version. Such aspects will still be discussed by the SWG in the comment resolution process.
> - Comments that break backwards compatibility are input to a future **major** revision (consistent with OGC policies).

The next step has been to address the remaining comments following the process shown in Figure 1 below.

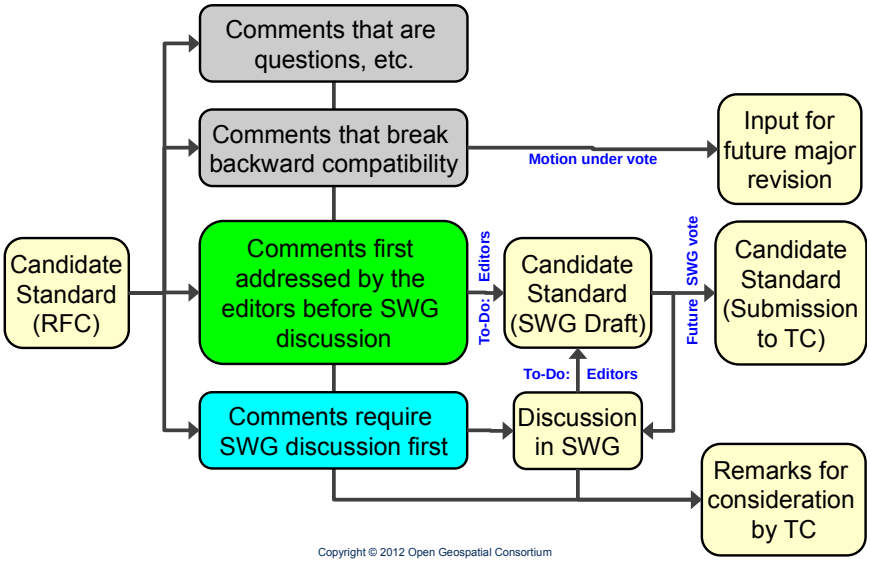

Figure - Process

As a result, this document reflects the different viewpoints about the value proposition that this candidate standard offers or not to the TC. It is up to the OGC TC and PC members to decide in

their adoption votes, if they consider the GeoServices REST API valuable to the wider community or not.

2 Discussion of key topics

1.3 Overview

Looking at the comments received, we can identify the following categories of issues that have been raised:

- Use of other standards:
    - "OGC Baseline": Drop GeoServices REST API and concentrate on developing new RESTful profiles of the existing OGC Web Service standards
    - "GeoJSON": Use GeoJSON for encoding features and geometry
    - "JSON vs XML": XML should be used instead of JSON
    - "Spatial Reference": WKT should use standard WKT
- REST-related topics:
    - "REST": Various issues related to the RESTfulness of the GeoServices REST API (e.g. use HTTP verbs, definition of resources, absence of links / hypermedia controls, HTTP error codes, Mime types, Content negotiation, discovery of query parameters without using out-of-band information)
- Standards writing:
    - "Title": Use a better title, do not use GeoServices, REST, and/or API in the title; also avoid the use of Map Service or Feature Service
    - "Modular Spec Policy": Unclear requirements and tests, conformance to the modular specification policy
- Process:
    - "Phased Approach": Standardise the specifications step-by-step
    - "Reference Impl.": Reference implementation required
- Other:
    - "Security": Some aspects of the GeoServices REST API raise security concerns
    - "Errors and clarifications": Errors in the drafts or cases where a clarification in the text is required
    - "Questions": Questions related to the candidate standard
    - "Specific Changes": Specific changes or additional capabilities are requested

Before looking at the individual comments, the most important topics that are raised in a number of comments here are discussed from the view of the editors. Other comments and topics are directly addressed in the context of each comment.

1.4 Drop GeoServices REST API and concentrate on developing new RESTful profiles of the existing OGC Web Service standards

Regarding standards philosophy, the specification initiators are big believers in multiple standards rather than in "one big standard to rule them all". It is ok to have different ways to encode geometry (well-known text and binary, ArcGIS XML, KML, GML, GeoRSS, GeoJSON, Google places, etc.) and multiple ways to 'discover datasets' (infinite variants from many different communities). The important thing is that the standards are associated with well-defined software engineering "contracts" that have robust implementations in the real

world. The GeoServices REST API specification endeavours to meet that requirement - well-defined software engineering "contracts" that have robust implementations in the real world.

In some sense, the fundamental goal of the submitters is to deliver (today) a well-defined, widely implemented and adopted RESTful interface – starting with a huge community of users. There are obviously other approaches that could involve trade-offs based on technical, political and timing considerations. We could all build a new different version of the GeoServices interface that gets more people on-board as co-creators. But that's not the intent behind this submission.

This submission like submissions of other works into the OGC (KML, netCDF, etc.) inherently depends heavily on „backwards compatibility". The „major" value here is that this specification in its current form boasts of tens of thousands of robust instances running with real data. Breaking backwards compatibility in any serious way would remove one of the primary „value propositions". The first version of this standard therefore provides a restrictive scope for a „collaborative ownership" of design decisions. What it offers though is an opportunity for the community to extend this framework to meet future unmet needs, and for OGC members to adopt such extensions in future revisions of the standard.

As it is a key point of the submission that the GeoServices REST API is a web service interface that is used and supported by many operational systems today and represents a de facto standard, the GeoServices REST API brings a number of benefits to the community:

- Unlike many of the OWS standards where the development in the SWG has sometimes been supported with parallel partial prototype implementations in OWS testbeds or other projects, the complete GeoServices REST API has been implemented, tested and is already used in large scale operational deployments. Its existing functionality has been borne out of feedback from a large community of GIS users. This standardization effort will now allow committed OGC members to play a lead in this wider ongoing discussion within the GIS community about possible enhancements to the GeoServices specification. Many of the major organizations who are users of the current GeoServices REST specification are also valued members of the OGC. It is expected that the adoption of the specification by OGC provides all these members and other members within OGC an opportunity to actively dictate future directions for this framework.
- A significant amount of services exist that provide geographic information via this API today, both for high quality base data (global imagery and topographic data) provided by Esri and for additional data by a wide range of data providers.
- A large number of clients exist already that can communicate with those services without any programming. One of the highlights of the REST specification is the value of the JSON binding it employs. For those that do not have an established client, it is extremely simple to ingest the JSON responses into their existing applications. For now, it is also true that Esri provides a whole suite of client libraries that can help people build powerful end-user applications that can ingest responses for the many Geoservices. Currently OpenLayers[1] and GDAL (Starting with OGR 1.8.0, the GeoJSON driver can read the JSON output of Feature Service request following the GeoServices REST Specification[2]) also offer tools that help users connect to and use many of the geoservices. interactive instruments has also added support for the GeoServices REST API Feature Service in its XtraServer product in addition to WFS. interactive instruments has been and is a big supporter of the OGC WFS standard and also provides one of the first implementations that support the latest WFS 2.0

1" http://www.mkgeomatics.com/apps/REST_Dev/

2" http://www.gdal.org/ogr/drv_geojson.html

standard. However, currently, many customers already use software that understand the GeoServices REST API, but not the full range of OWS standards, so the GeoServices REST API brings immediate interoperability benefits to them.

- Esri has created a wide ranging set of client tools on multiple platforms capable of working with these services. This has clearly shown the value of the API and also the level of field-testing this API has had based on the fact that client applications on multiple platforms (web applications, mobile and desktop) have been built to work with these services. With GDAL now supporting the GeoServices's Feature Service and the possibility of more open source solutions (for working with GeoServices) appearing on Github in the near future, there will be multiple libraries available for easily working with the GeoServices specification.

Breaking backwards compatibility with the current implementations in a OGC GeoServices REST API standard would defeat key benefits of standardising this API as no existing services would comply to the standard and no clients would exist that could talk to services implementing the standard.

It is the opinion of the submitters that the value offered by a backward compatible GeoServices REST API as an OGC standard is larger than the risk imposed by its standardisation as a separate set of standards in addition to the OWS standards.

The topic of competing or overlapping OGC standards was discussed in 2011 by the OAB, PC and OGC Board of Directors with the result that it was explicitly clarified that "competing or overlapping standards are OK as long as each standard is properly positioned in terms of the interoperability problem solved"[3]. Like it is normal in today's IT environment that there are multiple representations of a resource (e.g. XML, JSON, HTML, Shapefile, DXF) it is also quite common to support multiple service interfaces to access the same resource to target different users or use cases. As a vendor, Esri has always valued interoperability, and by the same token also respects that no one size „fits all". At a product level (for Esri), this translated to server, client and mobile technologies that can deal with RESTful, OWS and SOAP based interfaces (service protocols) and also the ability to deal with a wide variety of formats – KML, GML, JSON, DXF, etc. This is an „inherent reality" that is also an affirmed business need. As we move towards a world of mashups, it is common to see people use multiple interfaces and deal with multiple formats as they bring together their final application. It will not be uncommon for people to access the GeoServices REST specification along with other established OGC specifications to create a web map to tell their „geographic story". For example, in ArcGIS Online[4], web maps are compiled from a variety of sources and formats / interfaces including OGC KML, OGC WMS and GeoServices REST API services as well as CSV files.

At this time the API has mainly seen adoption from Esri and its partners, but it is expected that other vendors will add support for the API in the future. Organizations like EXELIS VIS,

3" From the minutes of the Board of Directors (9/21/2011):

"The question was then asked, what is OGC's stance on overlapping standards, and initial thoughts expressed that it would require a new thinking process to evaluate fit and assess the overlap. Next, the questions: are we trying to system engineer innovation, and why is overlap a problem? The user has freedom to choose – and request improvement through the change request process. Basically, the consensus was that competing or overlapping standards are OK as long as each standard is properly positioned in terms of the interoperability problem solved. It comes down to positioning and communication.

"The moderator summarized the opening question – how does OGC relate to a changeable world, and summarized the answer – by creating potential for different levels of harmonization. He added that communication is a key aspect of moving forward, from a polished, interoperable world with a complex architecture into a world with competing standards and services, while still paying attention to interoperability."

4" http://www.arcgis.com/

Arc2Earth and interactive instruments have adopted the GeoServices REST API specification on the server side. Open Source organizations also have announced plans[5] for supporting the GeoServices REST API within their open source platform. Trimble is another organization currently implementing the GeoServices specification on their TIMIS server. On the client side, there are innumerable implementations of applications that have been created using both Esri and non-Esri client tools that can consume services using the GeoServices REST API. OpenLayers and GDAL currently have libraries that can be used to build applications that consume the GeoServices REST services. Additionally, the specification is currently an open specification and being HTTP based with a JSON encoding, it is very easy for most developers to leverage these services within their „locations based“ applications.

As pointed out in the document 12-062r2 (GeoServices REST API relationship with the OGC baseline), the OWS standards and the GeoServices REST API are seen as complementary and the submitters see no issue with them continuing to co-exist. The submission team supports the continued development of the OWS standards, including adding RESTful bindings, and is an active participant in many of those SWGs. Eventually providers and users will use the interfaces and representations that are most useful to their specific use case.

The key aspects where the GeoServices REST API differs from the current OGC baseline standards are (this is mostly material taken from 12-062r2):

- The services use JSON as the standard encoding of data. Support for a JSON encoding is so far not standardized within the relevant standards of the OGC baseline.

  While full JSON support can be added to the existing OGC Web Service standards (and the software products implementing them), this will take considerable time to specify and test all JSON schemas of the operation messages, content representations and path expressions (like the Xpath in Filter expressions that depend on the GML encoding).

- The services of the GeoServices REST API have been developed as part of a single specification and are part of a consistent framework.

  While it would be possible to develop new versions of the OWS standards using a consistent framework and with support for JSON representations and a RESTful "binding", this will likely take significant time due to the unresolved REST-related discussion items, the current organization of OGC SWGs based on the individual standards and the fragmentation into separate standards.

  On the other hand, the GeoServices REST API offers a proven specification that is available today and which is used operationally in a large number of applications. I.e., it provides an existing network of services, data and applications that other implementations and deployments can build upon.

- The current OWS standards usually provide specific capabilities often targeted to comprehensively cover a range of complex use cases and requirements. The success of the OGC standards has shown that they clearly address a market requirement. However, for many use cases a simpler set of requirements is sufficient – and the GeoServices REST API addresses this market. This leads to some restrictions compared to capabilities included in other OGC standards (see below). Support for more complex use cases is traded for simplicity and in some cases improved performance. Simplicity in the sense that both the

5" http://info.opengeo.org/rs/openplans/images/OpenGeo_TDS_Followup_20120808.pdf

development of client software and the configuration of services will typically be relatively simple.

A key difference between the GeoServices REST API and the OGC and ISO 19100 standards is that the underlying data model is firmly based on the data model of an underlying database that supports SQL and Simple Feature for SQL (SF-SQL). The OGC standards on the other hand in general use models that are based on the capabilities of UML or XML Schema.

In implementations, this makes a significant difference. The implementation of encoding data in JSON or executing queries according to the GeoServices REST API is straightforward, if a SF-SQL-enabled database is used for data storage. On the other hand, WFS implementations that use a SF-SQL-enabled database for data storage (and this seems to be the most typical approach in practice) will have to implement a non-trivial mapping between XML Schema and the data model in the database – both for encoding the data in XML as well as to transform queries to SQL. As WFS provides no restrictions on the use of XML Schema, a conformant implementation in principle will have to support it all.

Similar issues exist on the client side where commonly used GIS clients are often capable of consuming data encoded using tabular structures, but not data that uses more complex data structures as supported by UML or XML Schema. However, if a client accesses a WFS it has to be prepared to consume data according to XML Schemas of any complexity and quite often this is not the case today. The level 0 of the GML Simple Features Profile is an attempt to address this issue, but to do this properly there would need to be a WFS standard (or conformance class) that restricts the model complexity accordingly. The same applies for WMS that support GetFeatureInfo or SLD. Etc.

There is without doubt a significant market demand for more expressive or complex models and the standards of the current OGC baseline support this demand well. But where this is not required, it is faster, simpler and cheaper to develop and deploy services and clients using data models aligned with the underlying data storage technology.

This is discussed in more detail in document 12-062r2.

As the comments show, other views exist and comments express the concern that the GeoServices REST API as an OGC standard would be negative for OGC. OGC members that share this view should vote against adoption of the candidate standard in the TC and PC adoption votes. If this view is shared by the majority of the OGC members, then this would show a consensus that OGC members would prefer to concentrate on the current OWS standards – maybe with RESTful bindings in the future – as the only type of geospatial web service standards and GeoServices REST API should not become an OGC standard.

1.5 Using GeoJSON

As the initial development of the GeoServices REST API predates the GeoJSON development, the GeoServices REST API extensions to JSON to add well-known JSON representations of geometries and features differs from those in GeoJSON. For backwards compatibility reasons, GeoServices JSON cannot be replaced by GeoJSON, see 2.2.

However, the difference between the GeoJSON and the GeoServices JSON encoding of geometries and features is relatively small and transformation of GeoServices JSON to GeoJSON is not difficult (sample JavaScript functions are available on github[6]).

Support for GeoJSON could be added (and probably should be added, in particular if GeoJSON becomes an OGC standard) in additional conformance classes – either in a future (minor) revision, in a separate GeoJSON extension or even as part of the GeoJSON standard.

1.6 RESTfulness of the GeoServices REST API

The GeoServices REST API is based on both RESTful principles as well as pragmatic considerations that were and are driven by support for various aspects of the HTTP protocol in commonly used environments like JavaScript, Adobe Flex or Microsoft Silverlight, in commonly used web browsers, or in proxies. These typically do not offer complete support for all of the HTTP standard. As a result, several comments raise issues regarding the RESTfulness of the API. The table below provides a rationale for deviating from a "pure" REST design. The long and unfinished debates about what is REST or RESTful and what is not, both within OGC and in the wider community, illustrate that this is a very fuzzy area.

At this time, OGC does not have any consensus on what would be requirements for a RESTful service standard within OGC. WMTS has a REST binding and a REST binding has been proposed for WFS that is currently being processed in the WFS SWG. In parallel the RESTful Service Policy SWG is working on developing the requirements for future RESTful OGC services, but no consensus has been reached even within the SWG.

We should take into consideration, too, that it is a fact that a large number of the "REST API"s (programmableweb.com alone has registered 4600+ "REST API"s[7]) would not be considered following the REST principles by Roy Fielding. The term "REST API" is as much a marketing term as it provides an idea about the general underlying architectural style. The GeoServices REST API wants to reach a large number of web developers and meet their expectations in providing a web service interface that is easy to use for them and follows standard web development practices. As a result, the GeoServices REST API indeed deviates from a "pure" REST style in some aspects, but provides a well-defined hierarchy of resources, each with associated URIs and representations, which is at the core of the REST principles. http://www.restfulwebservices.net/ provides an exhaustive collection of RESTful API's available on the web today. While it may not be considered definitive nor authoritative it does provide the reviewer a sense of where the market is when it comes to defining RESTful services.

Note that if the potential future REST bindings of the OWS standards would follow the REST principles very closely (although arguably the existing WMTS REST binding lacks there, too, but could be improved), then this would add another aspect where those standards are complementary to the GeoServices REST API.

The pragmatic considerations taken in the GeoServices REST API have in general already been documented in chapter 6 of Part 1: Core. Where comments point to aspects that are not yet covered in that chapter, these aspects will be covered, too. The following table summarises the main aspects raised in the comments.

6" for example: https://github.com/odoe/esritogeojson or https://github.com/Esri/geojson-utils/tree/master/src

7" http://www.programmableweb.com/apis/directory/1?protocol=REST

Aspect raised in comment	Pragmatic considerations based on common industry practice
Use of HTTP methods, in particular that PUT and DELETE are not used	As sections 6.2.1 and 6.2.2 of Part 1 (Core) describe, some web browsers, proxies and web development frameworks support HTTP methods only to the extent specified by HTML (GET plus POST with URL-encoded or multi-part content), sometimes because this is what is commonly used, sometimes due to security considerations. Requiring support for PUT and DELETE would make it difficult for some developers to use the API.
Use of HTTP POST for retrieving a resource representation	The use of HTTP POST is needed whenever the size of the URL may be longer than 2048 characters. This isn't usually a factor for most API designs, but in the context of geographic information it happens quite frequently (serialized geometries can easily be bigger than 2000 characters).
Not using HTTP error codes	Cross-domain scripting needs have been mostly solved using support for JSONP (JSON with padding). These make it impossible to support all the HTTP error codes. As discussed in Clause 8 of Part 1 (Core), most responses with JSON content will use an HTTP status code 200 and the JSON content will either be a resource representation, the result of a controller resource operation or an exception. If the status code is not 200, the browser's network stack rejects the response from the server, and the client call-backs are never even called. To overcome this, it is common practice to respond with errors wrapped inside the response. Example: Specific issues using the HTTPService Component in a FLEX based app development environment: a. You can only do a GET or a POST, no PUT or DELETE b. The HTTP status code returned in the response is not available (you cannot get the id of a newly created resource from the 'Location" header and you cannot tell the difference between a '500 internal server error' , a '404 not found' or a '422 validation error' c. There's no way to get the response body for anything not in the 2XX range (it fires a fault event for any HTTP response that does not have a status code of 200).
Not using HTTP content negotiation and the standard media types	The media types are advertised using query parameters in the URL (e.g., "?f=json") rather than using the HTTP headers. This too can be attributed to the fact that sometimes proxy servers tend to strip out header information and hence a more practical/safer approach of using parameters in the URL has been used for this purpose. Another aspect is that including the representation as part of the URI enables "clickable" URLs that may be copied in email, cut and pasted into browser address bars, etc. while Accept headers are usually only available to programmers. Also, while content negotiation is a nice concept provided by HTTP, many developers are not used to using it

and it is common practice to include information about the requested representation in the URL.

What should be added is a clarification how the Accept headers must be processed, if these are present. As the f parameter overrides any Accept header and is a required parameter, the Accept header shall be ignored.

One aspect that could be discussed in a future revision is adding support for media types as they are registered with IANA in addition to the (often convenient) shorthand notations like "json" or "jpeg".

Resources are operations

Some comments relate to the resources of the GeoServices REST API and that they are non-RESTful operations. As described in section 6.3.2 of Part 1 (Core) a number of resources of the API are special and usually invoke some processing task on the server:

- So-called controller resources that edit information in the server, sometimes they will edit more than one resource (e.g. add, edit or delete multiple features in one HTTP POST request).
- Resources that query information on the server and create transient resources that are not persistently stored on the server and that are not made available with their own URI, but returned in the response from the controller resource. These resources could also be viewed simply as accessing existing resources on the server, while in general these will be dynamically created by the controller resource. In most cases these resources will support both GET and POST. The use of GET is usually preferable as the responses to these requests may be cached.

Both types of resources are commonly used in practice and are consistent with the HTTP standard.

Another aspect raised in the comment is the use of verbs in the names of such resources (e.g. "addAttachment") which differs from the typical current practice in RESTful APIs. In the case of the Feature Service controller resources addAttachment, updateAttachment and deleteAttachment (similar for feature editing) are used. This is a result of using POST instead of PUT or DELETE. If the HTTP methods would be the verbs then POST, PUT, DELETE could be invoked upon the attachment aggregate (POST) or the attachment itself (PUT, DELETE). However, since controllers are used that all use POST only the different operations (adding, updating, deleting) have to be distinguished and thus the verbs become part of the URI.

In other cases (e.g. submitJob or executeTask) there is no apparent reason for using verbs except to make the URI as self-descriptive as possible - again since only a limited number of HTTP methods are used in the API.

Discovery of query

These two aspects directly relate to the REST principles of self-descriptive messages and "hypermedia as the engine of application

parameters via the API (not using out-of-band information)

Absence of hypermedia controls / links within the JSON representation

state". Both aspects are indeed not supported in the current candidate standard. This discussion is currently not included in Part 1 (Core) and should be added as explanation.

In the GeoServices REST API these aspects are in most cases part of the specification document and not discoverable for clients talking to the endpoint without any prior knowledge. Yes, it would be nice, if more of this information would be available through the API and this is certainly an area where work that could and maybe should be done in that direction in the future.

However, from a standardisation perspective this seems too early at this moment. The industry is still in an early stage with much of this. There is a lot of activity around these topics in the wider IT world, but there are no widely supported standards or practices and before we can safely adopt certain technologies for our geospatial services we should ensure that they support and not hinder our goals.

For example, we have introduced JSON Schema in the OGC drafts for the GeoServices REST API in order to be able to specify the JSON representation in a more formal way. At the same time, the future of JSON Schema is unclear (the IETF draft expired more than a year ago and there is not much momentum there, the author mentioned a lack of uptake as one of the reasons for not pushing more). As long as we use JSON Schema on the specification level only and do not create a mandatory dependency in operational deployments, this should not be a problem. This would be different, if we would force clients to rely on JSON Schema operationally. If JSON Schema matures, is adopted by the wider community, supported by the relevant tools and proven to be useful on the operational level in (extensions to) our APIs, this may change.

The situation is more or less the same when we look at support for describing

- query parameters (URI templates, a new RFC, do not provide support for specifying the range of values for query parameters)
- service descriptions or
- link relations (if we look at link relations in JSON representations there are so many parallel activities, all with different scopes and not consistent with each other including HAL, Siren and support for links in JSON Schema; all are work in progress and with an unclear future at this moment).

Related to the topic of links / hypermedia controls: While there is no industry practice for including links in JSON, HTML of course supports such a mechanism. Therefore, it is natural that any HTML representation of the GeoServices REST API resources would include such hypermedia controls. The ArcGIS for Server implementation supports HTML encoding, too, and provides hyperlinks for each resource. See for example

http://sampleserver1.arcgisonline.com/ArcGIS/rest/services?f=html.

What could be done to be clearer regarding the hypermedia aspects in the current candidate standard document is to specify the links for each resource. Once well-supported (well-supported by application development frameworks) practice for including hyperlinks in JSON emerges this should be considered for inclusion into the GeoServices REST API.

1.7 The title GeoServices REST API of the series and the titles of each part

The editors acknowledge the difficulties in finding the best title for a specification and the topic was discussed before in the SWG – with the result to keep the current title for now. Going forward, we still propose to stick to the current title(s). Comment DN-3 discusses a specific case. Let's look at the different parts of the titles:

- "GeoServices": This serves as a name for the family of services to distinguish it from the OWS standards, which follow the "Web Xxx Service" or "Sensor Xxx Service" pattern. While all OGC standards are (or should be) related to "geo", the UpperCamelCase notation also identifies "GeoServices" clearly as a new name and avoids confusion with the OWS standards.
- "REST API": See the arguments in section 2.4, why the use of "REST API" is justified and in line with common practice.
- "Map Service", "Feature Service", etc: These names are appropriate for the services as they reflect the scope of the service. The GeoServices REST API Feature Service is as much a feature service as the Web Feature Service – with different capabilities, but both serve features. The family name "GeoServices" provides a clear distinction. In addition, see the discussion in section 2.2 on overlapping standard scopes in OGC.

In the SWG there was consensus about keeping "GeoServices" and the specific names of the parts like "Feature Service".

There was a discussion about removing "REST" from the title (or replacing it with another term) to avoid potential confusion as long as there is no OGC policy. There was concern that this would not be helpful as it would remove a key aspect of the API from the title and would make it less discoverable by third parties.

This candidate standard is not the only OGC standard or candidate standard that uses the term without an existing guidance. At least, WMTS already uses the term "REST" and WFS is also working on a Change Request involving the term "REST".

A motion to "Replace 'REST' with the word 'Resource' in the title of the standards" was discussed during the SWG meeting in Seoul and a poll of the attending voting members resulted in 2 vote for and 6 votes against the motion, so it was decided to keep the title unchanged.

Regarding the use of the term "catalogue" (DN-3), this is a special case. See DN-3.

1.8 Phased Approach

A phased approach as proposed in comments should not be used as it would limit the benefits from adopting the GeoServices REST API as an OGC standard discussed in section 2.2.

The SWG agreed that a phased approach would not be beneficial in the context of this submission as typically one will interact with multiple service types.

However, potential changes may include dropping support for certain capabilities from the standard. I.e., if SWG members feel that some capabilities are rather "edge cases" and the candidate would benefit from dropping certain capabilities, then this is something for consideration by the SWG.

1.9 Spatial Reference

Comment: *How does wkid relate to EPSG codes? There's clearly overlap, but can we assume that if wkid XXXX match EPSG XXXX when both codes exist ?*

For any wkid that is less than 32767, it will mathematically match the EPSG entry with the exception that the axis order of the CRS in the GeoServices REST API is always longitude/easting, then latitude/northing. Any spatial reference defined by the GeoService REST API that is not in the EPSG Geodetic Parameter Dataset, is given a wkid that is larger than 32767. Most Esri-defined spatial references have wkid values in the 100000 range.

Comment: *WKT. Foreword: it is already a shame that in existing approved standards, the valid values for projection and parameter name are not more standardized than the few samples that are mentioned in 06-103r4 (Simple Feature Access - Part 1 - Common architecture) and 01-009 (Coordinate Transformation Services). But it is well known for long that ESRI WKT diverges from other implementations in some projection or parameter names : where are those specificities defined ? For example, from my search in sr.json, ESRI WKT has only "Lambert_Conformal_Conic", whereas 01-009 lists "Lambert_Conformal_Conic_1SP" and "Lambert_Conformal_Conic_2SP" (§ 10.6.1). Actually, that difference has been known for long (http://home.gdal.org/projects/opengis/wktproblems.html).*

As the link to the GDAL web page shows, it is a fact that different WKT implementations diverge and this is something that should be addressed. ISO/TC 211 and OGC have started a new work item " Geographic information — Well known text representation of coordinate reference systems" that provides an opportunity to address the known issues. Esri is actively involved in the work item.

Also note that GeoServices REST API references the Simple Features 1.1 / ISO 19125-1 standard as the normative source for the WKT specification and this WKT specification does not list "Lambert_Conformal_Conic_1SP" and "Lambert_Conformal_Conic_2SP" (see 05-126, 6.4 and B.7). Consistent with this, the GeoServices REST API currently conflates Lambert_Conformal_Conic_1SP and Lambert_Conformal_Conic_2SP into Lambert_Conformal_Conic which supports both implementations.

Comment: *And what is the Mercator_Auxiliary_Sphere projection used for wkid = 102100 ? Not defining more rigorously WKT severely impedes interoperability*

“Mercator_Auxiliary_Sphere”, used for 102100 and 3857 is a sphere-based implementation of the Mercator projection with an extra parameter. The extra parameter is

“Auxiliary_Sphere_Type” and used to determine what to do if the projected coordinate reference system is based on an ellipsoidal geographic coordinate reference system. The default is to use the semimajor axis of the ellipsoid (“spheroid”) but you can use the semiminor axis, use a calculated authalic radius, or use a calculated authalic radius and convert geodetic latitudes to authalic latitudes before use. EPSG’s Popular Visualisation Pseudo Mercator is doing the same thing as “Mercator_Auxiliary_Sphere” with Auxiliary_Sphere_Type = 0; the semimajor axis of GeoCRS’s ellipsoid is used for the radius. The GeoServices REST API supports several projections that use an auxiliary sphere.

Again, this is a topic that should be addressed as part of the new ISO/OGC work item.

1.10 Handling cross-domain issues

As a general rule, if a web application requests files and resources from the same origin (e.g. your domain name, port number and http protocol), then access is granted automatically. If you want to allow client apps to request resources from within your domain, but outside the origin, you will run into „cross domain issues“. There are many known techniques to deal with this problem. The techniques are specific to the development platform that you are working on. Currently, Esri supports various web API environments (JavaScript, Silverlight, Flex, etc.) allowing users to build web applications that can consume the GeoServices REST API. These employ various techniques that are specific to the platforms for dealing with cross-domain issues. For example, the JavaScript environments employ JSONP and CORS as possible techniques, and the Flex and Silverlight environments use „cross domain policy files").

The GeoServices REST API specification should not limit itself to any specific solution. The current version has a conformance class for JSONP support. The SWG agreed to add another conformance class for CORS support or in the future for any other technique that is in broad use and requires explicit support on the server side should be added, too.

3 Comments

Each comment includes

- a unique identifier composed from the initials of the submitter and a sequential number,
- a reference to the relevant document or section ("General" for general comments),
- the criticality according to the levels editorial/minor/major,
- the comment or change proposal,
- a categorisation of the comment, if this touches on one or more of the key topics discussed in chapter 2 and
- the original comment of the editors as well as the disposition of the comment by the SWG

Nr.	Reference	Criticality	Comment / justification for change	Topics	Resolution
AC-1	General	Major	The documents proposed by ESRI for adoption at the OGC under the collective title "GeoServices REST API" should become well written profiles of the existing OGC Web Services. The documents propose web services whose functionality largely overlaps that of existing services (and confusingly the documents reuse the names of the existing services) but whose functionality is uniquely constrained to a simple model of geospatial features. Since this is *exactly* the kind of usage for which the current services are being rewritten and modularized, the documents should become profiles of the current services, defining services backed by simplified, tabular feature stores. The amount of work would actually be less than the amount of work required to clean up the current documents so they correctly specify what they are trying to specify. In the mean	OGC Baseline, Title	(Editors comment: See the discussion in section 2.2 and 2.5 regarding the comments on the proposal to concentrate on revising the OWS standards and the naming of the standards. Regarding the comments on the wording of the requirements see the more specific comment AC-3.) This comment cannot be implemented without

time, since ESRI retains copyright to its submission, if it feels the specifications are adequate, ESRI could publish them as their own specification document for the existing ESRI service.

The overlap with existing web services is a key source of interoperability issues. The documents do not even provide way for OGC members to distinguish issues related to the one service kind versus the other: when we say "the map service parameter" how are we to distinguish which map service we are discussing? We currently have the "OGC Map Service" but what is this new one, the 'GeoService Map Service', the 'REST Map Service', or the 'ESRI Map Service'? Only the latter one makes semantic sense since the existing OGC Map Service is clearly a Geo* Service (as are all OGC Web Services) and will soon have RESTful profiles. The proposed web services are different in that they provide a simple communication style to services backed by feature stores with a simple, flat feature attribute model. That should be front and center so that we could talk of the 'Tabular Feature Map Service' but really it is the "Map request interface to the Multipurpose Tabular Feature Service". The proposed documents must distinguish the nature of the proposed services from the existing services to prevent tiresome confusion in the next few years.

The documents need an enormous amount of work to become useful specifications for the proposed services, work that could be better spent in a harmonization effort.

breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration by the OGC membership during the adoption vote and for consideration during future major versions of the standard.

The work required to turn the proposals into useful specifications is outlined in a series of Change Requests of which this is the first. In brief, the requirements need a global review because they are generally badly phrased and often propose untestable injunctions, the conformance tests need to actually be written since the proposed tests are stopgap placeholders that state 'test this stuff to make sure it is so' rather than specific tests which clarify the requirements, and the functionality of the services needs actually to be specified since the current documents will do things like introduce a request element but never even explain what it is supposed to change, let alone develop rigorous injunctions and targeted tests for the impact of that parameter.

The proposed documents essentially need to traverse the past decade of work of the current standards. Whereas existing web service standards are being rewritten to clarify all sorts of detailed aspects of their behavior and exchanged resources, the proposed documents offer none of this clarity. Similarly, the proposed documents do not even follow standard industry practices in things as simple as how they specify file formats. They are repeating errors made a decade ago in the existing OGC services.

The OGC membership should encourage the authors to develop the proposed functionality as a series of well written Profiles of existing services, profiles which define multipurpose geospatial services backed by a simplified, tabular feature model.

AC-2	The shared title of the specifications in the series: "GeoServices REST API"	Major	The proposed title shared by all the documents proposed by ESRI for adoption by the OGC, documents 12-054 through 12-062 "GeoServices REST API" should be changed for clarity prior to publication. The proposed title is confusing, does not describe the true nature of the documents, and suggests the documents tackle a greater scope than they actually do, leading directly to interoperability issues and conflicting with the existing set of Web Services already standardized at the OGC. Grammatically, the phrase suggest the specifications specify an API (a term which is never defined) but the formal requirements in the documents enjoin the Standardization Target Type "web service." Thus the documents are actually defining the behavior of web services in response to messages sent to those services. HTTP messages can be malformed in a myriad of ways that need to be adequately addressed by any web service standard, something that code APIs can avoid since language grammar checkers, pre-processors, compilers, or runtime engines will perform this fundamental checking. Thus a 'web API,' while a cute shorthand, does not adequately reflect the nature of what is needed in a standardization document. The specifications to not specify a "REST API" since "REST" of course *not* a type of API but a characterization of the behaviour of a web service. Wikipedia calls it "a style of software architecture for	Title, OGC Baseline	(Editors comment: See the discussion in section 2.5 regarding the title and section 2.4 regarding the use of "REST" in the title and the content of the specification.) Rejected, see 2.5.

distributed systems such as the World Wide Web", that is, a style of web services. So perhaps "REST Services" would be more accurate but since "REST" is never defined in the specification, even that is problematic. Many of the core issues motivating the concept of REST based service communication architecture are not discussed: while the services act like HTTP endpoints, it is not clear even that they are required to follow the HTTP standard, while the issues of caching resources are central to the discussion of REST, the issues are ignored by the proposed specifications, and while the impact of proxies central to the work of HTTP/1.1, HTTPbis, and REST, those issues are not considered by the documents. Indeed, the best justification given in the document is:

> *{I}n general the patterns used in the GeoServices REST API can be found in the ... {"RESTful Web Services Cookbook" from Subbu Allamaraju}*

which is to say that the document claims that its use of URL templates follows the pattern suggested by a REST book. Thus perhaps the documents propose 'HTTP Request Templates.' This is a long way from the notion of identifying the resources in the web architecture and building an architecture designed around the naming of those resources, the exchange of representations of those resources, and the update of those resources. The documents only consider partially these issues so that, in the specification, the word "REST" acts as more of a buzzword than as a concept

put to useful work to solve interoperability issues.

The specifications do define "GeoServices" if those are 'services related to geo(graphy/spatial information/...) but so do all the other 'web service specifications' at the OGC so one wonders why this word is given such prominence. The name probably made sense in the context of ESRI but no longer makes sense in the OGC context.

Jointly, the three words lead to confusion, since they seem to imply that these documents present the OGC wide approach to RESTful communication with OGC Web Services. However, that work is going on as separate efforts in the context of each respective SWGs. Almost all the OGC web services are being rewritten as new versions which are

(1) more rigorously defined
(2) better written
(3) modular

with the latter explicitly aiming to provide the foundations to develop RESTful approaches.

Developing a clear title for the specification series will require getting at the essence of the nature of the document.

In actuality, these specifications describe the set of web services developed by ESRI, wrapped inside an OGC document. These web services suffer all of the problems of services developed by a single vendor without formal design, discussion during development, feedback during implementation, or harmonization

work during standardization. The web services are ad-hoc, calling themselves one thing while dealing with others, e.g. a 'Map Service' that actually handles 'feature' queries and filters. Contributions towards standardization were explicitly refused during the development of this document since standardization would require harmonization with existing services which might require 'backwards incompatible' changes with the existing ESRI services, a type of change refused outright by the SWG. (That ESRI retains copyright to the document even at the RFC stage suggests as well that this is *not* an OGC developed document but a production of ESRI.) So, really, these documents define the ESRI services and the title should reflect that, with "ESRI Web Services" but, of course, that begs the question of why this work is happening at the OGC. Absent the use of the 'ESRI' name in the title, these services need to be *clearly* distinguished from the existing services by using different names.

The specifications express the functionality of services dealing with geospatial entities with a simple conceptual model. This is made clear in the document "...Relation to the OGC Baseline" (12-062):

> *However, for many use cases a simpler set of requirements is sufficient – and the GeoServices REST API addresses this market.*

so that could become part of the ultimate title. The simplicity comes in two forms, first the conceptual model of the geospatial entities manipulated by the

service is simple, i.e. it uses 'tabular features' ('tabular' is used instead of 'simple' because 'simple features' have a problematic dual meaning: for some standards 'simple features' are features with a simple, flat list of attributes that can form a row in a table of all attributes but, in other standards, 'simple features' are features with a single spatial representation made of relatively simple vector geometries but potentially arbitrary non-spatial features). Secondly, the services offer only a limited amount of functionality. However, the latter could change over time by extension to the proposed specifications whereas the former aspect is central to the stated goal of this set of specifications. So the title should reflect this notion of 'Multipurpose Web Services for tabular features.'

The specifications also undertake a communication style which adopts some notions of the RESTful design although the purpose or advantage of what is adopted is never stated. The clients issue service requests using URL templates, the services use Javascript friendly message formats, and the service behaviour does consider idempotency and safety of requests to the server (more on that in the next change request). However, as stated above, only very limited consideration is made for the issues generally discussed under the banner of REST.

So, it seems that these specifications describe the behaviour of:

- existing ESRI implementations,
- web services,

			• multipurpose services, • services using a simplified model of geospatial features that have linear interpolated vector based spatial descriptions and tabular attributes, • services that accept requests to endpoints defined using URL templates, • services that favour JSON, and • services that have functionality defined in various extensions. Indeed, it is actually unclear why this is not a single service, say the "Multipurpose Tabular Feature Service" with many kind of interactions, say the "Map Requests for the Multipurpose Tabular Feature Service" and the "Data Requests for the Multipurpose Tabular Feature Service". Given this nature, it should be possible to develop a shared title which better reflects the nature of the documents, which does not imply more than the candidate specifications offer, and which distinguish the contribution of the proposed specifications from the currently published standards. I am sure the authors could, if they chose to do so, find a title pleasing to them and less confusing to the whole community.		
AC-3	General	Major	The Requirements in the proposed specification are poorly written. They do not conform to the rules for clear specification documents required by the standard The Specification Model - A Standard for Modular specifications which is currently required by all OGC specifications. Notably, requirements must:	Modular Spec Policy	(Editors comment: We disagree with the general statement that the requirements are poorly written and in general not ready for publication, but we agree that we will

- be injunctions against a Standardization Target Type
- be clear
- be testable

to ensure good specifications are produced by the OGC.

As a single example suffices to demonstrate the lack of linguistic clarity that pervades the documents. Consider the very first requirement in the whole series of specifications:

Req 1 If a request uses the HTTP GET method, the request SHALL be safe and idempotent.

which is to be tested by:

A.1.1 Test: core/get Inspect the documentation to identify, if requests specified in the GeoServices REST API standard that support the HTTP GET method, are all safe and idempotent as specified in HTTP (RFC 2616, section 9.1).

This is hopelessly confused, let alone grammatically incorrect.

Clause 7 of this first proposed specification "Core" states that it "specifies principles that apply to all services" so, while the clause fails to state its Standardization Target Type, one can assume it to be a "Web Service" and, indeed, it must be if the various other documents are to define *extensions* of this core.

review the wording of the requirements and tests again to check, where they need to be improved. We propose that the editors first do this review/changes before the requirements and tests are discussed in the SWG.

Req 1 is a special case that was added as a result of the discussions in the SWG to explicitly clarify that the processing of all GET requests shall be safe and idempotent as described in 9.1 of RFC 2616. As RFC 2616 states "Naturally, it is not possible to ensure that the server does not generate side-effects as a result of performing a GET request", so the test has been to inspect the documentation. It is unclear why inspecting the documentation would not be an acceptable test. In this case it might also be discussed, if the

So it is the "web services" which are to have their behaviour constrained: it is *not* the "request" that "SHALL be" but the web service. The requirement is trying to impose that 'conformant web service instances' should handle particular requests in a particular manner. So far, it seems merely an issue of poor phrasing.

However, the issue is deeper. The requirement seems to enjoin conformant web service instances to handle all HTTP GET requests in a safe and idempotent manner. The former means that the user cannot be expected to know the request has any deleterious effects, and the latter means that the repetition of the request has no consequences. The HTTP standard, in section 9.1 "Safe and Idempotent Methods" states:

9.1 Safe and Idempotent Methods

9.1.1 Safe Methods

Implementors should be aware that the software represents the user in their interactions over the Internet, and should be careful to allow the user to be aware of any actions they might take which may have an unexpected significance to themselves or others.

In particular, the convention has been established that the GET and HEAD methods SHOULD NOT have the significance of taking an action other than retrieval. These methods ought to be considered "safe". This allows user agents to represent other methods, such as POST, PUT and DELETE, in a special way, so that the user is made aware of the fact that a possibly unsafe

requirement should be changed to non-normative text simply referencing the text in HTTP.

Regarding Clause 7 : The conformance classes and their standardisation target types are specified in clause 2 and the standardisation target type of the core conformance class is "web service".

The use of "requests" is similar to the use in RFC 2616. To avoid misunderstanding we propose to change to wording to make clearer that the requirement is on the service, e.g. 'If a request uses the HTTP GET method, the service SHALL process the request without side effects (safe and idempotent).' That is, if the requirement is not dropped altogether.

action is being requested.

Naturally, it is not possible to ensure that the server does not generate side-effects as a result of performing a GET request; in fact, some dynamic resources consider that a feature. The important distinction here is that the user did not request the side-effects, so therefore cannot be held accountable for them.

9.1.2 Idempotent Methods

Methods can also have the property of "idempotence" in that (aside from error or expiration issues) the side-effects of N > 0 identical requests is the same as for a single request. The methods GET, HEAD, PUT and DELETE share this property. Also, the methods OPTIONS and TRACE SHOULD NOT have side effects, and so are inherently idempotent.

However, it is possible that a sequence of several requests is non-idempotent, even if all of the methods executed in that sequence are idempotent. (A sequence is idempotent if a single execution of the entire sequence always yields a result that is not changed by a reexecution of all, or part, of that sequence.) For example, a sequence is non-idempotent if its result depends on a value that is later modified in the same sequence.

A sequence that never has side effects is idempotent, by definition (provided that no concurrent operations are being executed on the same set of resources).

These are *really* nice characteristics for services to

However, we believe that it is unnecessary to start all requirements with 'Conformant web services SHALL ...' as the standardisation target is clearly defined for all conformance classes.)

It was rejected to drop the safety and idempotency requirement and test.

Note that there is no requirement that all tests are automatable. In the particular case it would also be the responsibility of W3C as the HTTP-owners to provide such a test – the standard is simply normatively referencing the HTTP standard.

The wording of the requirements and tests has been reviewed again and have been improved where deficiencies were found.

have in response to GET requests. But note the "Naturally, it is not possible to ensure..." and "However, it is possible ..." phrasing in the respective sections which show us that these requirements are NOT TESTABLE.

If we look at the proposed test which starts "Inspect the document" we see right away that the test is not testing an instance of the Standardization Target Type! We should be trying to test a "web service instance!" The test gives no indication of how one could know the condition has been met.

Even worse, this is not a question of language, since if we actually try to come up with a test, we find that the test is not fixable. At best, we could require that, for every kind of GET request supported by the instance, the testing procedure select several specific requests and reissue each multiple times. That could, at least, potentially trigger a condition of non-safety or non-idempotence, if the implementation suffered from it. However, to verify that the requests had been handled safely or with idempotence, we would have to check that nothing of functional importance had been affected by these requests. Formally, we would at least have to repeat the entire suite of all tests before and after such repeated GET requests and compare the results to see if any of the answers had changed. But since the other requests might not be idempotent, that approach does not work either and we are stuck.

That particular injunction is untestable and therefore

			must be a recommendation not a requirement. A similar analysis on the other requirements reveals a myriad of issues with the proposed specifications. The requirements need to be reviewed one by one, assessed for language, for the enjoined target, for the meaning of the injunction, and to establish how an implementor could know that the injunctions have or have not been met. The authors would be well served to adopt the pedantic but easy form of phrasing all requirements in the form: 'Conformant <target type> (implementations) SHALL (qualifying clause) injunction' for example Conformant tabular feature service implementations SHALL handle all messages sent to the <host> and <port> defined in every Service Endpoint URI defined by the service in conformance with the requirements for an 'origin server' in the HTTP/1.1 standard. which provides a clear approach to each injunction. Then, the tests of each requirement must be reviewed, including writing a series of targeted specific tests which clarify the requirement and give implementors one source of validation. These documents are not ready for publication by the OGC.		
AC-4	Part 3: Map Service (12-	Major	The name 'Map Service' is currently in use at the OGC and reuse of this name by the proposed specification is	Title	(Editors comment: See the discussion in section

	056r1) – title page		liable to cause enormous confusion within the OGC community and in the general public. The proposal should change the title of this standard. The issues in the shared title of the series has been addressed elsewhere, leading to the conclusion that the series is defining a single service type backed by simplified features with tabular attributes, that is a 'Tabular Feature Service,' and that the series is defining multiple requests types for that service. That is the service is 'multipurpose' and uses URL templates for the HTTP requests. When applied to the specific issues of map requests, we get a title like: URL template map requests for the Multipurpose Tabular Feature Service. which is more accurate and mostly avoids the confusion of calling the service a Map Service.		2.5.) Rejected, see 2.5.
AC-5	Part 3: Map Service (12-056r1)	Blocker	The proposed specification needs an enormous amount of work in order to become a clear definition of a service furnishing cartographic images over the Internet for which multiple, interoperable instances could be developed. As written, the specification is exceedingly hard to understand, the tests unclear, and the actually specified functionality unknown. The result of adopting the standard as currently written will be a mess, with different implementations each working in their own way leaving clients to pick up the pieces and learn to address the quirks of each deployed	Modular Spec Policy	(Editors comment: We disagree with this comment and do not understand where the perceived problem is. But see also AC-3 on the general proposal that the editors first review the requirements and tests before they are discussed in more detail in the

endpoint separately.

The OGC membership should reject this standard until the hard work has been done of actually defining the service behavior, of codifying that behavior into clear requirements, and of further developing those requirements into clear, targeted testing procedures.

As the document currently stands, it is unsuitable for publication.

The proposed specification does not develop clear requirements matched to direct, logical tests, which is one of the requirements of "The Specification Model" standard designed to improve the specifications at the OGC.

As an example, start with the first requirement:

> Req 1 The Map Service Root resource SHALL accept requests that conform to the URI template in Table and use any HTTP method identified in the same table.

which requires some conceptual gymnastics to figure out what it might be saying (ignoring the broken link). The "Map Service Root" is not a standardization target type, indeed conceiving of resources being things that receive requests requires some gratuitous abstraction for seemingly no gain. The actual requirement would benefit from being reformulated into some injunction on an instance of a 'web service.' I take the requirement to mean that the 'web service' is expected to accept a request sent in an HTTP message with a

SWG.

The wording of Req 1 seems fine as in the service you interact with resources via HTTP requests.

A test that validates responses against a schema is quite common.

We had discussed in the SWG whether we should add more requirements related to the semantic correctness of the results, but decided against this as this seemed to go beyond the level of requirements that is usually seen in OGC standards. Resource representations in OGC standards are usually validated against a schema. It can be discussed whether we want to go beyond this for the GeoServices REST API.

Regarding the dpi

Request-Target which starts with the path of the 'web service root URI' and, presumably, is sent over a TCP/IP connection made to the host:port combination of the 'map service root URI'. Of course, according to the rules of TCP, the web service has to accept all requests, so presumably this requirement actually is trying to say something different. I might take the requirement to mean that the web service instance is expected to respond with an HTTP message with a code in the 200 series like "200 OK" and the requested content, i.e. furnish a functional response. Probably, the requirement should say that the web service shall respond to a request in the given pattern with a specifically formulated response. As it stands, as an implementor, I would simply ignore this requirement as ill founded, confused, or ridiculous; it is not worth my time to find out which.

One purpose of having a test suite is to disambiguate poorly written requirements of this kind. The test should tell me exactly how the requirement will be assessed which should help explain the requirement. However, where is the test? I might guess that the test of the first requirement would be the first test in the test suite. This test gives as its purpose: "Verify that the Map Service Root resource supports the request and response requirements." which pretty quickly lets us know that we have some kind of vague, hand waving test which is not going to help us figure out the requirement. I am expecting a test that is trying to cause a web service to 'refuse to accept a request'. However, the test seems to be testing the response and it looks like I could simply send back the same error

parameter, an updated description could be:

"Description: The device resolution of the exported image (dots per inch). If the dpi is not specified, an image with a default DPI of 96 will be exported.

The dpi parameter in the request is the way for a client application to request the server to produce an image suited for the specific resolution of the requesting client device.")

See AC-3 and the editors comments above. In general the requirements and tests seem to be appropriate. As stated in AC-3 the wording of the requirements and tests has been reviewed and improved where deficiencies were found.

The documentation of the dpi parameter has been updated.

message the whole time and pass the test.

> "Inspect the responses and validate them against the JSON Schema http://schemas.opengis.net/gsr-ms/1.0/root.json or for exceptions against http://schemas.opengis.net/gsr/1.0/exception.json."

In other words, the authors of the proposed document have just constructed a generic, catch all "test the service" test that does not help us implement the requirements of the document. This is a waste of implementor's time, of the CITE testing folks' time, and of the document author's time.

Beyond the poorly written requirements and tests, the specification does not undertake the most minimal explanatory effort to explain what behaviour is expected of the service. Take the 'dpi' parameter of the map request which presumably has some impact on the response. The ENTIRE discussion of this parameter comes in the one sentence definition:

> "The device resolution of the exported image (dots per inch)."

which is, of course, non-sensical since images do not have devices. The document completely lacks any explanation of what this parameter is supposed to change in the response and the document does not clarify that with a test that might explain the requirement more clearly. Again it seems the web service must only accept the parameter in the request,

			it may then simply ignore it completely. In other words, the document has not undertaken the most minimal effort of explaining the functionality of one of the elements proposed for the proposed web service. Presumably, the web service 'should work like ESRI's implementation' but that is not interoperability. This analysis could go on for a while. The authors have done heroic work writing this extended series of documents. Unfortunately, because the specification is merely providing window dressing for the one reference implementation, the authors did not do the hard work of defining the behaviour, developing well formulated requirements to enforce that behaviour, and conceiving of effective tests to ensure that behaviour is respected. The result is a document which leaves all the work of standardization to the CITE test developers and the implementors, a recipe for non-interoperable results. It is not worth this submitter's time to comment more extensively on this specification because the authors have made clear they do not intend to fix any of the substantive issues encountered. However, a simple look at the form of the document reveals that it is unsuited for publication absent some major, extensive editorial work.		
AC-6	Part 3: Map Service (12-056r1) - All the parameters	Major	The proposed standard includes parameters which are inexpressive, confusing, or otherwise problematic, hindering interoperability. For interoperability across users, implementers,	OGC Baseline, REST	(Editors comment: The reasons for the f parameter are explained in 2.4. It is unclear why

of the Map Service, sections 7 to 23	linguistic groups, and communities of interest, communication elements should express effectively their meaning. Communication elements should also reuse the language from the major, existing standards where possible, and only introduce new elements where strictly necessary, something which has been OGC practice up to now. Communication elements should be distinct from elements which already exist when expressing a different concept so as to prevent confusion. The proposed service violates these notions. For example, the proposed Map Service suggests the use of the parameter 'f' for communication. That parameter has strictly no meaning to anyone on its own, which is unfortunate because the concept could be relatively straight forwards and exists in most existing web services. The concept has an existing expression in the HTTP message grammar itself, as one of the message headers. (The proposed specification document fails to address conflicts in those two modes of communicating the same idea.) The concept also has existing expression in current OGC web services. Since the parameter 'f' communicates nothing in of itself and because more expressive alternatives exist, the use of 'f' is silly. Much worse, the use of bespoke values for this parameter, rather than the flexible, extensible, industry standard, MIME type notation, is downright foolish. For another example, the proposed Map Service suggests the use of the parameter 'format' for communication to indicate the format of the image returned. However, in existing OGC Web Services,	the comment considers "f" and inappropriate name for the parameter. The parameter name is basically a convention and using "f" seems as good a choice as other options. The comment is correct that the specification does not discuss how to treat Accept headers, if specified and this should be clarified, see 2.4. To utilize CORS (cross-origin resource sharing) it is recommended that client application include the HTTP origin header, and the server utilize this information to accommodate cross-domain considerations. The ArcGIS for Server (a reference implementation for the GeoServices REST API specification) supports CORS. More information on cross-domain issues are

that parameter means the format of the response (which the proposal calls 'f') and can be used for other types of request like the request for the capabilities document. Since the proposed document's usage conflicts with all existing OGC Services, this causes an interoperability issue which will last as long as the services need to co-exist.

These two examples just begin the process of formal analysis of the proposal. The issues could be resolved by changing the parameter names or even by prefixing them all with a token specific to the services, say "EWS_format" for the 'format' parameter used by the 'ESRI Web Services.' However, I will not take the time to develop this analysis further since it is clear that the authoring SWG members have no interest in improving their proposal beyond the currently defined ESRI implementation. If the authoring SWG wishes help in the future, the Web Map Service SWG has many members who would welcome an open, collaborative exchange on how to build the best standards for users, the industry, and our needs in the coming decades.

One of the proposed documents (12-062) states:

> ... the GeoServices REST API services should be seen as complementary to the existing OGC Web Services. Over time, harmonization between the specifications should be considered with regards to the necessary migration for existing implementations.

discussed in 2.8.

The additional "format" parameter for image formats is necessary as responses may be JSON that provide a URL of the image in the requested format. Hence the need for two parameters related to formats.)

For most parts, this comment cannot be implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration by the OGC membership during the adoption vote and for consideration during future major versions of the standard.

An additional requirement has been added to clarify the requirements regarding Accept headers.

			One wonders why the OGC should wait until *after* initial publication to begin the standardization and harmonization work.		
PB-1	Part 6: Image Service (12-059r1)	Major	The documents proposed by ESRI for adoption at the OGC under the collective title "GeoServices REST API" is not in sync with several relevant OGC standards: • OWS Common • GML 3.x • GML 3.2.1 Application Schema - Coverages (GMLCOV) • Web Coverage Service (WCS) • WCS Application Profile - Earth Observation (EO-WCS) The document introduces concepts, data structures, and services which are completely unsynced with the corresponding adopted standards. Re data, raster images are handled by this document - this is a coverage, and for coverages there are established data structures. However, the document does not at all rely on those, although terminology in many places heavily overlaps with OGC's coverage definitions. Additionally, the document lacks clear definitions of the terms used. Further, there is an unfortunate mix and binding of services (here: catalogue and imagery) which hampers modularity.	OGC Baseline	(Editors comment: See the discussion in section 2.2 regarding the comments on the proposal to concentrate on aligning with the OWS standards. The specific comments need to be addressed and the terminology / requirements need to be clarified.) For most parts, this comment cannot be implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration by the OGC membership during the adoption vote and for consideration during future major versions of the standard.

Finally, concepts appear rather low-level; for example, bands are numbered only, and pixel type does not contain rich enough information - e.g., GMLCOV foresees the physical measure, like radiance in "W/cm2", while the new document just knows signed/unsigned int and float. What about pixels with different types, such as rainfall (int) and wind speed x/y (float)?

Re service, the Web Coverage Service (WCS) core and extensions, together with EO-WCS, establishes a set of functionality clearly transcending what is described in this RESTful interface. Rather, the interface described in the proposed document presents a monolithic block of functionality.

Some randomly picked extra functionality is packed in, such as colormap (what about WMS?) and NDVI (why exactly vegetation and no other differential index?). Functions are not described, but just named, such as "Slope". The WCS suite, for such tasks, offers a generic language for specifying any kind of such algorithms, with an unambiguous semantics.

It is suggested, therefore, to re-establish this document as an extension to the WCS where a RESTful interface fits in well and is compatible in terms of data, functionality, and protocol standards. The same holds for JSON as a coverage exchange format: it can be embedded smoothly into the GMLCOV / WCS orchestration of different image formats; note that GMLCOV allows to have mixed representations (XML + a binary format), so this is supported likewise

Regarding definitions: The terms have been reviewed and consolidated. In particular, "raster" has been replaced by "image" where appropriate. Definitions have been provided for key terms.

Regarding functions: The mosaicRule and renderingRule parameters of the Export Image resource have been moved to separate conformance classes to remove the need for all implementations to implement such a capability. Extensive documentation of both parameters and the supported mosaic rules and raster functions has been added.

			and can be extended, e.g., with JSON. If adopted as is, OGC will have a new miniworld inside OGC which is isolated from and incompatible with OGC standards at large.		
PC-1	General	Major	The GeoServices REST API spans over several services: Core, Catalogue, Map, Feature, Geometry, Image, GeoProcessing and Geocoding services with the possibility of more services to come in the future. This is a radical new direction in terms of architectural approach for the OGC. This specification promises a standard way for web clients to communicate with geospatial services based on Representational State Transfer (REST). This candidate standard claims that the API is intended to make implementing servers instantly usable by developers. It also claims to allow end users to discover, access and use services in workflows. These are pretty substantial claims. For discovery, this REST API uses the catalogue service. This is actually misleading. Far from being a service, it is actually a custom discovery document that is supposed to help end-users or machines discover, access and use services. The end point of that document is not auto-discoverable (problem). It is at a fixed location that varies based on the service type. It is accessed using a non standard query parameter "?f=json". This is very unique to this API. Better choices would have been fmt, format or alt. This API does not seem to support Accept headers that machines are more likely to use such as Accept=application/json and custom types that would imply that the client machine agrees with the proposed API via content	REST, OGC Baseline	(Editors comment: Most issues raised in the comment are discussed in section 2.4 and 2.2. The comment about the synchronous and asynchronous processing tasks is unclear. If the service authors decide that some processing tasks are near instantaneous, the will likely use the synchronous variant. In this case using GET for information retrieval seems adequate. It is an implementation detail whether a process is executed when the request is received or whether the resource already exists on the server and is returned. This is not RPC.

negotiation. If multiple services are implemented on a specific server, the root document (if you find it) is supposed to provide you links to the respective catalogue documents. It does not. You have to build those links based on out-of-band information (problem). The catalogue document does not contain schema links or information (problem). This is a problem if the version changes. You may not have access to the new schemas. The catalogue document does not contain a standard list of resource collections (problem). The contents of a GeoProcessing service is very different from the content of a Map service. You need out-of-band information to parse those documents in a different manner, making it hard to be instantly useable. From the specification, the GeoProcessing service seems to have some concepts of resources (if you know in advance what they are). It has tasks resources that seem to morph into job resources based on some operations that are also called resources: ExecuteTask and SubmitJob. The confusion between resources and operations is pervasive in that document. We are back to a REST/RPC approach that is non standard. This approach will make it very difficult to harmonize with a RESTful SPS, WCPS or Workflow Chaining Service. The task resource is not really a task that can be submitted by a user but more like a process definition. There is an ExecuteTask 'supposedly' resource but really an operation that you can trigger with a GET (gasp) but you never create nor can retrieve after the fact. This is supposed to simulate a synchronous operation. There is a claim of a SubmitJob resource for asynchronous operation. You can do a POST but that post does not create a

For asynchronous processes, job resources are created using POST and the client can then poll the resource about the processing status.

E-tags, or entity tags, are a mechanism that Web servers and browsers use to determine whether a component in the browser's cache matches one on the original server. In this case, a component is the same as an entity; this refers to things from a Web application such as images, scripts, and style sheets. E-tags are used for validating entities and are considered the most flexible way of performing last-modified updates rather than component comparisons. The use of E-tags is therefore recommended. The ArcGIS for Server (a reference implementation for the GeoServices REST API specification)

SubmitJob resource (as one would expect) but mints a new type of Job resource. Its parameters can be found in table 9, which may make it hard for workflows that can't read that spec. There is no standard approach trying to identify generated products by mime-types, links to data files and/or use atom feeds. There is no standard approach to use any type of notification system. This will simply not scale up in an enterprise system. This GeoProcessing specification needs a complete redesigned approach. I have not found any architectural details concerning information caching, use of ETags… and other header parameters. This is an additional major issue. In general, how RESTful is this GeoServices specification? Leonard Richardson has defined a Maturity Model which is best explained here: http://martinfowler.com/articles/richardsonMaturityModel.html. It does pass level 0 using HTTP as the transport system. It fails level 1 as it mixes the concept of resources and operations. What you post is not what you get. It uses limited HTTP verbs (GET and POST), but no PUT/DELETE/HEAD/OPTIONS, no content negotiation, It fails miserably with level 3. There is no hypermedia controls, no concept of Hypertext As the Engine of Application State (HATEOAS). The discovery mechanism is extremely weak and misleading. It requires a considerable out-of-band knowledge. Resources do not describe their own capabilities, schemas, interconnections, next actions. It ought to be a requirement for upcoming "RESTful" APIs to support existing tools such as i/o docs for interactive APIs based solely on discovery documents (http://www.mashery.com/product/io-docs)

supports E-tags. It should be discussed, if requirements or recommendations related to E-tags should be added to the specification in a separate conformance class.)

For most parts, this comment cannot be implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration by the OGC membership during the adoption vote and for consideration during future major versions of the standard.

An additional conformance class has been added for E-Tag support.

			This specification is in the open so everyone is free to follow it or not. Is this a good thing for the OGC to accept? What would be the value added for the OGC Community? I think that this would be a huge problem if this specification were to be accepted. It is the wrong approach from an architecture standpoint. It will force the creation of divergent standards that will be more "RESTful" and will use JSON + Geo extensions + hypermedia extensions. This will create some major confusion in the market place. This will also be extremely costly for the developer community. How many standards are we going to have to support? This specification is far from any standard and does not cover all the needs of this community. Extensions such as SPS, WCPS, WfCS, WNS … are required that will complicate the issues even more. This specification does not even address security, notifications / publications / subscriptions, data feeds. From individual service standpoint, the GeoService components ought to be resubmitted individually to their respective SWGs (WMS, WFS, WPS…) The data models ought to be realigned with existing OGC models that have been developed over many years or at minimum, rediscussed in those specialized forums.		
ER-1	General	Major	I totally second all the comments made by Adrian Custer, Peter Baumann, Cameron Shorter and Pat Cappelaere on the GeoServices REST API proposal (see http://lists.opengeospatial.org/pipermail/requests/2012-July/date.html).	OGC Baseline, Spatial Reference	(Editors comment: The issues raised in the first part of the comment are discussed in section 2.2 The issues regarding WKT/CRS are explained

What is fundamentally shocking in the submission is the complete overlap, and sometimes contradictions, with many other existing OGC standards, and the lack of effort or willingness to resolve that overlap. This is acknowledged in

12-062r1 (GeoServices REST API – relationship with the OGC baseline), and anyone reading that document will wonder why OGC has even considered standardizing GeoServices REST API.

Quoting 12-062r1, "While it would be possible to develop new versions of the OGC Web Services standards using a consistent framework and with support for JSON representations and a RESTful "binding", this will likely take significant time due to the unresolved REST-related discussion items, the current organization of OGC SWGs based on the individual standards and the fragmentation into separate standards. " --> if this statement is true, how can OGC publish that without questioning how it works !!! So this proposal is in fact a way to "shortcut" the standardization procedures used by OGC. It is expected that standardization takes some time, otherwise you have a high risk of ending up with half-backed standards.

Other quote from that document : "Over time, harmonization between the specifications should be considered with regards to the necessary migration for existing implementations." Certainly, but if OGC adopts the proposal, it would have to provide migration plan for the existing standards AND the

in 2.7.)

This comment cannot be implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration by the OGC membership during the adoption vote and for consideration during future major versions of the standard.

Regarding WKT: see 2.7.

GeoServices. So the pain would be even greater.

Letting aside those general comments to concentrate on the specific issue of spatial reference (§9.2 of 12-054r1). Hardly anything is defined :

- How does wkid relate to EPSG codes ? There's clearly overlap, but can we assume that if wkid XXXX match EPSG XXXX when both codes exist ?
- WKT. Foreword: it is already a shame that in existing approved standards, the valid values for projection and parameter name are not more standardized than the few samples that are mentioned in 06-103r4 (Simple Feature Access - Part 1 - Common architecture) and 01-009 (Coordinate Transformation Services). But it is well known for long that ESRI WKT diverges from other implementations in some projection or parameter names : where are those specificities defined ? For example, from my search in sr.json, ESRI WKT has only "Lambert_Conformal_Conic", whereas 01-009 lists "Lambert_Conformal_Conic_1SP" and "Lambert_Conformal_Conic_2SP" (§ 10.6.1). Actually, that difference has been known for long (http://home.gdal.org/projects/opengis/wktproblems.html).

And what is the Mercator_Auxiliary_Sphere projection used for wkid = 102100 ? Not defining more rigorously WKT severely impedes interoperability.

			Finally, OGC must make a clear statement : if the GeoServices REST API represents the way to go, then OGC must clearly deprecate all other existing overlapping standards (WMS, WFS, WPS, WMTS, etc etc....). It would cause a lot of confusion if that GeoServices REST API proposal would coexist with existing adopted standards. ESRI is certainly free to publish and document the "API" to its services and promote it, but I believe OGC should not just act as a rubber stamp and should pay close attention of having a consistent body of standards. It would certainly loose a lot of credibility if it blesses as a standard a vendor specific protocol that clearly conflicts with other standards it has already developed and adopted (this reminds me of the controversy about ISO finally adopting OOXML after having also adopted ODF).		
HT-1	General	Major	Introducing this comprehensive API, in its current form, as an OGC standard would be very confusing to users, as the proposed API deviates from existing OGC standards in several ways (as documented in "GeoServices REST API - relationship with the OGC baseline (12-062r1)"). If REST APIs are to be standardised by the OGC, they will have to comply with current OGC standards.	OGC Baseline	(Editors comment: See section 2.2 for a discussion of the issue raised in this comment.) This comment cannot be implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration by the OGC membership

					during the adoption vote and for consideration during future major versions of the standard.
HT-2	Part 1: Core (12-054r1), section 9	Major	Even if GeoJSON is not an OGC standard, it seems unnecessary to introduce a "competing" JSON encoding. If GeoJSON does not cover the geometry types that are needed, extending GeoJSON would be preferable to introducing a new JSON encoding.	GeoJSON	(Editors comment: See section 2.3 for a discussion of the issue raised in this comment.) See 2.3.
VM-1	Part 1: Core (12-054r1), section 9	Major	The current draft uses its own way to encode geometries as JSON. Though there is an already existing specification (which is not an OGC standard) which is called GeoJSON [1]. It already provides an encoding for the geometry types (Multi-)Point, (Multi-)LineString, (Multi-)Polygon and Geometry Collections. It is already supported by a huge number of projects that implement OGC standards. To name a few: PostGIS, GeoServer, OpenLayers. Hence I propose that GeoServices REST API uses the existing widely used GeoJSON encoding, rather than inventing its own one. [1] http://geojson.org/	GeoJSON	(Editors comment: See section 2.3 for a discussion of the issue raised in this comment.) See 2.3.
CS-1	n/a (comment did not use the comment template)		As an OGC member (from LISAsoft), I would like to second Volker's comment's below regarding GeoJSON.	GeoJSON, Reference Impl.	(Editors comment: Implementations of the GeoServices REST API from multiple

			I would also like to ask: • Has the REST API standard been tested amongst multiple client and server applications, as is usually done in OGC Open Web Services Testbeds? • Is there an Open Source reference implementation of the REST API standard, as has been the case for most (all?) OGC standards to date. I'm nervous that the proposed REST API standard may be supported by one vendor, but has not been given the attention required to be adopted broadly by other applications.		organisations exist, see 2.2 for some examples. ArcGIS for Server can be considered as a reference implementation and Esri will offer to provide such a reference implementation to OGC. Note that the proposed revision of the CITE P&P – currently under vote in the OGC TC – no longer require that reference implementations are open source.) No change proposed.
DN-1	Part 4: Feature Service, 9.3.2, req. 20	Minor	Time shall be expressed in ISO 8601 time for single and start/end times.	Specific Changes	(Editors comment: The issue with this proposal is that it would break backwards compatibility. Also, the "unix time representation" used in the GeoServices REST API seems to be commonly used in JSON - and is directly supported by JSON Schema.)

					This comment cannot be implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration for later major versions of the standard.
DN-2	Part 4: Feature Service, 12, req. 54-73	Major	For symmetry with the identified json structure "attributes" and conformity with the ISO feature model, the Clause reference to "Attachment(s)" and "Attachement(s)" (misspelling) shall refer to "Attribute", "Attributes", or "Attributes Info" as appropriate. These are not just attachments - they are the visible properties of the feature. Attributes Info should expose which (if not all) attributes are available for display. It is not described in this document how "geometry" might be returned; if desired, geometry may be returned in the context of Attributes. *Updated document with proposed changes submitted with comment*	Specific Changes	(Editors comment: The attachments are attachments, i.e. resources (files) attached to a feature. Attributes and geometry are part of the feature resource, see Feature Service section 7.4. Is some text needed to clarify this?) A clarification about the distinction between attributes and attachments has been added to the document as well as to the document that describes the relationship with ISO 19109.
DN-3	Part 2: Catalog	Major	The Part name of "Catalog" is inappropriate relative to	Title	(Editors comment: This

Service	the current OGC definition of catalog and the existing catalog services standard. Although this may provide access to a 'catalog' of services, it is actually reporting service information or capabilities. I have changed all use of "catalog" to "services" to help avoid misunderstanding within the OGC community. This shall also be reflected in the document names and companion schema. *Updated document with proposed changes submitted with comment*	should be discussed. In general, the use of "catalog" seems to be consistent with the definition of the term in the abstract specification, but in practice the term catalog is used in OGC differently.) There is agreement that the use of "catalog" in OGC including in topic 13 in general implies a capability to both query the resource descriptions (not supported by part 2) and to store entries about distributed resources (part 2 supports only services hosted on this server). It was agreed that a better name would be "Service Directory". This is now stated explicitly in the document and its definitions to avoid confusion. A name change has been anticipated in version 2 in the future work

					section.
DN-4	Part 1: Core	Major	Edited for consistency with changes proposed in DN-2/DN-3. *Updated document with proposed changes submitted with comment*	see DN-2 and DN-3	(Editors comment: Depends on resolution of DN-2/DN-3.) Minor changes to be consistent with the edits from DN-2/3 have been made to the document.
AM-1	General	Major	Please remove the word "REST" from the title as it seems to be misleading and irrelevant for this submission. The use of the word "REST" in the title of this submission should be postponed until a firm and consensus definition exists within the OGC. But to the best of my knowledge, this is not the case. If not removed, this submission should clearly state what the submitting organizations' definition of "REST" is and provide a disclaimer that this does not necessarily represent the understanding of the OGC membership: The document 12-054r1 describes in section 6 "Fundamentals of the GeoServices REST API" and in particular in section 6.2 "REST and pragmatic considerations" the approach taken. I find the argumentation incomplete and not substantial enough as an argument for having the word "REST" in the title. I furthermore see it problematic to use a non standard publication as the "helper" for describing "REST" in an OGC implementation standard: "RESTful Web Services Cookbook from Subbu	REST, Title	(Editors comment: See sections 2.4 and 2.5 for a discussion of the issues raised in this comment. We do not think that it should be the role of the standard to define REST. Clause 6 is used to explain the design decisions, not to provide a definition of REST. The reference to the cookbook from Allamaraju had been added to provide some further information and background, also since this is one of the two books identified by the OAB as providing guidance on this topic. It

			Allamaraju". Following the "resolvable" reference from the Bibliography section, you find in the abstract a statement about REST: "While the REST design philosophy has captured the imagination of web and enterprise developers alike, using this approach to develop real web services is no picnic.". As this is maze fully true, I wonder if a more firm statement can be found in this publication. Also, I have heard other publications in this context that are not referenced. Because of a missing definition, I ask the submitters to rewrite section 6.2 and provide a clear definition of "REST" relevant within the context of this submission.		is not essential for the specification and we propose to remove the references.) Rejected, see 2.5. Regarding the cookbook references: all references have been removed.
EK-1	General	Major	The name of the specification is inappropriate in today's evolving state of REST technology and should be changed to exclude REST from its title. The overuse of GET and POST in the specification is considered by many as common mistakes in REST. The following comments illustrate that point: • The deletion of resources is not accomplished via the HTTP DELETE operation. Instead, they are accomplished via an overloaded POST operation that is contrary to the principals of REST. • Many of the URLs refer to operations rather than resources. E.g., the URL of a map is "<...>/export?<parameters>". The URL therefore specifies the operation and not the resources. • There are a few resources (such as maps) that can be requested via either HTTP GET or HTTP POST. The usage of HTTP POST as a	REST, Title	(Editors comment: See section 2.4 for a discussion of the issue raised in this comment.) This comment cannot be implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration by the OGC membership during the adoption vote and for consideration during future major versions of the standard. See also 2.4.

			glorified GET operation is contrary to the principals of REST. In addition, the initial constraints imposed by the selection of the software tools that lead to the GeoServices API specification have for most parts disappeared (see section 6.2.2 of GeoServices REST API - Part 1:Core). The specification as currently written is at best a work-around, already outdated, and does not respond to current requirements for using REST.		Regarding the title: see 2.5.
TS-1	Part 4: Feature Service (12-057r1), 7.4.3, req. 11	Major	Use GeoJSON (http://geojson.org/geojson-spec.html). GeoJSON is an openly-developed, widely-used specification for simple features and geometries. It would facilitate the adoption of the proposed standard to use an existing encoding. The GeoJSON Feature object maps well to the proposed http://schemas.opengis.net/gsr/1.0/feature.json.	GeoJSON	(Editors comment: See section 2.3 for a discussion of the issue raised in this comment.) See 2.3.
TS-2	Part 5: Geometry Service (12-058r1), 7.3, req. 3	Major	Use GeoJSON (http://geojson.org/geojson-spec.html). GeoJSON is an openly-developed, widely-used specification for simple features and geometries. It would facilitate the adoption of the proposed standard to use an existing encoding. The GeoJSON Geometry object maps well to the proposed http://schemas.opengis.net/gsr/1.0/geometry.json. The one exception of GeometryEnvelope, and that can be encoded without GeoJSON (or proposed as an addition to the spec).	GeoJSON	(Editors comment: See section 2.3 for a discussion of the issue raised in this comment.) See 2.3.

EK-2	General	Major	Introducing the GeoServices REST API is incompatible and contrary to OGC's mission "to advance the development of international standards for geospatial interoperability". The OGC's mission and mandate need to be reviewed and changed in light of this specification prior to its adoption by OGC members. The issue that few OGC members are attempting to resolve by introducing this specification to other OGC members and to the geospatial community is that the existing OGC service specifications are not RESTful. However, the GeoServices REST API introduces a redundant specification that is not directly compatible with the existing suite of OGC specifications developed during the last 15 years. This is resulting in a lot of confusion for all OGC members, higher financial risks for all organizations already compatible with current OGC specifications and interoperability problems that are worse than not having a RESTFul API. If adopted, this specification would drastically hinder interoperability, since organizations would likely implement support for only one or the other of the possible APIs into client and server applications. A client or server application that supports both possible APIs would take much more development effort, increasing development costs and seriously impacting adoption of OGC specifications in the market. If adopted, this specification will generate a large amount of confusion, higher financial risks, and a large number of interoperability problems in the OGC community for years to come.	OGC Baseline	(Editors comment: See section 2.2 for a discussion of the issue raised in this comment.) This decision should be left to the OGC membership.

AT-1	General	Major	The GeoServices REST API Candidate is a very large specification consisting of 8 parts that encompass the entirety of possible geospatial services. It is built upon a core that is then extended for subsequent services. Due to the encompassing nature of the specification along with the concerns of the baseline as well as individual components this suggestion is to adopt a phased approach to the proposal, feedback, development and acceptance of these parts. By adopting a phased approach discussion can be focused on developing a common, agreeable Core. Following that there could be grouped development and acceptance of comparable services. For example then develop the Geometry and Feature service, and in a third phase the Catalog and Map service, and finally the Geoprocessing and Geocoding services. In addition to more focused discussion this phased approach would allow developers to build reference implementations and verify concepts along the way. Within this development issues could be resolved before moving onto the next set of services. Through a stepped roadmap the entirety of the specification could be adopted but in a phased approach to achieve the best possible outcome with focused discussion and solid implementations to prove the concepts.	Phased Approach	(Editors comment: See section 2.6 for a discussion of the issue raised in this comment.) Rejected: see 2.6
JH-1	General	Major	Phased Approach - Hybrid Architecture As noted in other comments, the GeoServices REST	Phased Approach	(Editors comment: See section 2.6 for a discussion of the issue

API Candidate is a very extensive specification consisting of multiple component APIs encompassing a variety of geospatial services, many of which overlap with current OGC Services. It is built on a Core that is extended for associated GeoServices.

Due to the broad nature of the draft specification and the concerns of the OGC baseline, as well as alignment of individual API components, a phased approach to the proposal, feedback, harmonization, development and acceptance of these parts has been suggested. This suggestion recognizes comments on 'Phased Approach' by Andrew Turner, Esri, and extends them below by proposing integration of interoperability points to enable the use of existing OGC Services in a hybrid architecture.

For example, "By adopting a phased approach discussion can be focused on developing" a common, consensus-based Core. Following this there could be component API development and acceptance of associated services, which include interoperability points enabling use of both REST and SOA-based mechanisms depending on market or enterprise needs. As noted, "after the Core, develop the Geometry and Feature service, and in a third phase the Catalog and Map service, and finally the Geoprocessing and Geocoding services." Within the Core, for example, there could be a REST 'Landing Page' that allows discovery and invocation of REST services when the need is for optimized web performance, and the ability invoke SOA operations when more complex geospatial

raised in this comment.)

This comment cannot be implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration by the OGC membership during the adoption vote and for consideration during future major versions of the standard.

			needs are encountered by applications or users. In addition to more focused discussion a phased/hybrid approach "would allow developers to build reference implementations and verify concepts along the way. Within this development issues could be resolved before moving onto the next set of services." Simple interoperability points could also identified which allow use of existing services, mitigating issues voiced already by OGC Membership and the geospatial community. This approach would increase interoperability, enable data services reuse, support client and server applications with a consistent set of APIs, reduce development costs and risk, and promote adoption of OGC specifications in the market. *Note: This comment is a revision of the officially submitted comment.*		
AM-2	General	Major	Please provide comprehensive security considerations. This submission touches new ground in OGC standardizing describing an API for all OGC Web Services. It is hard to believe that such a comprehensive submission does not provide any security considerations at all. And this despite the fact that various activities in OGC regarding security have been taken place; also most recently. As it is good practice to follow other standardization organizations such as OASIS and IETF, I do encourage the submitting organizations to provide comprehensive	Security	(Editors comment: Agreed that we should add appropriate security considerations. It should be noted that the same should be done in all OGC standards, too, not just this specification.) The SWG agreed to focus on known aspects that are specific to the GeoServices REST API,

			security considerations outlining how security regarding confidentiality, integrity, authentication and authorization can be achieved. In particular please include a normative section regarding the use of HTTP error codes and exceptions in cases where a service endpoint requires authentication or the access is not authorized.		a new informative annex has been added. Any general concerns related to geo web services should be addressed by a general paper that could be developed by the Security working groups.
EK-3	General	Major	OGC Specifications are already being extended to support REST. RECOMMENDATION We recommend that OGC members continue to extend existing WxS standards to support REST, and decline the proposed standard entitled "GeoServices REST API". ALTERNATIVE A - Extend WxS to support REST OGC members have already been active in introducing support for REST with the adoption of the OGC WMTS specification in June 2010. Currently the WFS working group is extending WFS to support REST, while continuing support for existing bindings (KVP-Get, XML-POST, and SOAP). See CR# 157, OGC Document # 11-080 titled "A REST binding for WFS 2.0", dated 2011-07-15 at https://portal.opengeospatial.org/files/?artifact_id=44852 for more details. It is expected that the expertise gained from this effort through a standard and collaborative OGC consensus process will allow OGC members to introduce support for REST to all other WxS at a much faster pace than the time required	OGC Baseline	(Editors comment: See section 2.2 for a discussion of the issue raised in this comment.) This comment cannot be implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration by the OGC membership during the adoption vote and for consideration during future major versions of the standard.

to adopt the GeoServices REST API.

Benefits:

- This enhancement process will produce acceptable REST standards that meet current REST requirements much sooner than the proposed specification, which is entirely new, redundant and incompatible with current WxS specifications.
- These new WxS-REST extensions are simple and conceptually similar to the effort already undertaken by OGC members to introduce other bindings to OGC WxS.
- It preserves OGC's investment into OGC core geospatial web services standards, vendors' investment into products using those standards, significant investments made by OGC Sponsors for commissioning such interoperable standards, and reduces overall financial risks for a world-wide geospatial community who has already deployed OGC-based SDI solutions.
- It encourages product vendors to support richer products using more than one protocol within a single product, leading to better interoperability without increasing technical support costs.
- It permits web services to support more than one protocol accessing the same content without duplicating storage or synchronize update transactions, and without increasing operational costs.
- It supports an XML-based encoding.

- It supports GeoJSON, a popular open data format encoding.
- This approach, when extended in the decades to come to newer technologies, provides a graceful adoption of future innovations and protects investments in software & services & content.

We believe that none of the benefits from "Extending WxS to support REST" above would be possible with the proposed "GeoServices REST API", because of severe flaws in the concept, approach and implementation of the proposed specification.

ALTERNATIVE B - "GeoServices REST API"

After the initial comments received by the OGC REST SWG and over a year of effort to address them, the proposed "GeoServices REST API" still does not attempt to comply with, nor accommodate, existing OGC WxS standards; nor does it provide support for industry standard XML, nor GeoJSON. The proposed "GeoServices REST API", as proposed, is an inferior, confusing, single-vendor product that is incompatible and directly competing with existing OGC WxS standards in the same market.

The proposed "GeoServices REST API" is not necessary, given that current core OGC standards, and reference products based on them, are already being extended to support REST, probably with shorter

			delivery & approval times.		
EK-4	General	Major	Market Consequences: Proceeding with the adoption of the "GeoServices REST API" would harm everyone who has invested in OGC standards or products based on those standards, and inhibit industry adoption of OGC standards. If OGC members were to sanction this privately-developed service interface as a standard, one vendor would be handed a dominant lead, discouraging competition, and enabling the dominant vendor to arbitrarily control and extend the standard. Furthermore, ESRI CTO Andrew Turner in an official response to the RFC suggested "to adopt a phased approach to the proposal, feedback, development and acceptance of these parts." This may take years before acceptable standards are approved. During these years of uncertainty, risk-averse customers will avoid moving to standard-based products. During these years of uncertainty, few firms or sponsors would invest in the development of OGC standards or products until the proposed standards are actually approved, prolonging a monopolistic situation for the proponent. Alternatively, if the GeoServices REST API is accepted in the current multi-part form, customers will be confused as to the "old" and "new" OGC standards (which are incompatible at all levels). Confusing the market this way induces customers to defer purchasing of products based on so-called "old" standards (the	OGC Baseline	(Editors comment: See section 2.2 for a discussion of the issue raised in this comment.) No change proposed. This comment will be retained for consideration by the OGC membership during the adoption vote. See 2.2.

			only ones actually already approved by OGC). In the meantime the proponent may market their product as meeting a proposed "new" OGC standard, displacing products based on "old" OGC standards. It reduces the revenues of firms who contributed toward open OGC standards, invested in building compliant products, and have now started expensive marketing of those products to customers only now becoming aware of the advantages of OGC standards. Vendors of products that meet actual OGC standards would suffer, lose faith in OGC and may be forced to exit. Proceeding with this proposed incompatible standard repudiates OGC principles calling for interoperable web service interface standards that give a fair chance to all software vendors (including open source vendors) to reach customers with innovations that plug-in with other existing products, and that give customers greater choice of quality products from a healthy competitive ecosystem. Those who trusted that adopting OGC standards would protect their investment and give them access to an ecosystem of innovators will feel betrayed. In summary, allowing "GeoServices REST API" to proceed will chill the OGC web services market, and deter customers and developers from investing in products based on the excellent, already approved OGC standards.		
JH-2	General	Major	Phased Approach - Extend each existing WxS standard for REST	OGC Baseline	(Editors comment: See section 2.2 for a discussion of the issue	

The "phased approach" should be to rapidly extend each existing OGC WxS standard for REST, one by one. This will produce consensus-based, supportable REST standards much sooner than GeoServices REST API.

In my previous post, I discussed how "a phased approach ..., feedback, harmonization, development and acceptance of these parts" would be necessary before the GeoServices REST API Candidate could be accepted, because it would give time for "integration of interoperability points to enable the use of existing OGC Services in a hybrid architecture" and "allow developers to build reference implementations and verify concepts".

The post on "Extending current OGC WxS to support REST" outlines the most efficient way of achieving these goals - extending the approved WxS protocol bindings to WxS-REST. This builds on standards we already have instead of replicating them. Extending standards avoids overlap, duplication and confusion, shortens learning time for developers, and avoids the cost and delay of building new compliance tests, writing new documentation, and doubling the support effort.

I earlier discussed that it is important to "allow developers to build reference implementations and verify concepts". Reference implementations already exist for the core WxS family, and it will be straightforward to enhance them for the extended standards, saving even more effort and money for

raised in this comment.)

This comment cannot be implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration by the OGC membership during the adoption vote and for consideration during future major versions of the standard.

			OGC and developers alike. OGC accepted WMTS-REST in 2010, and has only months to go for WFS-REST. It's apparent we should not pursue redundant REST standards, so the most efficient choice is to continue extending WxS to support REST, enhance them with elements of 'GeoServices REST' functionality needed for the future, add support to GeoJSON and drop GeoServices REST API from consideration. In summary, the "phased approach" should be to extend each existing WxS standard for REST, one by one. This approach would increase interoperability, enable data services reuse, support client and server applications with a consistent set of APIs, reduce development costs and risk, and promote adoption of OGC specifications in the market.		
ML-1	General	Major	There are numerous problems with the standard and the specification. Some are already commented on by others, but in addition another non-exhaustive list: - It does not support cross-origin resource sharing (http:www.w3.org/TR/cors/). - Export Map does not specify ordering of layers, unlike WMS - Export Map does not support spatial filtering - Export Map does not support custom styles (for highlighting for example) - The Export Map "layerDef" expression is different from not only OGC CQL but also the "where" parameter for the Query service	OGC Baseline, Security, Specific Changes	(Editors comment: The GeoServices REST API is not intended to cover all the functionality supported by the OWS standards. Additional capabilities may be considered, but for the examples listed we would recommend to do this is a future revision. The comment about the URI name registration is

- It is a serious flaw that the "where" parameter for the Query service is specified as only "any legal SQL WHERE clause" without a grammar.
- Separating the spatial query relation and geometry means limited spatial querying where CQL like "geometry intersects polygon(...) OR other_attribute = ..." and "geometry intersects polygon(...) and geometry intersects linestring(...)"are not supported. DWITHIN and BEYOND are not supported either.
- Not supporting response paging is a serious limitation as is the lack of a sorting capability.
- It is apparently the intention to establish a new naming authority for "urn:ogc:def:crs:gsr" and "urn:ogc:def:uom:gsr" URN's which - if submitted at all - will only serve to codify the incompatibilities and redundancy with existing naming authorities. If the naming authority is submitted to OGC-NA it will have to be seen if it is accepted and will live up to the responsibilities in ISO 19135 (referenced from OGC 09-046r2) and make any effort to resolve the noted incompatibilities with other authorities concerning referencing.

Apart from these problems and others, the "12-062r1" document is a lengthy document trying to provide a justification for the standard. Whether it is convincing anyone that this standard is a serious effort to promote interoperability or just an attempt at standardization of a single-vendor implementation with all its limitations

not understood. If concepts are defined as part of the specification, it is customary in OGC to define them within the authority of the specification. In any case, the name registrations will be submitted to the OGC Naming Authority and any issues will be resolved during this process.

See 2.8 and AC-6 on CORS.

It is unclear where the flaw in the "where" parameter is as the syntax is given by SQL and the feature model.

Regarding the relationship with the OWS standards, see section 2.2.)

For most parts, this comment cannot be

			and technological debt is up to the community to decide. Our opinion is that this draft standard is be a step backwards for interoperability and technical progress for the spatial community and efforts are better directed to improving the WxS standards instead of creating limiting, incompatible, under-defined and already outdated new standards.		implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration by the OGC membership during the adoption vote and for consideration during future major versions of the standard. Regarding CORS: see 2.8 Regarding URIs: After a discussion with the OGC-NA chair, we will not define URIs for the CRSs as at the moment they will not be referred to by URI, only by the wkid. The CRS URIs have been removed.
CG-1	General	Major	JSON is a great output format for web applications; however, the mandated JSON (Java Script Object Notation) output is a perceived weakness. It limits the clients to web pages running Java Script (aka ECMA-Script). All of the other OGC standards that we use	JSON vs XML	(Editors comment: JSON is not limited to clients in web pages. Many web developers now prefer JSON over XML, so this

			mandate output of some form of XML - typically GML. This enables a client-agnostic approach as any Java or C++ program can parse and manipulate the generated XML, as can any web page that uses AJAX (Asynchronous Javascript And Xml).		addresses a demand. If there is sufficient demand, an XML representation could be supported in a future revision, too.) JSON support cannot be removed without breaking backward compatibility with current implementations and thus is out of scope for this version. Adding XML as an output format will be retained for consideration for later versions of the standard.
CG-2	Part 4: Feature Service (12-057r1), page 64 Table 36 and page 67 Table 38	Editorial	Part 4 has misspelled "attachment" as "attachement" in several places, including parameter specifications (attachmentId(s): The parameters are misspelled: {attachementId(s)} vice {attachmentId(s)}). Anyone attempting to implement this standard would be required to accept the misspelled parameter name.	Errors and clarifications	(Editors comment: Thanks for catching this.) Accepted
CG-3	Part 1: Core (12-054r1), page viii	Minor	There is a loophole in the backward compatibility rules. The proposed specification seems to imply that backwards compatibility is not certain. This may just be a disclaimer, but who gets to decide? OGC? Vendors?	Errors and clarifications	(Editors comment: This is not meant to open a loophole for implementations, but to allow OGC members to

					approve non-backwards-compatible changes in the specification, if there are good reasons. However, this would require a positive adoption vote by OGC members, so this is full control of the OGC membership. We should discuss if the wording should be changed to clarify this.) Clarification text added.
CG-4	Part 1: Core (12-054r1), page11 Section 6.2.2	Editorial	At the beginning of this section, there is an incomplete sentence.	Errors and clarifications	Changed to " Since the API was originally developed **several years** ago, ..."
CG-5	Part 1: Core (12-054r1), page 20 9.6	Minor	The order is contrary to other specifications like KML.	Specific Changes	(Editors comment: Yes, different conventions exist. However, changing the orientation would break backwards compatibility.) This comment cannot be implemented without breaking backward

					compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration for later major versions of the standard.
CG-6	Part 1: Core (12-054r1), page 28 Section 11.2	Minor	Color example is opposite the OGC KML standard and uses RGB vice hexadecimal. These should be consistent in the OGC family of standards.	Specific Changes	(Editors comment: Same as CG-5.) This comment cannot be implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration for later major versions of the standard.
CG-7	Part 4: Feature Service (12-057r1), page 6 Section 7.2.2	Minor	The "FeatureServer" endpoint is required. Why? We should be able to modify the endpoint. Why not "fs" or whatever the implementer chooses?	Specific Changes	(Editors comment: The fixed pattern is needed, so that the relative URI from the catalog is clear.) No change required.
CG-8	Part 4: Feature Service (12-057r1),	Major	geometry: The note that coordinates always use a	Specific	(Editors comment: As the JSON is for machine-to-

	page 21 Table 12		decimal separator breaks i18n rules.	Changes	machine communication i18n rules do not really apply here. GML, the OWS standards as well as XML Schema and JSON Schema do this, too.) This comment cannot be implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration for later major versions of the standard.
CG-9	Part 4: Feature Service (12-057r1), page 21 Table 12	Major	where: This has the potential to open up a SQL injection attack.	Security	(Editors comment: It is assumed that implementations will take the necessary precautions to prevent such attacks, e.g. by parsing the where clause and constructing the SQL statement after validation against the feature model, etc. This should be

					clarified as part of the security considerations, see AM-2.) Include a statement in the Security Consideration clause.
SC-1	Part 3: Map Service (12-056r1), Section: 20.2.3, page: 65, 66	Minor	To get data from a JSON structure, you must know exactly where it is located or iterate over everything until you find it. *The comment had not been understood and the following clarification was provided:* If data/objects about natural hierarchical relationships (tree based) are stored in one JSON file then how to retrieve inner objects? For example, data/objects about a 'Company', its 'Employee' and various projects on which Employees are working is stored in one JSON file, then how to retrieve information about 'A list of projects on which a given Employee is working'? Is there any API/call which will allow a programmer to retrieve required objects using iterations?	Questions	In the example case, company, employee and project would be separate object classes with relationships between them. The feature service provides the capabilities to query, for example, the projects an employee is working on using Query Related Records.
SC-2	JSON Schemas and Examples(12-068r1)	Major	Is there any standard available to convert JSON in to other formats like GeoJSON, XML, HTML representation etc? There is no easy way to change JSON data into other data format. DeserializeJSON and SerializeJSON requires to convert JSON data into another data type, other data format into JSON data	Questions	(Editors comment: See 2.3 for GeoJSON. There are numerous rules how to convert JSON to other formats, in particular

			respectively.		XML[8].) No change proposed.
SC-3	Part 4: Feature Service (12-057r1), Section: 8 & 9, page: 26, 27,29	Major	The only way to parse JSON into Java Script objects is to use eval() function. This function is quite known to all, so any attacker/Intruder can misuse eval() function and perform data modification. There is a need to decrypt the data. Modification is harmless for Feature service because end users are able to change an attribute of a service.	Security	(Editors comment: The problems with using eval() in JavaScript have been well documented over the years. There are varying alternate solutions and advice available for dealing with the specifics[9]. The scope of this falls outside the discussion of the Geoservices REST API spec. The topic should be mentioned in the security considerations section, see AM-2.) Discuss this risk as part of the security considerations section, see AM-2.
SC-4	Relationship with the OGC baseline(12-062r1), page: 5	Major	JSON does not have a native hyperlink type. It may create problems as it may be unacceptable in a web format. For example a REST interface requires native links and link types. For example, there is no way to find, &employee;id; (part of URI) and more data about	REST	(Editors comment: See section 2.4 for a discussion of this issue. For the moment the knowledge of the

8" see for example: JsonML http://www.jsonml.org/, BadgerFish http://badgerfish.ning.com/, RayFish http://onperl.org/blog/onperl/page/rayfish/, or http://www.bramstein.com/projects/xsltjson/

9" http://stackoverflow.com/questions/86513/why-is-using-the-javascript-eval-function-a-bad-idea is just one example

			that thing can be found over &employee;info;. One needs to provide everything in one JSON file, or specify out-of-bound. The question is how a client can find other piece of data?		specification is needed to know the links to other resources. What should be added to clarify the links is to add the links for each resource.) Explicit links cannot be implemented without breaking backward compatibility with current implementations and thus is out of scope for this version. It will be retained for consideration for later major versions of the standard. But see also 2.4.
SC-5	General	Minor	How JSON can be more relevant as an output format rather than GeoJSON to pull GeoSpatial data? Is there any standard available to convert JSON in to GeoJSON?	GeoJSON	(Editors comment: The feature and geometry encoding of the GeoServices JSON is quite similar to the GeoJSON encoding. See section 2.3. This can also be seen from existing code that converts between GeoJSON to the GeoServices JSON.)

					No change proposed. See also 2.3.
SC-6	General	Editorial	ESRI REST API has been available since few months. It seems it has not been used widely. One such example is 'Arc2Earth's cloud' work. It will be interesting to know many other projects using ESRI REST API to get concrete examples.	?	(Editors comment: see 2.2 for some examples) No change proposed. See also 2.2 for some references.
SC-7	General	Minor	In the core document, how JSON can be used with REST for pulling data from server/database is demonstrated but there should be pictorial explanation which shows how RESTful web service is enabled to pull the data. *The comment had not been understood and the following clarification was provided:* Is there any tutorial type demonstration showing development of one RESTful service using and/or processing JSON file containing objects? It will become effective for a beginner to learn programming, if each and every call is explained to manage life cycle of a given object for a specific use case scenario or example.	Questions	Tutorial material is out of scope for the standard or our SWG, but of course very helpful to be available in addition to the specification. Existing tutorial material will at this time be, of course, mostly from Esri and tailored towards their implementation. The comment will be addressed by adding a page on OGC Network with additional information about the GeoServices REST API once it has been approved. This page will include links to tutorial material. This page will be listed in the

bibliography of part 1.

The comments have been submitted by:

AC	Adrian Custer, ac@pocz.org
PB	Peter Baumann, p.baumann@jacobs-university.de
PC	Pat Cappelaere, Vightel Corporation, pat@vightel.com
ER	Even Rouault, even.rouault@mines-paris.org
HT	Håvard Tveite, UMB - havard.tveite@umb.no
VM	Volker Mische, volker@couchbase.com
CS	Cameron Shorter
DN	Douglas Nebert, USGS
AM	Andreas Matheus, Secure Dimensions GmbH
EK	Edric Keighan, CubeWerx Inc. some comments co-authored by Panagiotis (Peter) A. Vretanos and Jim Supple
TS	Tim Schaub, tschaub@opengeo.org
AT	Andrew Turner, Esri
JH	Jeff Harrison, The Carbon Project
ML	Matthijs Laan, B3Partners, matthijslaan@b3partners.nl
CG	Chris Guthrie
SC	Dr. Sanjay Chaudhary, DA-IICT, sanjay_chaudhary@daiict.ac.in