
Subject: [railML3] Proposing new @epsgCode for <geometricPositioningSystem>
Posted by [Marharyta Vyskarka](#) on Tue, 17 Feb 2026 10:38:37 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hello everyone,

We have been taking an extensive look at the attributes related to the reference systems. It was noticed that there are multiple such attributes between railML 2 and 3. For example, railML 2 has <geoCoord>/@epsgCode [1], railML 3 has @srsName (in such elements: <lineString> [2], <posList> [3], <point> [4], <pos> [5]) and <geometricPositioningSystem>/@crsDefinition [6].

It is also worth to note that the mentioned attributes have different types of data (xs:anyURI and xs:string), which doesn't put many restrictions on the expected data. Both of these options allow such values as "4326", "EPSG:4326", "urn:ogc:def:crs:EPSG::4326", "urn:ogc:def:crs:EPSG:9.8.7:4326", "https://www.opengis.net/def/crs/EPSG/0/4326".

To bring more clarity and to be consistent we wanted to propose replacing @crsDefinition [6] by a new attribute @epsgCode (xs:int) in railML 3.4. This proposal is based on the idea that the EPSG code was intended by the previous crsDefinition, and that making it just a code instead of a relative URI would make it more uniform, removing ambiguity. I also want to note that @srsName would stay the same, as it is part of GML, and we cannot change it.

Therefore I want to ask, does anyone use anything else other than EPSG for @crsDefinition? Are there any other concerns regarding such change?

- [1] <https://wiki2.railml.org/wiki/IS:geoCoord>
- [2] <https://wiki3.railml.org/wiki/IS:lineString>
- [3] <https://wiki3.railml.org/wiki/GML4RAILML:posList>
- [4] <https://wiki3.railml.org/wiki/IS:point>
- [5] <https://wiki3.railml.org/wiki/GML4RAILML:pos>
- [6] <https://wiki3.railml.org/wiki/RTM:geometricPositioningSystem>

Best regards,
Margo Vyskarka

Subject: Re: [railml3] Proposing new @epsgCode for
<geometricPositioningSystem>
Posted by [Mathias Vanden Auweele](#) on Sun, 22 Feb 2026 10:52:11 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hello Marharyta,

I'm not sure I think it's a good idea to go from @crsDefinition to @epsgCode because of the loss of functionality:

- no possibility to reference non-epsg CRS (probably not a very big problem but I was planning on using this in the future for a more schematic representation of the network)

- no longer dereferenceable, I really like the use of URI's instead of ints and strings... Makes it much easier to really understand what you are reading
- axis order... just the EPSG will need the data consumer to assume axis ordering.. OGC:CRS84 (lon/lat) vs EPSG:4326 (lat/lon). Although there is a lot of confusion about this and some implementations are not doing it correctly
- from memory, there is also something about the vertical axis

I did some research about this a while ago and my conclusion is to prefer URI's. It also seems strange to have GML on the one side that uses srsName and then a simple epsgcode on the other side.

I don't think I would agree with this statement: "This proposal is based on the idea that the EPSG code was intended by the previous crsDefinition". I know several people that have developed RTM and I don't think that they would have chosen crsDefinition if they meant to use EPSG code. So for me, that's really intentional.

Although I would very much agree on a more strick set of possible values for the crsDefinition. I would force the use of URI's and disallow the use of strings such as EPSG:4326

Subject: Re: [railml3] Proposing new @epsgCode for <geometricPositioningSystem>
Posted by [Milan Wölke](#) on Wed, 25 Feb 2026 17:15:08 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hi Mathias,

The problem is, that URI is not a URL. So your example of EPSG:4326 is an URI but it doesn't help you understanding the way the coordinates are formed. The idea of using a simple int is to work around this, so that an importer can decide if he can work with the coordinates or not. After all the vast majority of systems will only support a limited set of coordinate systems that is specifically implemented. Converting CH1903 or DB_REF to WGS84 is not always straight-forward.

The problem with the string or URI approach is that not even the format of the URI is standardized. You can indicate usage of WGS84 with an URI like this: EPSG:4326 or like this: urn:ogc:def:crs:EPSG::4326. I have also seen sth like this:
<https://spatialreference.org/ref/epsg/4326/projjson.json>

When implementing an importer I think the only thing you can do is to use a regular expression, and that is probably not exactly safe.

Does this make sense to you, or am I overlooking something here?

Best regards, Milan

Subject: Re: [railml3] Proposing new @epsgCode for

<geometricPositioningSystem>

Posted by [Mathias Vanden Auweele](#) on Wed, 25 Feb 2026 23:36:25 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi Milan,

Yes you are correct and I needed to be more clear.

This is written in the W3C & OGC standard GeoSPARQL specification

<https://docs.ogc.org/is/22-047r1/22-047r1.html>:

Quote:The OGC maintains a set of SRS IRIs under the <http://www.opengis.net/def/crs/> namespace and IRIs from this set are recommended for use. However others may also be used, as long as they are valid IRIs.

So while I would allow the use of ANY URI in railml for the crs, it will be up to the software supplier to be very clear in which CRS that he will support. And by listing these as OGC managed SRS IRI's, it becomes very very clear what is meant. Furthermore, conversion between OGC managed CRS is trivial with libraries such as betterknown <https://github.com/placemark/betterknown> and proj4

Subject: Re: [railml3] Proposing new @epsgCode for
<geometricPositioningSystem>

Posted by [Milan Wölke](#) on Thu, 26 Feb 2026 11:13:45 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi Mathias,

so what you are trying to say is, that you would only allow URIs that point to valid OGC WTK definitions? Because otherwise the software supplier would not just need to be specific about what CRS is used in the software but also about the way it is indicated.

BTW, I wouldn't use trivial when the actual reprojection that is necessary for the conversion is not part of the project you are referencing (at least as far as I could see at a first glance).

Best regards, Milan

Subject: Re: [railml3] Proposing new @epsgCode for
<geometricPositioningSystem>

Posted by [Mathias Vanden Auweele](#) on Fri, 27 Feb 2026 23:51:01 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi Milan,

No, I would allow all URI's for CRS in railML. And then it's up to the software vendor to specify which URI CRS's he supports. And he could say that he supports all <http://www.opengis.net/def/crs/> URI's for example. Or only WGS84 or UTM32.

I would definitely also allow custom URI's so that railML providers can use their own coordinate

system URI and request software vendors to support them by providing them with the required information to do so.

I haven't had any issues with conversions between coordinate systems (for OGC defined ones), but I might not understand what coordinate systems that you are referring to. For my yasgui fork, it happens on the fly from any coordinate system: <https://shorter.matdata.eu/w>

Subject: Re: [railml3] Proposing new @epsgCode for <geometricPositioningSystem>

Posted by [Thomas Nygreen](#) on Thu, 09 Apr 2026 15:47:13 GMT

[View Forum Message](#) <> [Reply to Message](#)

Dear Mathias,

Of the concerns you raise in your first reply, I think the main one is the first: if we decide to replace crsDefinition with epsgCode it is exactly to restrict references to EPSG CRSs in a completely unambiguous manner, which of course removes the option to reference non-EPSG CRSs.

Regarding your other concerns:

- An int is not directly dereferenceable, but consider it equivalent to require a URI starting with "http://www.opengis.net/def/crs/EPSSG/0/". Most valid URIs are also not dereferenceable, so unless we impose further restrictions than "any URI", this concern will remain.
- The data consumer does *not* have to assume the axis ordering of an EPSG CRS. It is given in the CRS definition that is as uniquely identified with an int as with an opengis URI. OGC:CRS84 is not the same CRS as EPSG:4326, because the axes are switched, and their definitions are therefore not the same.

I agree that this would introduce one more difference between the railML and GML modelling, and we will need a best practice for srsName in gmlLocation no matter what we do with crsDefinition (because we cannot change the syntax of GML). From OGC, the GML recommendation has for quite many years been to use a dereferenceable URI/IRI, but for "well-known references it is not required" [GML 3.2 specs], allowing for things like the short strings mentioned above. As far as I know, the definition of "well-known" is left to the exchanging parties. Before recommending URIs, the recommendation was to use URNs. OGC no longer operates the URN resolver, but they can easily be translated to resolvable URIs.

I agree that the flexibility to use non-EPSG CRSs was probably intended. And this is not the only issue where we see that originally intended flexibility when exporting creates too much ambiguity or too many cases that importers have to prepare for.

You write that you would like a more strict value space for crsDefinition. You also write that you would allow any URI. In XML Schema Definition anyURI also allows relative URI references, allowing basically any string, including things like "WGS84", "4326" or "EPSG:4326" (which could in theory resolve in the unlikely case that a resource with that name is available in the same directory as the railML file), or a URN (which requires a dedicated resolver).

Do you have any suggestions on how you would restrict the value space of crsDefinition,

syntactically or by best practice?

Best regards,
Thomas
