
Subject: [all versions] Using sequences in XSD schemas
Posted by [Thomas Nygreen](#) on Thu, 06 May 2021 20:37:31 GMT
[View Forum Message](#) <> [Reply to Message](#)

Dear community!

A topic that arises from time to time is the use of `xs:sequence` vs. `xs:all` in the railML XSDs. Together with `xs:choice`, these are three options for how to attach child elements to a parent. I'll try to sum up their differences briefly:

sequence:

- * the multiplicity of the children is specified individually
- * children must appear in the same order as in the XSD
- * supports `xs:any`

all:

- * the multiplicity of the children can only be 0..1 or 1..1, and is specified individually
- * children can appear in any order
- * does not support `xs:any`

choice:

- * designed for selecting one child from a set of possible children

The most common background for these questions is that with `xs:sequence`, the order of the child elements has to be the same as in the schema file, while with `xs:all` the order is arbitrary. This may seem simpler, but it leads to ambiguities more easily than with the stricter `xs:sequence`. That is why `xs:all` neither supports `xs:any` or children with multiplicity greater than 1. With the restricted multiplicity, `xs:all` is mostly not suitable for our purposes, apart from in the root element and the "views", where no children have multiplicity greater than 1.

Since `xs:all` has this limited functionality, and since mixing `xs:all` and `xs:sequence` depending on the multiplicity of the children will most likely confuse developers using railML as to when the order of the children matters, the coordinators recommend that we keep using `xs:sequence` for railML.

As a side note, there is one element in railML 3 element with `xs:all`, and that is the `<railML>` root element, allowing an arbitrary order of the top-level elements for each schema. So you can have `<timetable>` before `<infrastructure>` if you want. I don't know why, but it is there already, so we should probably not change it.

Best regards,
Thomas Nygreen
