Background

• Bugfixes and clarifications. Our view is that this is not supposed to be NETCONF v2

• The issues we present are based on implementation experience, interoperability testing of three independent implementations, and mailing list comments.
• `<bad-namespace>`
  - Is a xs:QName. Should be xs:uri (or xs:string).

• `<error-app-tag>`
  - Is a xs:string which means that there is a single flat naming scope for all app-tags. Should be xs:QName, which makes the app-tags scoped by namespace.

  - One example is wrong.
• The validate operation can validate a data store or an inline configuration subtree.
  
  – The problem is that it is unclear what this configuration subtree is. Is it like in edit-config, with operation attributes? If so, there is no way to specify the default operation like in edit-config.
  
  – Does anyone implement inline validation?

• Proposed solution: remove inline validation, and add a test-option parameter 'test-only' to be used in edit-config. (at least two implementations support this already)
• Clarify what 'startup' is.

• Clarify what delete of startup means (reset to factory defaults?)

• Is :startup and :candidate allowed?

• Is :startup and :confirmed-commit allowed?
  – If so, specify how it works.

• Fix XSD so that startup cannot be a target to edit-config.
• The error-type refers to 'protocol', 'application', 'rpc','transport' while the layer model has transport, rpc, operations, content. It is not clear how this relates.

• The error-types are not defined. What is the difference between 'rpc' and 'protocol'?

• Is the error-type really needed?

• It seems error-severity is always "error" and so the this error element seems unneeded and perhaps should be removed.
• Clarify the intended meaning of continue-on-error. Specifically, what does the “error” mean? Also, clarify that the rpc error partial-operation MUST (?) be returned if such an error occurs.

• The error-info elements in partial-operation (ok-element, err-element etc) are defined as xs:QName. Is this just a bug? Should they be XPath strings?

```xml
<x:interface>
  <x:name>eth0</x:name>
  <x:type>atm</x:type>
</x:interface>
<x:server>
  <x:name>my web</x:name>
  <x:type>http</x:type>
</x:server>
```

What does <err-element>x:type<err-element> refer to?
• Allow rpc-error inline in <data> reply.
  – The problem is how internal errors are reported during <get> and <get-config> processing without requiring the agent to first buffer the complete reply.

• Align the XSD with the text about rpc-reply – the rpc-reply element should allow any other element, not just <ok>, <data> and <rpc-error>.
  – Operations that need to return something should stick this something directly under <rpc-reply>, not <data>.
• Return from XPath filter. Suppose an XPath expression selects a text node - how should the XML look? E.g. "/system/sysName/text()". We always return a XML subtree, i.e. in this case, we would return "<system><sysName>foo</sysName></system>" not just "foo".

• The XPath context should be properly defined for the select attribute and error-path.

• Clarify what the error-path points to. Always something in the <rpc> request instance document? What if the operation is validate of candidate, and validation fails for some element in the data store?
• Clarify the intention of the XSD. Specifically if capabilities are allowed to modify existing operation, although the XSD does not really allow it. For example, suppose a capability adds an “test-option” enumeration – is that allowed?
• Clarify that an XML preamble is optional.
  – `<?xml version="1.0" encoding="utf-8"?>`

• RFC 4741 allows arbitrary content of the message-id attribute.
  – Some implementations seem to run into problems if the message-id (or other attributes) contains "]]>]]>" or "</rpc>". Perhaps this is not a problem with RFC 4741 per se but just an implementation problem
  – But arbitrary complicated and arbitrary long message-id attributes also do not seem very useful to have.
RFC 4741 requires that all attributes of an `<rpc>` are returned in the `<rpc-reply>`.

- Is the intention really that xmlns attributes also are returned as-is?
- This can lead to duplicated attributes and invalid XML documents. (A good example is a namespace attribute which is echoed back while the implementation also generates a second namespace attribute.)