<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE rfc SYSTEM "http://xml.resource.org/authoring/rfc2629.dtd"

[

<!ENTITY RFC7159 PUBLIC ''

'http://xml.resource.org/public/rfc/bibxml/reference.RFC.7159.xml'>

]>

<?rfc toc="true"?>

<rfc category="std" docName="draft-cordell-jcr-co-constraints-00" ipr="trust200902">

<front>

<title abbrev="JCR Co-Constraints">Co-Constraints for JSON Content Rules</title>

<author fullname="Pete Cordell" initials="P." surname="Cordell">

<organization>Codalogic</organization>

<address>

<postal>

<street>PO Box 30</street>

<city>Ipswitch</city>

<country>UK</country>

<code>IPX 2WY</code>

</postal>

<email>pete.cordell@codalogic.com</email>

<uri>http://www.codalogic.com</uri>

</address>

</author>

<author fullname="Andrew Lee Newton" initials="A.L." surname="Newton">

<organization abbrev="ARIN">American Registry for Internet Numbers</organization>

<address>

<postal>

<street>3635 Concorde Parkway</street>

<city>Chantilly</city>

<region>VA</region>

<country>US</country>

<code>20151</code>

</postal>

<email>andy@arin.net</email>

<uri>http://www.arin.net</uri>

</address>

</author>

<date/>

<abstract>

<t>

JSON Content Rules (JCR) provides a powerful, intuitive and concise method for defining the structure of <xref target="RFC7159">JSON</xref> messages. However, modern JSON usage patterns occasionally mean that JCR alone is not able to capture the required constraints in a satisfactory way. The document describes JCR Co-Constraints (JCRCC) which defines additional JCR directives and annotations that can be added to a JCR ruleset in order to define more detailed constraints on JSON messages.

</t>

</abstract>

</front>

<middle>

<section title="Introduction">

<t>

JSON Content Rules provides a powerful, intuitive and concise method for defining the structure of JSON messages. In addition to describing the overall structure of JSON messages, JCR aims capture the constraints that are imposed on individual items within a message. However, modern JSON usage occasionally requires constraints that can’t be expressed by JCR alone. JCR Co-Constraints (JCRCC) defines additional JCR directives and annotations that can be added to a JCR ruleset in order to define more detailed constraints on items within a JSON message, and also supports specifying constraints that depend on the relationship of multiple JSON items.

</t>

<t>

JCRCC constraints represent an additional layer of validation on top of the validation offered by JCR alone. JCRCC constraints may indicate that a JSON instance that was determined to be valid by the rules of a JCR ruleset, is in fact invalid. However, if the JCR ruleset indicates that the JSON instance is invalid, JCRCC constraints can not override that and declare the instance valid. A JCR processor may ignore the JCRCC annotations and directives, perhaps only issuing a warning for encountering an unknown annotation or directive.

</t>

<t>

JCRCC uses the annotations @{id}, @{when} and @{assert} along with the directive #{constraint}. The @{id} annotation is used to indicate an item in a JSON message that contributes to the assessment of a JSON instances validity. The other three each include a 'condition' expression that yields a Boolean true or false result. The validity of the JSON instance is dependent on the results of the various condition expressions. Condition expressions are made up of identifiers, comparators, combiners and functions. Processing of the condition expressions is triggered according to a 'conceptual processing model'. Each of these aspects is described in more detail below.

</t>

</section>

<section title="Annotations and Directives">

<t>

JCRCC uses the annotations @{id}, @{when} and @{assert} plus the directive #{constraint}.</t>

<section title="The @{id} Annotation">

<t>

The @{id} annotation indicates a rule in a JCR ruleset that corresponds to an item in a JSON instance that is involved in assessing the validity of the JSON instance. It has the form:</t>

<figure>

<artwork xml:space="preserve">

@{id name}

</artwork>

</figure>

<t>where 'name' corresponds to the 'name' production in the JCR ABNF. </t>

<t>The @{id} annotation associates an identifier with the rule on which it is placed, through which condition expressions can access the JSON instance items that are mapped to the JCR rule.</t>

<t>For example, with a JCR rule of:</t>

<figure>

<artwork xml:space="preserve">

"type" @{id t} : string

</artwork>

</figure>

<t>might associate the identifier 't' with a JSON instance item such as:</t>

<figure>

<artwork xml:space="preserve">

"type" : "shutdown"

</artwork>

</figure>

</section>

<section title="The @{when} Annotation">

<t>The @{when} annotation has two similar roles. It a JCR rule indicates that a JSON instance item is optional, then it can be used to describe the conditions when the item is present or absent. Similarly, if a JCR rule indicates that an item has a group or type choice as it's type, then the @{when} annotation can be used to indicate which of the possible types is applicable in the current instance.</t>

<t>The @{when} annotation includes a single 'condition'. In the case of using the @{when} annotation with an optional instance, if the condition yields a 'true' result, then the item associated with the JCR rule should be present. When the @{when} annotation is used to select the applicable member/type within a group or type choice, the condition of each @{when} annotation is evaluated in turn (from left to right as shown in the rule) and the member/type that corresponds to the first condition that yields a 'true' result is selected. If none of the @{when} annotations on a group or type choice yields true, this indicates an invalid instance. When a member/type within a group or type choice (that has @{when} annotations on other members) is encountered, this indicates the default case. In essence, if a rule has @{when} annotations, then an absent @{when} annotation on a member/type is equivalent to @{when true}.</t>

<t>As an example, a @{when} annotation on an optional item may look as follows:</t>

<figure>

<artwork xml:space="preserve">

? "uptime" @{when $t == "shutdown"} : integer

</artwork>

</figure>

<t>This indicates that the "uptime" member should be present if the JSON instance item referenced by a @{id t} annotation has the value "shutdown".</t>

<t>A @{when} annotation on a group may look as follows:</t>

<figure>

<artwork xml:space="preserve">

details ( @{when $t == "boot"} boot-details |

@{when $t == "shutdown"} shutdown-details |

default-details )

</artwork>

</figure>

<t>This indicates that the JCR rule named 'boot-details' is applicable when the JSON instance item referenced by a @{id t} annotation has the value "boot", the rule 'shutdown-details' is applicable when the value of the $t item is "shutdown", otherwise the rule 'default-details' is applicable. (The rules identified by 'boot-details', 'shutdown-details' and 'default-details' might be groups that act as mixins for the rule in which the 'details' rule is used.)</t>

<t>To avoid circular or ambiguous dependencies, the identifiers in a @{when} annotation can only reference ancestors, siblings, and descendants that are not part of arrays or descendants of itself or sibling @{when} annotations. The latter restriction avoids needing to know whether a @{when} annotation yields 'true' in order to determine if the annotation yields 'true'.</t>

</section>

<section title="The @{assert} Annotation">

</section>

<section title="The #{constraint} Directive">

</section>

</section>

<section title="Conditions">

<section title="Identifiers">

</section>

<section title="Comparators">

<t>When an identifier referenced by a comparator is absent, then the comparison return 'false'. For example:</t>

<figure>

<artwork xml:space="preserve">

$t == "boot"

</artwork>

</figure>

<t>is equivalent to:</t>

<figure>

<artwork xml:space="preserve">

( $t &amp;&amp; $t == "boot" )

</artwork>

</figure>

<t>Similarly:</t>

<figure>

<artwork xml:space="preserve">

$t == "boot" || $other == "close"

</artwork>

</figure>

<t>is equivalent to:</t>

<figure>

<artwork xml:space="preserve">

( $t &amp;&amp; $t == "boot" ) || ( $other &amp;&amp; $other == "close" )

</artwork>

</figure>

<t>And:</t>

<figure>

<artwork xml:space="preserve">

length( $first ) > length( $second )

</artwork>

</figure>

<t>is equivalent to:</t>

<figure>

<artwork xml:space="preserve">

( $first &amp;&amp; $second &amp;&amp; length( $first ) > length( $second ) )

</artwork>

</figure>

</section>

<section title="Combiners">

</section>

<section title="Functions">

</section>

</section>

<section title="Conceptual Processing Model">

</section>

<section title="ABNF">

</section>

</middle>

<back>

<references title="Normative References">

&RFC7159;

</references>

<references title="Infomative References">

<reference anchor="ARIN\_JCR\_VALIDATOR" target="https://github.com/arineng/jcrvalidator">

<front>

<title>JSON Content Rules Validator (Work In Progress)</title>

<author>

<organization>American Registry for Internet Numbers</organization>

</author>

<date/>

</front>

</reference>

<reference anchor="CODALOGIC\_JCR\_VALIDATOR" target="https://github.com/codalogic/cl-jcr-parser">

<front>

<title>cl-jcr-parser (Work In Progress)</title>

<author>

<organization>Codalogic</organization>

</author>

<date/>

</front>

</reference>

</references>

<section title="JCR Implementations">

<t>

The following implementations, <xref target="ARIN\_JCR\_VALIDATOR"></xref> and

<xref target="CODALOGIC\_JCR\_VALIDATOR"></xref> have influenced the development

of this document.

</t>

</section>

</back>

</rfc>