3.1.5 History Mechanism

The effectiveTime and active fields in the release file enable the use of a "log style" append-only data model to track all changes to each component, providing full traceability. Once released, a row in any of these files will always remain unchanged. Historic data is supplied in the RF2release files, dating back to the first release in RF1 format in 2002.

In order to change the properties of a current component a new version of that component is created with the same identifier. This done by adding a new row to the relevant release file, with the column values updated to represent the changes. The active field must be set to true and the timestamp in the effectiveTime field indicating the nominal date on which the new version was released. Note that the existing row is not changed in any way.

To inactivate a component, a new row is added, containing the same data as the final valid version of the component, but with the active field set to false and the timestamp in the effectiveTime fi eld indicating the nominal date of the release in which the final version ceased being valid. Note again that the existing row is not changed in any way.

Where editorial policy does not allow a particular property of a component to be changed whilst keeping the same Identifier, the component as a whole is inactivated (as described above), and a new row added with a new id, the effectiveTime set to the nominal date of the release in which this version of the component became valid, and the active field set to true.

It is thus possible to see both the current values and any historical values of a component at any point in time.

New content, changes and inactivations must have the effectiveTime for the release that it appears in. Pre-releases for testing may set the effectiveTime as the date of the future scheduled release but in general the effectiveTime must not be later that the scheduled release data, Where there is a business requirement for specifying a future activation date for some components, this may be represented using reference sets.

The following example demonstrates how the *history mechanism* works on the Concept, but the same rules apply equally well to the Description,Relationship and Reference set member files. In this example, the descriptions associated with the moduleId and definitionStatusId have been shown in place of their SCTID values.

A new concept (101291009) is ad ded on the 1st July 2007:

Table 3.1.5-1: History Example - Concept Added

Id	effectiveTime	active	moduleId	definitionStatusId
101291009	20070701	1	| Module 1|	900000000000074008 | Primitive|

In the following release (on 1st January 2008), the concept is moved from |Module 1| to |Module 2|. Because the moduleId field is not immutable, the concept may be updated simply by adding a new record with the same Id.

Table 3.1.5-2: History Example - Module Change

Id	effectiveTime	active	moduleId	definitionStatusId
101291009	20070701	1	| Module 1|	900000000000074008 | Primitive|
101291009	20080101	1	| Module 2|	900000000000074008 | Primitive|

In the following release (on 1st July 2008), the concept is changed from being Primitive to being Fully defined.

Table 3.1.5-3: History Example - Definition Status Changed

Id	effectiveTime	active	moduleId	definitionStatusId
101291009	20070701	1	| Module 1|	900000000000074008 | Primitive|
101291009	20080101	1	| Module 2|	900000000000074008 | Primitive|
101291009	20080701	1	| Module 2|	900000000000073002 | Defined|

In the following release (on 1st January 2009), the concept is deactivated:

Table 3.1.5-4: History Example - Concept Made Inactive

Id	effectiveTime	active	moduleId	definitionStatusId
101291009	20070701	1	| Module 1|	900000000000074008 | Primitive|
101291009	20080101	1	| Module 2|	900000000000074008 | Primitive|
101291009	20080701	1	| Module 2|	900000000000073002 | Defined|
101291009	20090101	0	| Module 2|	900000000000074008 | Primitive|

Notes

1. At no stage in this process are previously written records ever amended. Once a record has been released in a release file, it will continue to be released in exactly the same form in future release files.
2. Changes are only recorded at the point of release in the RF2release files. If a component record is changed a number of times between releases (during an edit and review process), only the most recently amended record will be appended to the release file, not individual records showing each separate edit to the released component.
3. In the last example, as well as inactivating the concept (active=0), the definitionStatusId is changed from 900000000000073002 \| Defined\| to 900000000000074008 \| Primitive\| . In practice this change is not essential since the value of data columns is ignored when a component is inactive. Although the change is unnecessary and insignificant, it typically occurs since all the relationships of an inactive concept must also be inactive, and as a result, from the perspective of the authoring environment the concept cannot be regarded as 900000000000073002 \| D efined\| .

Related Links

- 3.1.4.1. Component features - History