Limitations

Protection

By default, any string between '{..}' is considered a template name which should be substituted. And when that template cannot be found (or have no production status), an error is raised. That can be troublesome if the string is NOT supposed to be a template. A single '{' or '}' will not cause this behaviour.

To prevent the '{' or '}' to be delimiting a template name, you can protect these characters by inserting a backslash '\'. So when the sting `one{two}three` is needed in a template, use `one\{two\}three` instead to prevent the error message of a missing template 'two'.

However if this string is used as a value of a variable, the NetYCE system must substitute the value indirectly, causing the single-protected string again to be suspected to contain a template reference. Only by double-protecting the '{..}' will the variable-based values be properly become a string: `one\\{two\\}three`.

The above behaviour is not only applicable to the templates '{..}', but also to variables '<..>'.

Newline characters

When creating commands for a job or template, sometimes it desirable to include an extra newline within the command. Often to confirm or parameterize the anticipated CLI prompt. To include such a newline, insert the sting `\n`. This will be translated as a newline character on execution.

Likewise, the string `\r` will be translated in a carriage-return and a `\t` becomes a tab character. To create the old DOS carriage-return/newline, use `\r\n`.

Keyed condition:

Not supported are

```
|<param@ctx:’key’>|
|<param@ctx:’key’>=...|
```

Also the following syntax can be used:

```
|<param@ctx> = ’key’|
```

Nested conditions

This is not supported

Nested sub-templates

Nesting of sub-templates is fully supported. A circular reference of sub-templates is also feasible but is limited to 50 iterations.

Maximum index

The configuration generator in theory can work with the maximum database size. Due to several technical reasons preventing endless loops, the maximum index is limited to 5000 records. This number is far above any real time use for the number of interfaces, vlans or sla tests, to be any limitation.

Indirect parameter loops

An indirect parameter reference can result into a circular reference if `<parameter>` receives a reference to itself after substitution (the value `<parameter>`). In that case the configurator will give an error message after 50 iterations and stop generating.

Logical operators

Logical operators, like <, >, <=, >= are currently not supported.

= or != are supported. An example is given for the count function.