1. Introduction

The use of hierarchical ValueSets is supported by FHIR. At least the expansion supports nesting of subconcepts by using `ValueSet.expansion.contains.contains`.

For ValueSets the `$expand` operation (http://www.hl7.org/fhir/valueset-operation-expand.html) exists in order to generate the list of concepts the ValueSet represents based on `ValueSet.compose`. Furthermore, the `$expand` operation comes with two parameters which control how the result of an expanded hierarchical ValueSet may look like, namely `excludeNested` and `excludeNotForUI`.

Both of these parameters are part of FHIR Core.

Furthermore, for ValueSets the extensions `valueset-expand-group` (http://www.hl7.org/fhir/extension-valueset-expand-group.html) and `valueset-expand-rules` (http://www.hl7.org/fhir/extension-valueset-expand-rules.html) exist.

Both of these extensions are not part of FHIR Core.

Eventually, an example of a hierarchical ValueSet exists in FHIR Core: https://www.hl7.org/fhir/valueset-example-hierarchical.html

1.1. Current Description Of `excludeNested`

Controls whether or not the value set expansion nests codes or not (i.e. ValueSet.expansion.contains.contains)

1.2. Current Description Of `excludeNotForUI`

Controls whether or not the value set expansion is assembled for a user interface use or not. Value sets intended for User Interface might include 'abstract' codes or have nested contains with items with no code or abstract = true, with the sole purpose of helping a user navigate through the list efficiently, where as a value set not generated for UI use might be flat, and only contain the selectable codes in the value set. The exact implications of 'for UI' depend on the code system, and what properties it exposes for a terminology server to use. In the FHIR Specification itself, the value set expansions are generated with excludeNotForUI = false, and the expansions used when generated schema / code etc, or performing validation, are all excludeNotForUI = true.

2. Problems

The naming of the parameters `excludeNested` and `excludeNotForUI` together with their respective description do not make entirely clear what they are supposed to do in terms of creating an expansion for a ValueSet.

The description of the two operations `valueset-expand-group` and `valueset-expand-rules` do not clarify when and how they should be taken into account when creating an expansion. Especially, when someone uses `excludeNested` and/or `excludeNotForUI` along with these extensions.

3. Assumptions Made

- `excludeNested` and `excludeNotForUI` were primarily thought for ValueSets that included some or all codes from a CodeSystem. If the concepts within the underlying CodeSystem were structured hierarchically, one could control the result of the `$expand` operation with these two parameters. That could be the reason why these two parameters are part of FHIR Core.
- The extensions `valueset-expand-group` and `valueset-expand-rules` were added later due to someone's requirement wanting to specify a hierarchy within a ValueSet regardless of a possible hierarchy within the underlying CodeSystem(s).

3.1. Understanding Of `excludeNested`

Based on the parameter's description, we have come to the following conclusions:

- if `excludeNested` is `true` **no** hierarchy will be applied to the expansion, i.e. the expansion will be a flat list, no `ValueSet.expansion.contains.contains` exists.
- if `excludeNested` is `false` either the hierarchy of the underlying CodeSystem or the hierarchy specified within the ValueSet using the `valueset-expand-group` and `valueset-expand-rules` extensions will be applied to the expansion, if there exists any of both. Currently it is assumed that it is at the server's discretion if the CodeSystem's or the ValueSet's hierarchy definition takes precedence over the other if both exist. However, in the end `ValueSet.expansion.contains.contains` is possible.

3.2. Understanding Of `excludeNotForUI`

Up front, the name of the parameter is quite unclear. If set to `true` one might think that anything what is *not for UI* is *excluded* from the expansion. But which information of a ValueSet is *not for UI*?

Anyway, based on the parameter's description and internal discussions, we have come to the following conclusions:

- if `excludeNotForUI` is `true`, concepts which are primarily for grouping and also not selectable will be excluded from the expansion. Not selectable grouping concepts are
  - codes from the CodeSystem with a `notSelectable` property set to `true` as specified in http://www.hl7.org/fhir/codesystem.html#status and in https://hl7.org/fhir/R4/codesystem-concept-properties.html.
  - nested `contains` in the ValueSet with no code
  - nested `contains` in the ValueSet with `abstract = true`
- if `excludeNotForUI` is `false`, the expansion will include all codes defined in the compose.

3.3. Interaction Of `excludeNested` And `excludeNotForUI`

Based on the understanding of the parameters `excludeNested` and `excludeNotForUI`, the following combinations are possible with their respective outcome in the expansion described in the table.

From our point of view, only two combinations (green) make sense.

			excludeNested	
			true	**false**
			no hierarchy	hierarchy exists
excludeNotForUI	**true**	not selectable (abstract) codes are removed	**Flat list without abstract codes.**	Should result in an error as it would be difficult to create a hierarchy without abstract codes. Where to put concepts that belong to an abstract grouping code?
	false	all codes are included (incl abstract)	Flat list containing abstract codes. What should I do with not-selectable codes in a flat list?	**Hierarchy with all codes.**

4. Proposed Solution

Based on the given problems and on our assumptions, the following solutions are proposed. This will also cover `valueset-expand-group` and `valueset-expand-rules`.

4.1. Introduction Of `hierarchyMode`

The `$expand` operation should be equipped with a new parameter called `hierarchyMode` of type `code`.

The description of this parameter would be as follows:

Controls how the concepts within the expansion are structured.

- *`flat` - The expansion of the ValueSet will be a flat list. Any abstract codes that might have been used for grouping within the CodeSystem or the ValueSet will be omitted.*

- *`codeSystemBased` - The expansion of the ValueSet will be based on the underlying CodeSystem's concept definition. I.e. if there are 'abstract' codes they will taken into account. Note, that it also depends on the code system, and what properties it exposes for a terminology server to use.*
- *`valueSetBased` - The expansion of the ValueSet will be based on `ValueSet.grouping`. If `ValueSet.grouping` has been used, the hierarchy specified there will be applied to the expansion. Within `ValueSet.grouping` abstract codes from the underlying CodeSystem can be used as groups. If `ValueSet.grouping` does not exist the expansion will result in a flat list.*

For the scenario, where a CodeSystem's hierarchy and a ValueSet's hierarchy are mixed (e.g. `codeSystemAndValueSetBased`), it was not easy to find a clear and concise description. In the discussions we came to the conclusion that in case of the requirement of a mixed hierarchy, the hierarchy sould be fully specified within the ValueSet and as a result `valueSetBased` should be used. Thus, there would be one single point of truth defining the hierarchy.

IMPORTANT: The description of `valueSetBased` relies on the fact that the current extensions `valueset-expand-group` and `valueset-expand-rules` have been moved to FHIR Core. See subsequent sections.

4.2. Deprecation Of `excludeNested` And `excludeNotForUI`

With the introduction of the new parameter `hierarchyMode` the two parameters `excludeNested` and `excludeNotForUI` should be deprecated.

4.3. Move `valueset-expand-group` To FHIR Core

The elementes from this extension should be moved to FHIR Core in order to have all hierarchy related control items in FHIR Core.

The extension could be included into ValueSet's definition at `ValueSet` after `compose` and before `expansion`. To enable CodeSystem independent groupings, following structure is proposed:

- `grouping [0..1] BackboneElement`
  - `contains [1..*] see contains`

The example value set hierarchical example would have to be adapted accordingly.

The `expandGroupingRule` input parameter for the `$expand` operation determines the effect of the concepts in the `grouping`.

4.3.1. If Move To FHIR Core Is Not Possible

If for any reason the movement of the extenstion to FHIR Core is not possible, at least the description of the extension should be updated:

*Current **Comment***: Note that there are inter-relationships between concept status/properties and the way the groups are built; these are described and documented in the (value set hierarchical example)[d.html]. Note that this extension should be ignored when the expansion is not generated for UI.

*Updated **Comment***: Note that there are inter-relationships between concept status/properties and the way the groups are built; these are described and documented in the (value set hierarchical example) [https://www.hl7.org/fhir/valueset-example-hierarchical.html]. Note that this extension SHOULD be ignored when the `$expand` input parameter `hierarchyMode` is `flat`.

4.4. Move `valueset-expand-rules` To FHIR Core

The one element from this extension should be moved to FHIR Core in order to have all hierarchy related control items in FHIR Core.

This one element from the extension could be included into ValueSet's definition as an `$expand` input parameter:

- `expandGroupingRule [0..1] code`

The example value set hierarchical example would have to be adapted accordingly.

4.4.1. If Move To FHIR Core Is Not Possible

If for any reason the movement of the extenstion to FHIR Core is not possible, at least the description of the extension should be updated:

Current description: Defines how concepts are processed into the expansion when it's for UI presentation.

Updated description: Defines how concepts are processed when the `$expand` input parameter `hierarchyMode` is set to `codeSystemBased` or to `valueSetBased`. Note that this extension SHOULD be ignored when the `$expand` input parameter `hierarchyMode` is set to `flat`.

5. Related Jira-Issues

- https://jira.hl7.org/browse/FHIR-30700
- https://jira.hl7.org/browse/FHIR-30702
- https://jira.hl7.org/browse/FHIR-30703