NLQTI: Dutch Profile for Question and Test Interoperability

Items

Version: V1.0 (November 2012)

Table of Contents

Colophon

Document history

Versie	Datum	Omschrijving
V0.9	7 december 2011	Eerste volledige en door de reviewgroep goedgekeurde versie, plateau 1
V0.91	April 2011	Uitwerking voor plateau 2
V0.92	Mei 2012	Kleine tekstuele bijstellingen n.a.v. expert groep bijeenkomst
V0.93	Sep 2012	Translated to English
V1.0	November 2012	Small textual changes and typos. Abbreviation changed to NLQTI. Colophon added. Final version.

1   Introduction

This document is part of the technical documentation set for the Dutch Profile for Question and Test Interoperability, a.k.a. NLQTI. The full set of documents is listed in [NLQTI-ICS] (see sect. 1.1/pg. 4). Their goal is to supply content and software developers with enough information to implement this profile.
A common introduction and general information can be found in the document *"NLQTI: Dutch Profile for Question and Test Interoperability – Introduction and Common Sections"*.

This part of the profile describes:

- QTI Items. Technically these are the QTI documents with root element `<assessmentItem>`
- The interaction types allowed and their limitations
- The response and feedback processing for items

1.1   Additional documents

1.1.1   Profile documents

The Dutch Profile for Question and Test Interoperability consists of the following documents:

[NLQTI-AB]   NLQTI: Dutch Profile for Question and Test Interoperability - Algemene beschrijving toepassingsprofiel op basis van IMS QTI v2.1
*Functional description of the profile, **Dutch only**.*

[NLQTI-ICS]   NLQTI: Dutch Profile for Question and Test Interoperability – Introduction and common sections
Common parts of the profile.

[NLQTI-ITEM]   NLQTI: Dutch Profile for Question and Test Interoperability - Items
Description of items within this profile.

[NLQTI-TEST]   Dutch Profile for Question and Test Interoperability - Tests
Description of tests within this profile

[NLQTI-CP]   NLQTI: Dutch Profile for Question and Test Interoperability - Content Packaging
Describes the way items and tests should be packaged according to this profile.

1.1.2   Other documents and sources of information

A list with further additional documents and other sources of information can be found in [NLQTI-ICS].

2 General rules

2.1 Contents of an item

- This profile allows an item to contain a single or no interaction type only. In other words: An item is only allowed to contains one single question or no question at all (information without interaction)
- The only exception to this rule is the `<mediaInteraction>`. Every item is allowed to contain, besides a question/interaction type, zero or more `<mediaInteraction>`'s.
- There are two different situations to consider:
  - Interaction types for which the interaction is fully contained within a single XML element container. An item with such an interaction type must contain only one of these container elements).
    Example: The interaction type `choiceInteraction` is fully contained within the element `<choiceInteraction>`. An item of this type is allowed to have a single `<choiceInteraction>` element in its body only and no other kinds of interaction (again with the exception of optional `<mediaInteraction>`'s).
  - Interaction types for which there is no single surrounding element are allowed to occur multiple times but not in combination with other interaction types (with the exception of the aforementioned `<mediaInteraction>`'s.
    Example: An `inlineChoice` interaction type item contains multiple `<inlineChoiceInteraction>` elements in its body. No other interaction type elements are allowed (again with the exception of optional `<mediaInteraction>`'s).
- Item templates are not allowed within this profile.

2.2 Item lifecycle support

QTI items can have a complicated lifecycle (see [QTI-INFO], sect. 4.1). This profile supports those parts of the lifecycle meant for the learner (QTI terminology: "candidate") only.
As a result of this, the `view` attribute (like on `<rubricBlock>` elements) can have the value `view="candidate"` only.

2.3 Stylesheet usage

A QTI item in general allows multiple stylesheets of different types (using the `<stylesheet>` element underneath the root element, see sect. 3/pg. 6). This profile however allows CSS (version 3) stylesheets (MIME type `text/css`) only. For example:

```
<stylesheet href="css/ItemStyles.css" type="text/css"/>
```

Attributes on `<stylesheet>`:

Name	Prf?	M	Type	Remarks
href	Yes	1	URI	See [NLQTI-ICS], sect. 4.4
type	Lim	1	string	`text/css` always
media	No			
title	No			

Stylesheet usage is closely related to graphical design and layout. Please refer to [NLQTI-ICS], sect. 5 for guidelines about design, layout and stylesheet usage.

3 QTI items

3.1   Anatomy of an item

The basic building block for QTI is the *item* (or *assessment item*). It contains content (explanation, introduction, etc.) and usually one or more questions. It also describes when the answer is correct and what to do with the result. As described in sect. 2.1/pg. 5, this profile limits this to one question per item.
The basic anatomy of an item is as follows:

```
<assessmentItem xmlns="http://www.imsglobal.org/xsd/imsqti_v2p1" ...>

  <responseDeclaration  ...>
    <!-- Information about the answer -->
  </responseDeclaration>

  <outcomeDeclaration ...>
    <!-- Variables with information about the outcome (score, etc.) -->
  </outcomeDeclaration>

  <itemBody ...>
    <!-- The question itself -->
  </itemBody>

  <responseProcessing ...>
    <!-- What to do with the result -->
  </responseProcessing>

  <modalFeedback ...>
    <!-- Optional feedback to the learner -->
  </modalFeedback>

</assessmentItem>
```

- The `<responseDeclaration>` element contains information about how to interpret the response to the question. When is the answer correct and the resulting score.
- An `<outcomeDeclaration>` element declares a variable, for instance for passing the score to the environment or for guiding the feedback. More about this in sect. 5.2.2/pg. 24
- The element contains the fixed content and the interaction for the item. Its content looks like XHTML (strictly speaking it isn't XHTML: it's in another namespace and rendering using a browser is not compulsory)
- The `<responseProcessing>` element defines the actions to perform when a learner has answered the item's interaction. Its content is a kind of programming language. More about this in sect. 5.2.3/pg. 25.
- The `<modalFeedback>` elements define the feedback to the learner (what to show when the answer is right/wrong). More about this in sect. 5.2.4/pg. 28

3.2   Items without interaction

This profile explicitly allows items without interaction. In other words: items in which none of the interaction types allowed (sect. 4/pg. 8) are used.
The following additional rules apply for items without interaction:

- The item is not allowed to contain `<responseDeclaration>`, `<outcomeDeclaration>`, `<responseProcessing>` or `<modalFeedback>` elements
- It contains an `<itemBody>` element only, without any interaction type as defined in sect. 4/pg. 8.

3.3   Technical implementation

A QTI item is a single XML document. The root element is always `<assessmentItem>`.

Attributes on `<assessmentItem>`:

Name	Prf?	M	Type	Remarks
identifier	Yes	1	string	Preferably, an identifier should adhere to the rules stated in [KN-PID]
title	Yes	1	string	
adaptive	Lim	1	boolean	Limited to `false` only (non-adaptive items)
timeDependent	Lim	1	boolean	Limited to `false` only (non-time-dependent items)
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
toolName	Yes	?	string256	
toolVersion	Yes	?	string256	

Child elements of `<assessmentItem>`:

Name	Prf?	M	Remarks
responseDeclaration	Lim	*	See sect. 5.2.1/pg. 23
outcomeDeclaration	Lim	*	See sect. 5.2.2/pg. 24
templateDeclaration	No		
templateProcessing	No		
stylesheet	Lim	*	Referencing to CSS stylesheets only. See sect. sect. 2.3/pg. 5.
itemBody	Lim	1	The item body contains the content and the interaction (sect. 4/pg. 8). Design and layout is handled in [NLQTI-ICS], sect. 5.
responseProcessing	Lim	?	See sect. 5.2.3/pg. 25
modalFeedback	Lim	*	See sect. 5.2.4/pg. 28

4 Interaction types

The contents of the `<itemBody>` element determines the interaction type of the item.
As described in sect. 2.1/pg. 5, only one (or none) interaction type is allowed inside a single item.
In other words: Mingling of interaction types is not allowed.

The interaction types are separated into three categories:
- Singular interaction types, for which everything is contained in a single element, like `<choiceInteraction>`: sect. 4.1/pg. 8
- Plural interaction types, which always have more than one element within the item's body, like `<inlineChoiceInteraction>`: sect. 4.2/pg. 19
- Disallowed interaction types: sect. 4.3/pg. 21

The main reason for separating the singular and plural interaction types is the response and feedback processing. This is much more difficult for the plural types. See sect. 5/pg. 22.

4.1 Singular interaction types

Singular interaction types are contained in a single element. Only one singular interaction type element is allowed inside an item's `<itemBody>` element.

4.1.1 Interaction type element *<associateInteraction>*

Functional description:
A QTI `<associateInteraction>` allows associating elements from a set with each other. For example:
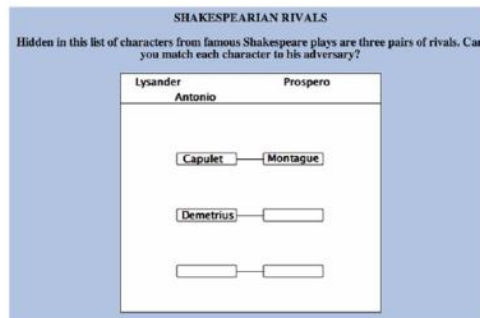
Profile functional limitations:
- When using randomization of element order: It is not possible to give an element a fixed position.
- It is not possible to set the minimum number of elements to associate

Profile technical implementation:
- A QTI item body is allowed a single `<associateInteraction>` element only
- Attributes of `<associateInteraction>`:

Name	Prf?	M	Type	Remarks
xml:base	No			
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
responseIdentifier	Lim	1	identifier	Fixed value: "RESPONSE"
shuffle	Yes	1	boolean	
maxAssociations	Lim	1		Maximum number of associations to make
minAssociations	No			

- Child elements of `<associateInteraction>`:

Name	Prf?	M	Remarks
prompt	Yes	?	
simpleAssociableChoice	Yes	+	

- Attributes of `<simpleAssociableChoice>`:

Name	Prf?	M	Type	Remarks
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
identifier	Yes	1	identifier	
fixed	No			
templateIdentifier	No			
showHide	No			
matchMax	Lim	1	integer	Fixed value: "1"
matchMin	Lim	1	integer	Fixed value: "1"

4.1.2 Interaction element **<choiceInteraction>**

Functional description:

A QTI `<choiceInteraction>` is what is usually called a multiple choice question
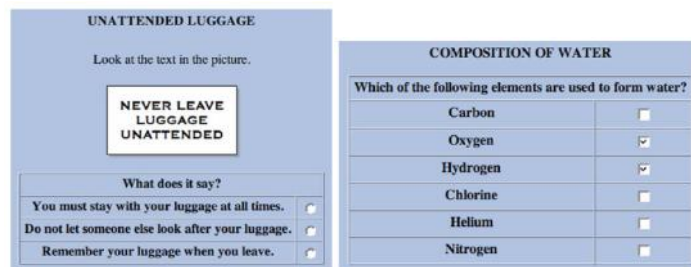
Profile functional limitations:

- The profile only allows Multiple Choice Single Answer (MCSA) or Multiple Choice Multiple Answer (MCMA) without additional limitations (no restrictions for the minimum/maximum number of answers to be ticked)
- When using randomization of answer order: It is not possible to give an answer a fixed position.

Profile technical implementation:

- A QTI item body is allowed a single `<choiceInteraction>` element only
- Attributes of `<choiceInteraction>`:

Name	Prf?	M	Type	Remarks
xml:base	No			
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
responseIdentifier	Lim	1	identifier	Fixed value: "RESPONSE"
shuffle	Yes	1	boolean	
maxChoices	Lim	1	integer	Only two values allowed: 0 For questions with (optional) multiple answers (Multiple Choice, Multiple Answer, MCMA) 1 For questions with a single answer (Multiple Choice, Single Answer, MCSA)
minChoices	No			

- Child elements of `<choiceInteraction>`:

Name	Prf?	M	Remarks
prompt	Yes	?	
simpleChoice	Yes	+	For attributes see below

- Attributes of `<simpleChoice>`:

Name	Prf?	M	Type	Remarks
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
identifier	Yes	1	identifier	
fixed	No			
templateIdentifier	No			
showHide	No			

4.1.3 Interaction element *<extendedTextInteraction>*

Functional description:
A QTI `<extendedTextInteraction>` allows providing an answer consisting of a block of text:

Profile functional limitations:
- Only a single answer is allowed
- Only plain text input (text without markup) is allowed
- Text for speech synthesis is not supported (this is the text that is read when the full question is read out loud (e.g. "fill in the text here").

Profile technical implementation:
- A QTI item body is allowed a single `<extendedTextInteraction>` element only
- Attributes of `<extendedTextInteraction>`:

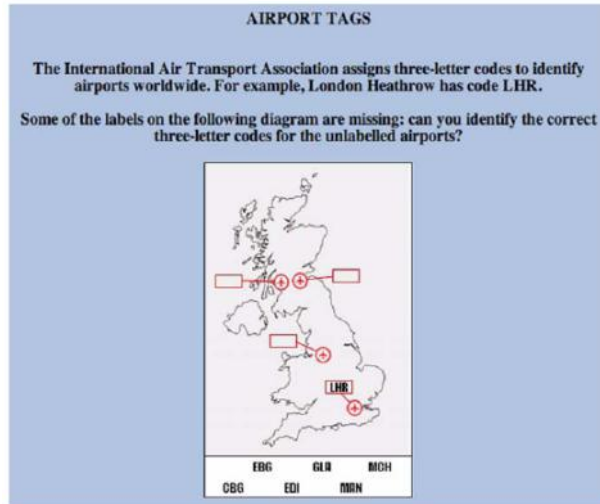| Name	Prf?	M	Type	Remarks
xml:base	No			
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
responseIdentifier	Lim	1	identifier	Fixed value: "RESPONSE"
base	No			
stringIdentifier	No			
expectedLength	Yes	?	integer	Expressed as the number of characters. This is a hint, not a rule.
patternMask	No			
placeholderText	No			
maxStrings	No			
minStrings	No			
expectedLines	Yes	?	integer	Expressed as the number of lines. This is a hint, not a rule.
format	Lim	?	string	Omit or use the fixed value "plain"

- Child elements of `<extendedTextInteraction>`:

Name	Prf?	M	Remarks
prompt	Yes	?	

4.1.4 Interaction type element *<graphicGapMatchInteraction>*

Functional description:

A QTI `<graphicGapMatchInteraction>` allows the learner to locate images on a bigger main image (usually implemented with drag-and-drop). This is the graphical variant of the `<gapMatchInteraction>` interaction type. In the example below the "airport tags" are images and not text!

Profile functional limitations:

- It is not allowed to locate an image more than once

Profile technical implementation:

- A QTI item body is allowed a single `<graphicGapMatchInteraction>` element only
- Attributes of `<graphicGapMatchInteraction>`:

Name	Prf?	M	Type	Remarks
xml:base	No			
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
responseIdentifier	Lim	1	identifier	Fixed value: "RESPONSE"

- Child elements of `<graphicGapMatchInteraction>`:

Name	Prf?	M	Remarks
prompt	Yes	?	
object	Yes	1	The main image to show. See [NLQTI-ICS], sect. 6
gapImg	Yes	+	The images for locating
associableHotSpot	Yes	+	The locations in the main image where the smaller images should be located.

- Attributes of `<gapImg>`:

Name	Prf?	M	Type	Remarks
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
identifier	Yes	1	identifier	
fixed	No			
templateIdentifier	No			
showHide	No			
matchGroup	No			
matchMax	Yes	1	integer	Fixed value "1" (can only be used once)
matchMin	Lim	?	integer	Fixed value "1" (must be used) of "0" (may be used). Default is "0".
objectLabel	Yes	?	string	

- Child elements of `<gapImg>`:

Name	Prf?	M	Remarks
object	Yes	1	See [NLQTI-ICS], sect. 6

- Attributes of `<associableHotspot>`:

Name	Prf?	M	Type	Remarks
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
identifier	Yes	1	identifier	
fixed	No			
templateIdentifier	No			
showHide	No			
matchGroup	No			
shape	Yes	1	shape	
coords	Yes	1	coordinates	
hotspotLabel	Yes	?	string256	
matchMax	Yes	1	integer	Fixed value "1" (must be used only once)
matchMin	Lim	?	integer	Fixed value "1" (must be used) of "0" (may be used). Default is "0".

4.1.5 Interaction type element *<gapMatchInteraction>*

Functional description:

A QTI allows to learner to fill holes in a text from a fixed set of elements/answers.
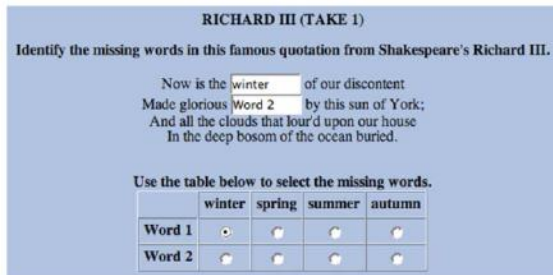
Profile functional limitations:

- When using randomization of element/answer order: It is not possible to give an element/answer a fixed position.
- Only textual elements/answers are allowed

Profile technical implementation:

- A QTI item body is allowed a single `<gapMatchInteraction>` element only
- An element/answer may only be used once
- Attributes of `<gapMatchInteraction>`:

Name	Prf?	M	Type	Remarks
xml:base	No			
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
responseIdentifier	Lim	1	identifier	Fixed value: "RESPONSE"
shuffle	Yes	1	boolean	

- Child elements of `<gapMatchInteraction>`:

Name	Prf?	M	Remarks
prompt	Yes	?	
gapText	Yes	+	See below for attributes
gapImg	No		
Text with optional markup			Main text. Holes are marked with `<gap>` elements. For attributes of `<gap>`, see below.

- Attributes of `<gapText>`:

Name	Prf?	M	Type	Remarks
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
identifier	Yes	1	identifier	
fixed	No			
templateIdentifier	No			
showHide	No			
matchGroup	No			
matchMax	Lim	1	integer	Fixed value "1" (must be used only once)
matchMin	Lim	?	integer	Fixed value "1" (must be used) of "0" (may be used). Default is "0".

- Attributes of `<gap>`:

Name	Prf?	M	Type	Remarks
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
identifier	Yes	1	identifier	
fixed	No			
templateIdentifier	No			
showHide	No			
matchGroup	No			
required	Yes	?	boolean	Default value is "false"

4.1.6   Interaction type element *<hotspotInteraction>*

Functional description:
A QTI `<hotspotInteraction>` allows the learner to point out one or more marked points

Profile functional limitations:
- For the number of points to point out: You can only set a maximum, not a minimum.

Profile technical implementation:

- A QTI item body is allowed a single `<hotspotInteraction>` element only
- Attributes of `<hotspotInteraction>`:

Name	Prf?	M	Type	Remarks
`xml:base`	No			
`id`	Yes	?	identifier	
`class`	Yes	?	identifier	
`xml:lang`	Yes	?	language	
`label`	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
`responseIdentifier`	Lim	1	identifier	Fixed value: "RESPONSE"
`maxChoices`	Yes	1	integer	
`minChoices`	No			

- Child elements of `<hotspotInteraction>`:

Name	Prf?	M	Remarks
`prompt`	Yes	?	
`object`	Yes	1	Main image. See [NLQTI-ICS], sect. 6
`hotspotChoice`	Yes	+	The points to choose from in the main image. For attributes see below.

- Attributes of `<hotspotChoice>`:

Name	Prf?	M	Type	Remarks
`id`	Yes	?	identifier	
`class`	Yes	?	identifier	
`xml:lang`	Yes	?	language	
`label`	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
`identifier`	Yes	1	identifier	
`fixed`	No			
`templateIdentifier`	No			
`showHide`	No			
`shape`	Yes	1	shape	
`coords`	Yes	1	coordinates	
`hotspotLabel`	Yes	?	string256	

4.1.7   Interaction type element *<hottextInteraction>*

Functional description:

A QTI allows the learner to point out one or more locations in a text, for instance the locations of errors.

Profile functional limitations:

- The functionality is limited to choosing just a single location or multiple locations (and not a fixed number).

Profile technical implementation:

- A QTI item body is allowed a single `<hottextInteraction>` element only
- Attributes of `<hottextInteraction>`:

Name	Prf?	M	Type	Remarks
`xml:base`	No			
`id`	Yes	?	identifier	
`class`	Yes	?	identifier	
`xml:lang`	Yes	?	language	
`label`	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
`responseIdentifier`	Lim	1	identifier	Fixed value: "RESPONSE"
`maxChoices`	Lim	1	integer	Only two values allowed: 0 Questions with multiple answers 1 Questions with a single answer
`minChoices`	No			

- Child elements of `<hottextInteraction>`:

Name	Prf?	M	Remarks
`prompt`	Yes	?	
Text with optional markup			Text containing the choices to make

- The text of a `<hottextInteraction>` must contain `<hottext>` elements
- Attributes of `<hottext>`:

Name	Prf?	M	Type	Remarks
`id`	Yes	?	identifier	
`class`	Yes	?	identifier	
`xml:lang`	Yes	?	language	
`label`	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
`identifier`	Yes	1	identifier	
`fixed`	No			
`templateIdentifier`	No			
`showHide`	No			

4.1.8   Interaction type element *<matchInteraction>*

Functional description:

A QTI `<matchInteraction>` allows the learner to associate elements from two different sets with each other. Also known as a "matrix question". For example:

Profile functional limitations:

- When using randomization of element/answer order: It is not possible to give an element/answer a fixed position.
- You can only specify that using an element/answer is mandatory or optional (not that it must be used a minimum number of times)

Profile technical implementation:

- A QTI item body is allowed a single `<matchInteraction>` element only
- Attributes of `<matchInteraction>`:

Name	Prf?	M	Type	Remarks
`xml:base`	No			
`id`	Yes	?	identifier	
`class`	Yes	?	identifier	
`xml:lang`	Yes	?	language	
`label`	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
`responseIdentifier`	Lim	1	identifier	Fixed value: "RESPONSE"
`shuffle`	Yes	1	boolean	
`maxAssociations`	Lim	1		Fixed value: "0" (unlimited)
`minAssociations`	No			

- Child elements of `<matchInteraction>`:

Name	Prf?	M	Remarks
`prompt`	Yes	?	
`simpleMatchSet`	Yes	2	Always appears twice. No attributes.

- Child elements of `<simpleMatchSet>`:

Name	Prf?	M	Remarks
`simpleAssociableChoice`	Yes	*	For attributes see below

- Attributes of `<simpleAssociableChoice>`:

Name	Prf?	M	Type	Remarks
`id`	Yes	?	identifier	
`class`	Yes	?	identifier	
`xml:lang`	Yes	?	language	
`label`	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
`identifier`	Yes	1	identifier	
`fixed`	No			
`templateIdentifier`	No			
`showHide`	No			
`matchGroup`	No			
`matchMax`	Yes	1	integer	We allow that an element/answer can have multiple associations
`matchMin`	Lim	?	integer	Only two values allowed: 0 (default) Answer does not need to be associated 1 Answer must be associated

4.1.9   Interaction type element **<orderInteraction>**

Functional description:

A QTI `<orderInteraction>` allows the learner to order elements/answers. For instance:
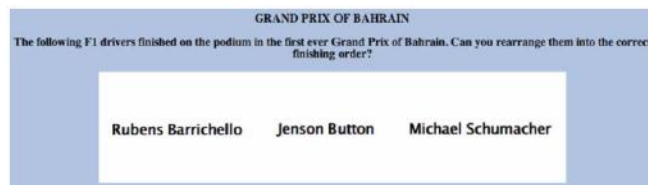
Profile functional limitations:

- When using randomization of element/answer order: It is not possible to give an element/answer a fixed position.
- You cannot specify a minimum/maximum to the number of elements/answers to order. In other words: The learner must order every element/answer.

Profile technical implementation:

- A QTI item body is allowed a single `<orderInteraction>` element only
- Attributes of `<orderInteraction>`:

Name	Prf?	M	Type	Remarks
`xml:base`	No			
`id`	Yes	?	identifier	
`class`	Yes	?	identifier	
`xml:lang`	Yes	?	language	
`label`	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
`responseIdentifier`	Lim	1	identifier	Fixed value: "RESPONSE"
`shuffle`	Yes	1	boolean	
`minChoices`	No			
`maxChoices`	No			
`orientation`	Yes	?	orientation	This is a hint, not a requirement (values: "horizontal" (default) or "vertical").

- Child elements of `<orderInteraction>`:

Name	Prf?	M	Remarks
`prompt`	Yes	?	
`simpleChoice`	Yes	+	For attributes see below

- Attributes of `<simpleChoice>`:

Name	Prf?	M	Type	Remarks
`id`	Yes	?	identifier	
`class`	Yes	?	identifier	
`xml:lang`	Yes	?	language	
`label`	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
`identifier`	Yes	1	identifier	
`fixed`	No			
`templateIdentifier`	No			
`showHide`	No			

4.1.10 Interaction type element *<selectPointInteraction>*

Functional description:

A QTI `<selectPointInteraction>` allows the learner to select one or more unmarked spots on a main graphic. For instance:
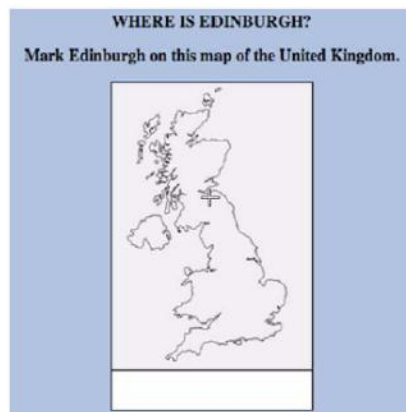
Profile functional limitations:

- It is only possible to specify a maximum for the number of spots to select, no minimum.

Profile technical implementation:

- A QTI item body is allowed a single `<selectPointInteraction>` element only
- Attributes of `<selectPointInteraction>`:

Name	Prf?	M	Type	Remarks
xml:base	No			
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
responseIdentifier	Lim	1	identifier	Fixed value: "RESPONSE"
maxChoices	Yes	1	integer	
minChoices	No			

- Child elements of `<selectPointInteraction>`:

| Name	Prf?	M	Remarks
prompt	Yes	?	
object	Yes	1	See [NLQTI-ICS], sect. 6

Remarks:

- Different from other interaction types, a `<selectPointInteraction>` element does not contain the correct answers directly. For a `<selectPointInteraction>` these are specified in the associated `<responseProcessing>`.

4.1.11 Interaction type element *<sliderInteraction>*

Functional description:
A QTI allows the learner to choose a number on a given scale, for instance a percentage. Usually implemented as a slider. For instance:
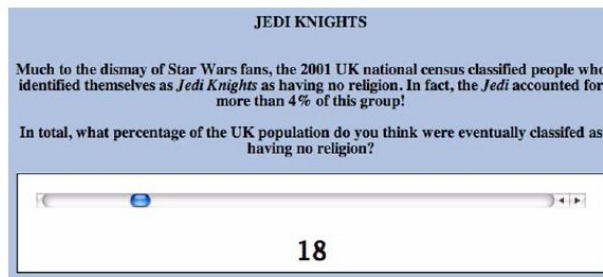
Profile functional limitations:

- Showing a scale division cannot be specified
- Er kan geen schaalverdeling op de slider getoond worden
- The slider's orientation (low to high or vice versa) or its horizontal/vertical orientation cannot be specified

Profile technical implementation:

- A QTI item body is allowed a single `<sliderInteraction>` element only
- Attributes of `<sliderInteraction>`:

Name	Prf?	M	Type	Remarks
xml:base	No			
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
responseIdentifier	Lim	1	identifier	Fixed value: "RESPONSE"
lowerBound	Yes	1	float	
upperBound	Yes	1	float	
step	Yes	?	integer	
stepLabel	No			Implementation left to the rendering engine
orientation	No			Custom: Left to right
reverse	No			Custom: Low to high

- Child elements of `<sliderInteraction>`:

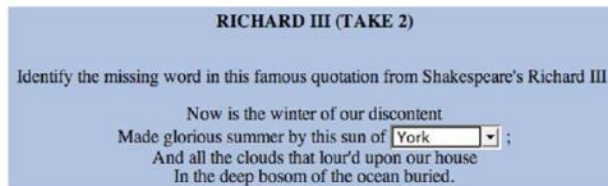| Name	Prf?	M	Remarks
prompt	Yes	?	

4.2 Plural interaction types

These interaction types are allowed to appear more than once in an item.

4.2.1 Interaction type element *<inlineChoiceInteraction>*

Functional description:
A allows the learner to fill holes in a text from fixed sets of answers. For instance:

Profile functional limitations:
- When using randomization of answer order: It is not possible to give an answer a fixed position.
- Specifying an answer (for every "hole") is mandatory

Profile technical implementation:
- A QTI item body is allowed to have more than one `<inlineChoiceInteraction>` element.
- Attributes of `<inlineChoiceInteraction>`:

Name	Prf?	M	Type	Remarks
xml:base	No			
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
responseIdentifier	Lim	1	identifier	Always starts with "RESPONSE_"
shuffle	Yes	1	boolean	
required	Lim	1	boolean	Always "true" (answer is mandatory)

- Child elements of `<inlineChoiceInteraction>`:

Name	Prf?	M	Remarks
inlineChoice	Yes	+	

- Attributes of `<inlineChoice>`:

Name	Prf?	M	Type	Remarks
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
identifier	Yes	1	identifier	
fixed	No			

4.2.2 Interaction type element *<mediaInteraction>*

Functional description:
A QTI `<mediaInteraction>` shows a media object, for instance, a movie or a flash animation.
It is possible to implement this with an item without interaction (see sect. 3.2/pg. 6) also.
However, a `<mediaInteraction>` provides you with additional functionality like limiting the number of views.
A `<mediaInteraction>` is the only interaction type can (but does not need to) be combined with other interaction types

Profile functional limitations:
- You cannot specify that the media object is played in a loop

Profile technical implementation:

- Any QTI item body is allowed to contain zero or more `<mediaInteraction>` elements beside another interaction type

- Attributes of `<mediaInteraction>`:

Name	Prf?	M	Type	Remarks
xml:base	No			
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
responseIdentifier	Lim	1	identifier	Always starts with "MEDIA_"
autostart	Yes	1	boolean	
minPlays	Yes	?	integer	
maxPlays	Yes	?	integer	
loop	No			

- Child elements of `<mediaInteraction>`:

Name	Prf?	M	Remarks
prompt	Yes	?	
object	Yes	1	Reference to the media object, see [NLQTI-ICS], sect. 6

4.2.3   Interaction type element **<positionObjectInteraction>**

Functional description:
A QTI allows the learner to position one or more graphics on the right unmarked spots on a main graphic (usually implemented with drag-and-drop). For example:
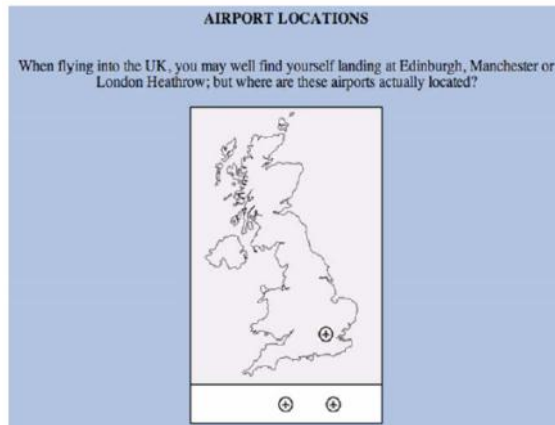
- Remark: Because of the implementation of the response processing, every graphic must located at a specific location on the map. So the learner must be able to distinguish the different graphics. This is not the case in the example above!

Profile functional limitations:

- None

Profile technical implementation:

- An interaction type element `<positionObjectInteraction>` is always surrounded by the `<positionObjectStage>` container element. A QTI item body is only allowed a single `<positionObjectStage>` element.

- Child elements of `<positionObjectStage>`:

Name	Prf?	M	Remarks
object	Yes	1	Main graphic. See [NLQTI-ICS], sect. 6
positionObjectInteraction	Yes	+	Graphics to locate

- Attributes of `<positionObjectInteraction>`:

Name	Prf?	M	Type	Remarks
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
identifier	Yes	1	identifier	
fixed	No			
responseIdentifier	Lim	1	identifier	Always starts with "RESPONSE_"
showHide	No			
centerPoint	Yes	?	coordinate	
maxChoices	Lim	1	integer	Fixed value: "1"
minChoices	No			

- Child elements of `<positionObjectInteraction>`:

Name	Prf?	M	Remarks
object	Yes	1	The graphics to locate. See [NLQTI-ICS], sect. 6

4.2.4 Interaction type element *<textEntryInteraction>*

Functional description:

A QTI `<textEntryInteraction>` allows the learner to fill gaps in a text with free text. For instance:
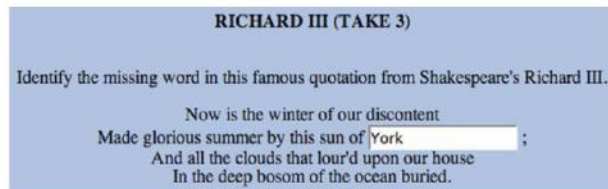
Profile functional limitations:

- Numeric entries (comparing the answers based on numeric instead of textual comparison) is not part of this profile.
- Text for speech synthesis is not allowed

Profile technical implementation:

- A QTI item body is allowed to have more than one `<textEntryInteraction>` element.
- Attributes of `<textEntryInteraction>`:

Name	Prf?	M	Type	Remarks
xml:base	No			
id	Yes	?	identifier	
class	Yes	?	identifier	
xml:lang	Yes	?	language	
label	Yes	?	string256	See [NLQTI-ICS], sect. 4.3
responseIdentifier	Lim	1	identifier	Always starts with "RESPONSE_"
base	No			
stringIdentifier	No			
expectedLength	Yes	?	integer	This is a hint, not a requirement
patternMask	Yes	?	string	Regular expression as defined in the W3C XML Schema standard. See [XML-SDT]
placeholderText	No			

4.3 Disallowed interaction types

The profile forbids the use of the following interaction types:

- `<customInteraction>`
- `<drawingInteraction>`
- `<graphicAssociateInteraction>`
- `<graphicOrderInteraction>`
- `<uploadInteraction>`

5 Response en feedback processing

This profile limits the response and feedback processing to:
- Computing a item's score
- Controlling the feedback to the learner

5.1 Functional limitations

5.1.1 Score

Remark: Scoring of items is closely linked to the subject of the *weight* of a score: the contribution of the score of a single item to the overall score. However, the weight of a score is not defined at item level but at test level. After all, the same question can be weighted differently in different tests. See [NLQTI-TEST] for more information.

This profile defines the following rules for the scoring of an item:
- The score of an item is always a floating point number in between 0.0 (wrong/bad) and 1.0 (correct/very good).
- With respect to scoring there are two types of items:
  - Items that are scored correct (1.0) or incorrect (0.0) (and nothing in between)
  - Items that compute the score. The result must be in between 0.0 and 1.0
- The only means allowed for computing a score are the `<mapping>` or `<areaMapping>` elements within a `<responseDeclaration>`. This means you can attach a sub-score (positive or negative) to different answers given. The sum of the sub-scores is the final score.
- Explicitly computing a score by calculating it in the `<responseProcessing>` is not allowed within this profile.
- Passing the score to the environment is done using an `<outcomeDeclaration>` with the fixed name SCORE (sect. 5.2.2/pg. 24).

5.1.2 Feedback

This profile limits feedback to the learner to no feedback or feedback based only on whether the answer is correct or incorrect. It is not allowed to add feedback based on more complex criteria like for instance the number of times the learner tried to answer the question or the time passed. Most important reasons for this limitation:
- This profile limits itself to testing (in contrast to exercising). Feedback plays a much less prominent role in testing then it does in exercising.
- Allowing more complex feedback would have caused a disproportional increase in complexity of the profile (causing more complex implementations of authoring systems, rendering engines, etc.). Allowing QTI items to change their behavior based on criteria like the number tries or the time passed would mean allowing complex and non-standard `<responseProcessing>`, something we're trying to avoid in this profile.

So this profile limits feedback to:
- No feedback: There are no feedback elements in the item
- Singular feedback:
  - There is feedback present for "wrong answer given"
  - There is optional feedback present for "correct answer given"
- More detailed feedback, based on other criteria or based on the actual answer given, is not allowed.

5.2 Technical implementation

The definition of response and feedback processing is scattered over a number of different sections/elements within an item:
- The `<responseDeclaration>` declarations
- The `<outcomeDeclaration>` declarations
- The `<responseProcessing>` section
- The `<modalFeedback>` definitions

5.2.1 *<responseDeclaration>* elements

5.2.1.1 General

- A `<responseDeclaration>` determines the system's reaction to the response for an item: Is the answer given correct or incorrect, what's the score, etc. Interactions in the item's body are always bound to a specific `<responseDeclaration>`. Here is an example:

```
<responseDeclaration identifier="RESPONSE" cardinality="single" baseType="identifier">
  <correctResponse>
    <value>C</value>
  </correctResponse>
</responseDeclaration>
```

- Attributes of `<responseDeclaration>`:

Name	Prf?	M	Type	Remarks
identifier	Yes	1	identifier	
cardinality	Yes	1	cardinality	Value depends on the interaction type. See table below.
baseType	Lim	1	type	Mandatory for this profile. Value depends on the interaction type. See table below.

- Child elements of `<responseDeclaration>`:

Name	Prf?	M	Remarks
defaultValue	No		
correctResponse	Lim	1	Mandatory for this profile. With the exception of the extendedTextInteraction and mediaInteraction interaction types. These types have a pro forma response declaration only without a defined correct answer.
mapping	Yes	?	Only one of both allowed. See sect. 5.2.1.2/pg. 24
areaMapping	Yes	?	

- There are always exactly the same `<responseDeclaration>` elements as interactions in the item's body. This means:
  - For singular interaction types (sect. 4.1/pg. 8), there is exactly one `<responseDeclaration>`. This is always called "RESPONSE".
  - For plural interaction types (sect. 4.2/pg. 19), the number of `<responseDeclaration>` elements is exactly the same as the number of interactions. The identifiers always start with "RESPONSE_", for example "RESPONSE_01", "RESPONSE_A", etc.
    Remark: When using the standard templates for plural interaction types (sect. 5.2.3.3/pg. 27), the identifiers must be called "RESPONSE_xx". xx is a two digit number in between 00 and 10.
  - For `<mediaInteraction>`'s, the identifiers of the corresponding (pro forma) `<responseDeclaration>` always start with "MEDIA_" (for example "MEDIA_01", "MEDIA_MOVIE3", etc.).

- A `<responseDeclaration>` has a data type (attribute baseType) and a cardinality (attribute cardinality). This depends on the interaction type:

Interaction type	baseType	cardinality	Remarks
associateInteraction	pair	multiple	
choiceInteraction	identifier	single (SA) multiple (MA)	
extendedTextInteraction	string	single	Pro forma declaration. It is not possible to automatically score this item type.
gapMatchInteraction	directedPair	multiple	
graphicGapMatchInteraction	directedPair	multiple	
hotspotInteraction	identifier	single	
hottextInteraction	identifier	single (SA) multiple (MA)	
inlineChoiceInteraction	identifier	single	
matchInteraction	directedPair	multiple	
mediaInteraction	string	single	Pro forma declaration. It is not possible to automatically score this item type.
orderInteraction	identifier	ordered	
positionObjectInteraction	point	single	

Interaction type	baseType	cardinality	Remarks
selectPointInteraction	point	single	
sliderInteraction	integer of float	single	
textEntryInteraction	string	single	

- `<responseDeclaration>`'s of type record are not allowed within this profile: The value "record" for the attribute `cardinality` is not allowed.

5.2.1.2   Scores and mapping

- If an item results in a computed score (instead of just a correct/incorrect indication), the mapping mechanism in the `<responseDeclaration>` must be used. This is an example of a `<responseDeclaration>` belonging to a `<choiceInteraction>` MCMA type question:

```
<responseDeclaration identifier="RESPONSE" cardinality="multiple"
                                             baseType="identifier">
  <correctResponse>
    <value>CORRECT1</value>
    <value>CORRECT2</value>
  </correctResponse>
  <mapping lowerBound="0.0" upperBound="1.0" defaultValue="0.0">
    <mapEntry mapKey="CORRECT1" mappedValue="0.5"/>
    <mapEntry mapKey="CORRECT2" mappedValue="0.5"/>
    <mapEntry mapKey="WRONGANSWER1" mappedValue="-0.5"/>
    <mapEntry mapKey="WRONGANSWER2" mappedValue="-0.5"/>
  </mapping>
</responseDeclaration>
```

Every correct answer yields ½ point, every incorrect answer subtracts ½ point. The mapping defines a lower and upper bound (attributes `lowerBound` and `upperBound`), keeping the score in between 0.0 and 1.0 as laid down by the profile.

- Graphical questions use an `<areaMapping>` instead of a `<mapping>` element.
- Plural interaction types (sect. 4.2/pg. 19) have plural `<responseDeclarations>`'s (one for every interaction), each one defining a score. The `<responseProcessing>` adds the separate scores and limits the result in between 0.0 and 1.0 (sect. 5.2.3.3/pg. 27)

5.2.2   *<outcomeDeclaration>* elements

5.2.2.1   General

- An `<outcomeDeclaration>` is an internal variable in a QTI item. For instance:

```
<outcomeDeclaration identifier="SCORE" cardinality="single" baseType="float"
   normalMinimum="0.0" normalMaximum=1.0"/>
```

- Different `<outcomeDeclaration>`'s than defined in the profile are not allowed.
- Attributes of `<outcomeDeclaration>`:

Name	Prf?	M	Type	Remarks
identifier	Lim	1	identifier	Three possible values: "SCORE", "FEEDBACK", "FEEDBACK_THRESHOLD"
cardinality	Lim	1	cardinality	Always "single"
baseType	Lim	1	type	For "SCORE", "FEEDBACK_THRESHOLD": "float" For "FEEDBACK": "identifier"
view	Lim	?	view	Don't use or use the fixed value "candidate"
interpretation	Yes	?	string	
longInterpretation	No			
normalMaximum	Lim	?	float	For SCORE always "1.0" Others: Don't use
normalMinimum	Lim	?	float	For SCORE always "0.0" Others: Don't use
masteryValue	No			

- Child elements of `<outcomeDeclaration>`:

Name	Prf?	M	Remarks
defaultValue	Lim	?	Only when specified below
matchTable	No		
interpolationTable	No		

5.2.2.2   Score

- QTI items with interaction must always define an `<outcomeDeclaration>` with the fixed name SCORE and of type `float`. The attributes `normalMimnimum` and `normalMaximum` must always be present (necessary for the correct processing of scores in tests):

```
<outcomeDeclaration identifier="SCORE" cardinality="single" baseType="float"
    normalMinimum="0.0" normalMaximum=1.0"/>
```

- Because it's a numeric variable, its default value is zero. In contrast to what happens in some of the QTI examples, you do not have to set this explicitly.
- The value of the SCORE `<outcomeDeclaration>` is always in between 0.0 (wrong/incorrect) and 1.0 (very good/correct).
- It's the task of the response processing to give the SCORE `<outcomeDeclaration>` the right value.

5.2.2.3   Feedback

- In case of feedback, there must be an `<outcomeDeclaration>` with the fixed name FEEDBACK of type `identifier`:

```
<outcomeDeclaration identifier="FEEDBACK" cardinality="single" baseType="identifier"/>
```

- The FEEDBACK `<outcomeDeclaration>` must take one of the following three values only:

Value	Meaning
(null)	No answer given (yet)
ANSWER_CORRECT	The answer given was correct
FAILURE	The answer given was incorrect

- It's the task of the response processing to give the FEEDBACK `<outcomeDeclaration>` the right value.

5.2.2.4   Feedback threshold

- In case of feedback and the answer is a score (and not just correct/incorrect), the response processing must, to display the right feedback, know when to consider the answer correct. This is done by defining a special `<outcomeDeclaration>` with the name FEEDBACK_THRESHOLD:

```
<outcomeDeclaration identifier="FEEDBACK_THRESHOLD" cardinality="single"
                                                    baseType="float">
  <defaultValue>
    <value>0.75</value>
  </defaultValue>
</outcomeDeclaration>
```

- The FEEDBACK_THRESHOLD `<outcomeDeclaration>` is always of type `float` and always has a default value in between 0.0 and 1.0.
- It is used by the response processing as follows:
  - If the final score is greater or equal to the defined threshold value, the answer is considered correct and the FEEDBACK `<outcomeDeclaration>` is set to ANSWER_CORRECT.
  - If the final score is less than the defined threshold value, the answer is considered incorrect and the FEEDBACK `<outcomeDeclaration>` is set to FAILURE.

5.2.3   *<responseProcessing>* element

5.2.3.1   General

- A `<responseProcessing>` section is responsible for dealing with the responses of the learner. Within this profile, the only things it is allowed to do are:
  - Set the SCORE `<outcomeDeclaration>` to the final score:
    - For a correct/incorrect situation: 0.0 (incorrect) or 1.0 (correct)
    - For a score: A value in between 0.0 (wrong/incorrect) and 1.0 (very good/correct).
  - In case of feedback: Set the FEEDBACK `<outcomeDeclaration>` to one of the fixed values defined in sect. 5.2.2/pg. 24.

- This profile limits response processing in such a way that engines can implement it without implementing the QTI "programming language". This is organized differently for singular (sect. 5.2.3.2/pg. 26) and plural (sect. 5.2.3.3/pg. 27) interaction types.

5.2.3.2   Response processing for singular interaction types

- This section is about response processing for singular interaction types (sect. 4.1/pg. 8) only.
- The profile dictates that for a singular interaction type the response processing is done using one of the predefined "response processing templates". A template contains standard response processing code and is identified using a fixed URI/identifier.
- Exceptions to this rule are `extendedTextInteraction` and `mediaInteraction`: These interaction types have no response processing.
- The profile defines two types of predefined templates:
  - The standard QTI templates. URI's for these templates start with `http://www.imsglobal.org/question/qti_v2p0/rptemplates/`
  - Templates defined by the profile. URI's for these templates start with `http://www.edustandaard.nl/nl-qti/1/rptemplates/`
- Here is an example of a reference to one of the templates:

```
<responseProcessing
    template="http://www.edustandaard.nl/nl-qti/1/rptemplates/RPTEMPLATE_GF"
    templateLocation="../rptemplates/RPTEMPLATE_GF.xml"/>
```

- A rendering engine, encountering a template reference for the response processing, can choose between:
  - If response processing is not fully supported, the engine can fall back to the behavior defined for the template, based on the template's URI (in the `template` attribute).
  - When response processing is supported, the engine can execute the response processing code in the template (the template can be found using the `templateLocation` attribute).
- This profile defines the following templates for singular interaction types:

Template URI	Meaning
`http://www.edustandaard.nl/nl-qti/1/rptemplates/RPTEMPLATE_GF`	Compare the answer given to the correct answer in the response declaration (always called `RESPONSE`). A correct answer yields a score of 1.0, an incorrect (or no) answer 0.0.
`http://www.edustandaard.nl/nl-qti/1/rptemplates/RPTEMPLATE_GF_FB1`	Compare the answer given to the correct answer in the response declaration (always called `RESPONSE`). A correct answer yields a score of 1.0, no or a wrong answer 0.0. A correct answer sets the `FEEDBACK` outcome declaration to `ANSWER_CORRECT`, an incorrect (or no) answer to `FAILURE`.
`http://www.edustandaard.nl/nl-qti/1/rptemplates/RPTEMPLATE_SCORE`	The score is computed using the mapping declarations in the response declaration (always called `RESPONSE`). No answer yields a score of 0.0.
`http://www.edustandaard.nl/nl-qti/1/rptemplates/RPTEMPLATE_SCORE_FB1`	The score is computed using the mapping declarations in the response declaration (always called `RESPONSE`). No answer yields a score of 0.0. If the score is greater than the default value of the outcome declaration `TRESHOLD_VALUE`, the `FEEDBACK` outcome declaration is set to `ANSWER_CORRECT`. If not it is set to `FAILURE`.
`http://www.edustandaard.nl/nl-qti/1/rptemplates/RPTEMPLATE_POINT_SCORE`	The score is computed using the point mapping declarations in the response declaration (always called `RESPONSE`). No answer yields a score of 0.0.
`http://www.edustandaard.nl/nl-qti/1/rptemplates/RPTEMPLATE_POINT_SCORE_FB1`	The score is computed using the point mapping declarations in the response declaration (always called `RESPONSE`). No answer yields a score of 0.0. If the score is greater than the default value of the outcome declaration `TRESHOLD_VALUE`, the `FEEDBACK` outcome declaration is set to `ANSWER_CORRECT`. If not it is set to `FAILURE`.
`http://www.imsglobal.org/question/qti_v2p0/rptemplates/Match_Correct.xml`	Standard QTI template, equivalent to `RPTEMPLATE_GF`
`http://www.imsglobal.org/question/qti_v2p0/rptemplates/Map_Response.xml`	Standard QTI template, equivalent to `RPTEMPLATE_SCORE`
`http://www.imsglobal.org/question/qti_v2p0/rptemplates/Map_Response_Point.xml`	Standard QTI template, equivalent to `RPTEMPLATE_POINT_SCORE`

- When delivering content according to this profile, the referenced templates must be present also. It is not allowed to assume the content's receiver has the templates available. This prevents items that follow the profile becoming unusable in more general QTI environments. See also [NLQTI-CP].

5.2.3.3 Response processing for plural interaction types

- This section is about response processing for plural interaction types (sect. 4.2/pg. 19) only.
- Since the number of interactions in an item is variable, fixed processing templates (like for singular interaction types) are not possible. Response processing always depends on the number of interactions. For instance, here is the response processing for an item with three interactions, resulting in a correct/incorrect score:

```
<responseProcessing>
  <responseCondition>
    <responseIf>
      <and>
        <match>
          <variable identifier="RESPONSE_01"/>
          <correct identifier="RESPONSE_01"/>
        </match>
        <match>
          <variable identifier="RESPONSE_02"/>
          <correct identifier="RESPONSE_02"/>
        </match>
        <match>
          <variable identifier="RESPONSE_03"/>
          <correct identifier="RESPONSE_03"/>
        </match>
      </and>
      <setOutcomeValue identifier="SCORE">
        <baseValue baseType="float">1.0</baseValue>
      </setOutcomeValue>
    </responseIf>
    <responseElse>
      <setOutcomeValue identifier="SCORE">
        <baseValue baseType="float">0.0</baseValue>
      </setOutcomeValue>
    </responseElse>
  </responseCondition>
</responseProcessing>
```

- Limitations are defined to allow for engines that do not implement full response processing.
- There are however a number of predefined templates for interactions with up to 10 interactions. You can use these templates but this is not mandatory. For interactions with more than 10 interactions, the result processing must defined otherwise.
- So a rendering engine, encountering a plural interaction type, can choose between:
  - If response processing is not fully supported and no standard template is used, the engine can fall back to standard behavior (see below).
  - If response processing is not fully supported and a standard template is used, the engine can fall back to the behavior defined for the template, based on the template's URI (in the `template` attribute).
  - When response processing is supported, the engine can execute the response processing code. This code must follow the rules as defined blow, however there is no way the profile can extort this.
- You can find examples of response processing within the profile's definition in the accompanying example code.

Rules for handling score:

- **For a correct/incorrect result:**
  - None of the `<responseDeclaration>`'s present have a `<mapping>` or `<areaMapping>` subelement
  - The result is considered correct if all interactions are correct (resulting in a 1.0 score). If one or more interactions are incorrect, the full item is incorrect (resulting in a 0.0 score).

- For a scored result:
  - All of the `<responseDeclaration>`'s present have a `<mapping>` or `<areaMapping>` sub-element
  - De total score is computed by adding the sub-scores of all `<mapping>` or `<areaMapping>` definitions.
    - If the result is negative, the score is limited to 0.0
    - If the result is greater than 1.0, the score is limited to 1.0

- Mixed situations (some `<responseDeclaration>`'s have `<mapping>` or `<areaMapping>` subelement) are not allowed

Regels for handling feedback:

- An item is considered to have feedback if there is an `<outcomeDeclaration>` called FEEDBACK.
- Er is sprake van een vraag met feedback als er een `<outcomeDeclaration>` aanwezig is met de naam/identifier FEEDBACK.
- For a correct/incorrect result:
  - A correct result sets this `<outcomeDeclaration>` to ANSWER_CORRECT
  - An incorrect result sets this `<outcomeDeclaration>` to FAILURE
- For a scored result:
  - There must be an `<outcomeDeclaration>` of type float called FEEDBACK_THRESHOLD with a defined default value.
  - If the total score is greater than or equal to the value of FEEDBACK_THRESHOLD: Set `<outcomeDeclaration>` FEEDBACK to ANSWER_CORRECT
  - If the total score is less than the value of FEEDBACK_THRESHOLD: Set `<outcomeDeclaration>` FEEDBACK to FAILURE

Optional usage of standard templates

- In the descriptions below, xx is always a two digit number in between 01 and 10
- If standard templates are used for plural interaction types, the response identifiers *must always be called* RESPONSE_xx
- This profile defines the following templates for plural interaction types:

Template URI	Meaning
http://www.edustandaard.nl/nl-qti/1/rptemplates/ RPTEMPLATE_GF_xx	Compare the answers given to the correct answers in the response declarations (always called RESPONSE_xx). All correct yields a score of 1.0, any incorrect (or no) answer 0.0.
http://www.edustandaard.nl/nl-qti/1/rptemplates/ RPTEMPLATE_GF_FB1_xx	Compare the answers given to the correct answers in the response declarations (always called RESPONSE_xx). All correct yields a score of 1.0, any incorrect (or no) answer 0.0. A correct answer sets the FEEDBACK outcome declaration to ANSWER_CORRECT, an incorrect (or no) answer to FAILURE.
http://www.edustandaard.nl/nl-qti/1/rptemplates/ RPTEMPLATE_SCORE_xx	The score is computed using the mapping declarations in the response declarations (always called RESPONSE_xx). No answer yields a score of 0.0.
http://www.edustandaard.nl/nl-qti/1/rptemplates/ RPTEMPLATE_SCORE_FB1_xx	The score is computed using the mapping declarations in the response declarations (always called RESPONSE_xx). No answer yields a score of 0.0. If the score is greater than the default value of the outcome declaration TRESHOLD_VALUE, the FEEDBACK outcome declaration is set to ANSWER_CORRECT. If not it is set to FAILURE.
http://www.edustandaard.nl/nl-qti/1/rptemplates/ RPTEMPLATE_POINT_SCORE_xx	The score is computed using the point mapping declarations in the response declarations (always called RESPONSE_xx). No answer yields a score of 0.0.
http://www.edustandaard.nl/nl-qti/1/rptemplates/ RPTEMPLATE_POINT_SCORE_FB1_xx	The score is computed using the point mapping declarations in the response declarations (always called RESPONSE_xx). No answer yields a score of 0.0. If the score is greater than the default value of the outcome declaration TRESHOLD_VALUE, the FEEDBACK outcome declaration is set to ANSWER_CORRECT. If not it is set to FAILURE.

5.2.4 **\<modalFeedback\>** *elements*

- `<modalFeedback>` elements are only allowed to occur when an item has feedback
- Here is an example (the ANSWER_CORRECT entry is optional):

```
<!-- modalFeedback voor ANSWER_CORRECT is optioneel -->
<modalFeedback outcomeIdentifier="FEEDBACK" identifier="ANSWER_CORRECT"
    showHide="show"> … </modalFeedback>
<modalFeedback outcomeIdentifier="FEEDBACK" identifier="FAILURE"
    showHide="show"> … </modalFeedback>
```

- Attributes of `<modalFeedback>`:

Name	Prf?	M	Type	Remarks
outcomeIdentifier	Lim	1	identifier	Fixed value: "FEEDBACK"
showHide	Lim	1	string	Fixed value: "show"
identifier	Lim	1	identifier	Always one of: "ANSWER_CORRECT" "FAILURE"
title	No			

5.2.5 Disallowed elements

The following response processing or feedback related elements are not allowed:

- `<feedbackInline>`
- `<feedbackBlock>`
- `<printedVariable>`
- `<endAttemptInteraction>`