MIRACLE

Machine-readable and interoperable
age classification labels in Europe

Grant agreement no: 621059

MIRACLE's common data model for age classification information and electronic labels

MIRACLE specification v1.0

October 20th 2014

Deliverable D1.1

Deliverable Type: Other (data vocabulary & documentation)

Nature of the Deliverable: PU

Editors: HBI, JUSPROG, FSM

Contributors: BBFC, NICAM, PEGI, NBCI, OPTENET

Contents

A. Background and objective

Age ratings are highly fragmented, in Europe and globally. In digital contexts, though, electronic age labels open the chance for utilizing classification knowledge across borders. MIRACLE (Machine-readable and interoperable age classification labels in Europe) is a pilot project that aims at providing a common data model for age classification information, implementing it in different national and regional schemes and showing the surplus of interoperable data in applications and end devices.

Accordingly, the first step of the project aimed at developing a common information exchange reference model that will be used as a data specification for all data sets provided throughout the project. The data model proposed in this paper builds on the draft proposed by the Task Force CEO Coalition's Technical Task Force but added several elements in the aftermath of both the public consultation and the discussion with the project's advisory board.

By providing a common data scheme, MIRACLE makes existent rating information machine-readable and technically interoperable. Such data enables software and electronic services to exchange and process data in an easy way, leading to both cost synergies for content and service providers as well as to more information available for service providers, parents and consumers. Additionally the specification proposed here will serve as a guideline for either existing players planning to implement the data model in their schemes or for new players currently considering labelling content to reduce the risk of unrecoverable costs due to individual, less interoperable approaches.

B. MIRACLE's data model for machine-readable age classification data and online labels

Basic principles of the data model

The data model builds on currently existing age labelling practices, as it otherwise would undermine the efforts already taken by companies and rating bodies as well as the classification knowledge that goes with such schemes. For companies and bodies that already provide classification data or label online content electronically no disadvantages should result from this specification.

This starting point leads to three basic principles of the data model:

(1) The data model will be provided in a **technology-neutral** way to reach maximum openness and compatibility between different systems and languages. Therefore, the specification will propose the data structure, its categories, their fields and the possible values of single fields. Examples provided in this document use XML for now; additional examples for JSON and RDF will be provided later on during implementation phase.

(2) It **considers existing electronic labelling schemes** to ensure that those approaches are not undermined by the interoperable data model and can easily be mapped onto it.

(3) It **takes into account existing national and supranational classification schemes**. By doing so, existing visual labels can easily be extended by respective electronic labels, ensuring backwards-compatibility with both the data model and the underlying traditional scheme.

Proposing a data model that adheres to these principles aims at fostering a technically interoperable data structure among different rating schemes, technical implementations and distribution contexts of rated content. Information following this specification is machine-readable and can be processed in different software, apps and electronic appliances.

The fundamental principle of the data model is that every scheme only delivers what it can: Neither existing approaches and schemes nor future ones have to provide information in *all* categories provided by the specification. As long as the data bits that are provided by a label can be mapped on any of the proposed categories, the system is technically interoperable. However, the more information a system or label provides, the better other IT systems will be able to use and process the data.

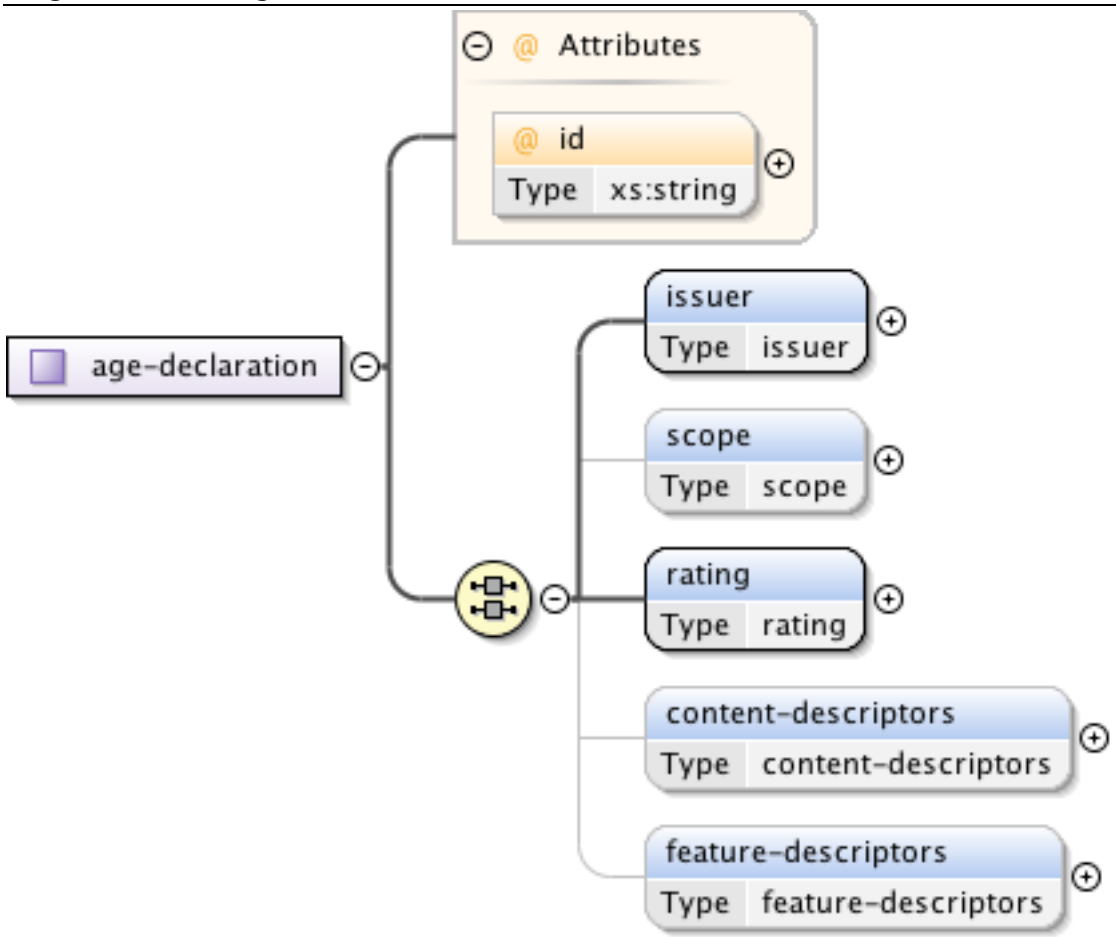
During the development of the specification the MIRACLE consortium has decided not to make use of existing metadata schemes, since they either focus on web resource descriptions only or on media metadata in general; both approaches are fruitful and may be used to extent the MIRACLE specification at a later stage, but both do not fit into specific age classification information-related vocabularies of *all* kinds of media, like movies, games, episodes, video clips, apps – and all being *online and offline* resources. From the experiences made with former approaches (e.g. ICRA, QUATRO) and current practices (Germany: age-de.xml), the labelling process should be understandable by professional metadata engineers, in-house youth protection or editorial staff as well as layman content providers. Complex specifications for describing web resources a label applies to is in risk of overstraining implementation.

Blocks (categories) and fields of the MIRACLE specification

Main blocks of data elements within the data model are:

- The body issuing the classification (<issuer>),
- the scope of the dataset (<scope>),
- age labels (<rating>),
- content descriptors (<content-descriptors>) and
- feature descriptors (<feature-descriptors>).

Diagram data set <age-declaration>



General data set structure in XML

```
<age-declaration>
  <issuer>…</issuer>
  <scope>…</scope>
  <rating>…</rating>
  <content-descriptors>…</content-descriptors>
  <feature-descriptors>…</feature-descriptors>
</age-declaration>
```

Blocks are not in fixed order [xs:all].
Required: <issuer>, <rating>.
Optional: <scope>, <content-descriptors>, <feature-descriptors>.

<age-declaration> may have an attribute "id" in case of multiple datasets regarding the same content, e.g. in case more than one scheme provides a data set.

Block 1: Issuing body - <issuer>

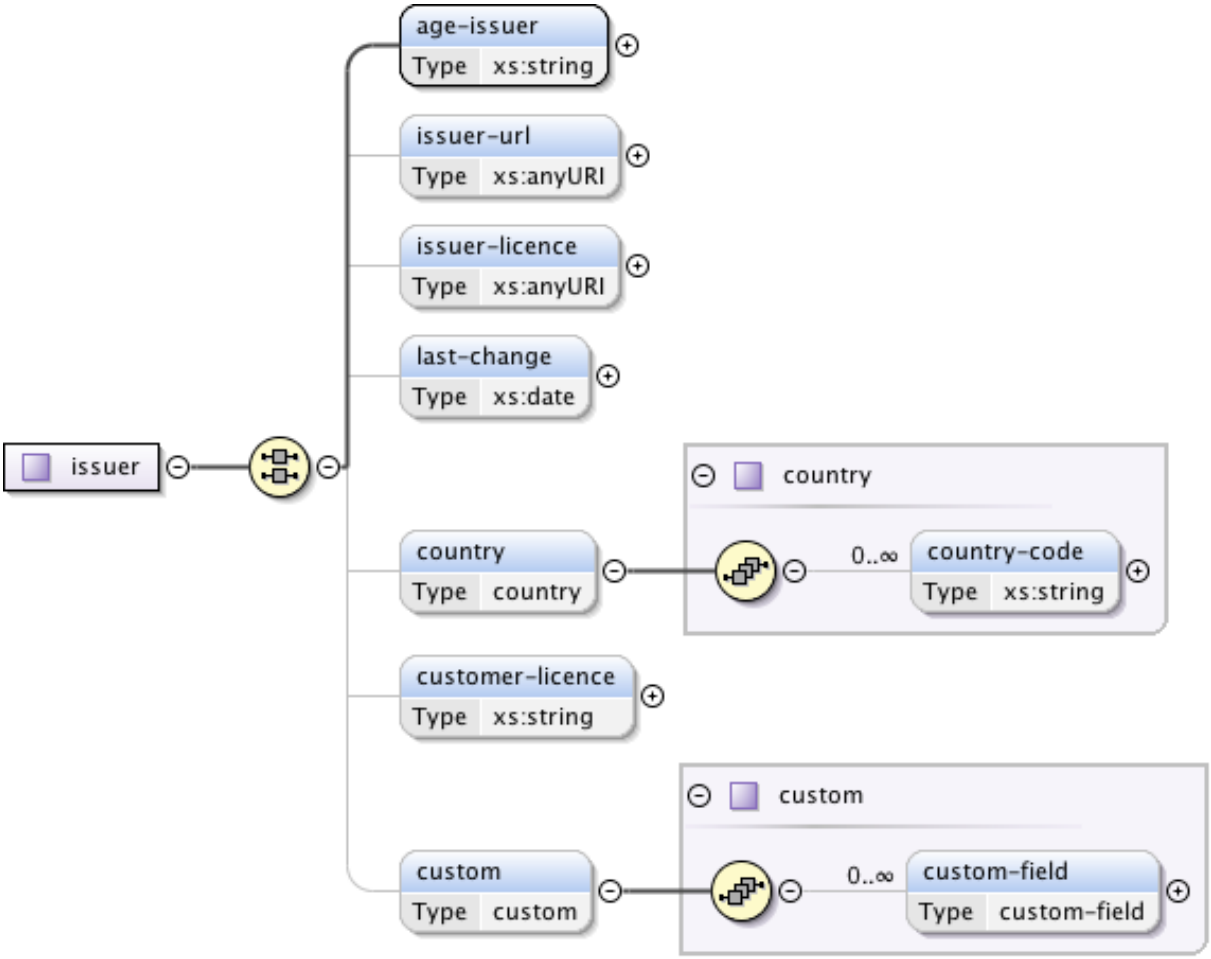
It is important for all content ratings to refer to the body or creator that issued the specific information/label. First, this information links the age label and descriptors to brand or institutional trust. Second, this block allows the assessment of the regional origin of the label and its individual relevance. By providing the date of the last revision of the classification the label can also show its actuality and help in processing ex post changes.

Elements of <issuer> and children do not have to be provided in a fixed order [xs:all].

Name	**Possible values**	**Comments**	**Manda-tory?**
<age-issuer>	shortText [xs:string]	Since age classification can be based on self-classification, this field is open for all kinds of issuing bodies (e.g. PEGI, NICAM, FSM, FSK, „own", „Company Name"etc.).	yes
<issuer-url>	URL [xs:anyURI]	Reference to web address of issuing body, where additional information on institutional background and rating procedures are provided, for instance.	no
<issuer-licence>	shortText [xs:anyURI]	URL with link to the issuer's licence regarding the classification information, i.e. the type of licence under which the data is being provided (e.g. copyright, creative commons, public domain).	no
<last-change>	Date (yyyy-mm-dd) [xs:date]	The date of the most recent decision or update regarding a classification (e.g. 2014-08-24).	no
<country> children: <country-code>	Two-letter ISO country code (upper case); special values for EU (eu) and worldwide (all). [xs:string]	Country abbreviation (e.g. DE, NL, GB, etc.; EU-wide: eu; worldwide: all; multiple countries use child-tag <country-code> for each country. This tag informs about region(s) the label	no

		comes from and/or is (not necessarily legally) applied to.	
<customer-licence>	shortText [xs:string]	Information or ID of the customer that provides this information (licensee); not the issuing body	no
<custom> children: <custom-field class="">	shortText [xs:string]	Open text field element that the issuing body can freely use for various reasons; the class-attribute is obligatory in case a <custom-field> is used; it can be used both for internal reasons as well as for expressly pointing out underlying schemes.	no

Diagram – block <issuer>



Block snippet from an exemplary dataset in XML

```
<issuer>
    <age-issuer>PEGI</age-issuer>
    <issuer-url>http://www.pegi.eu</issuer-url>
    <issuer-licence>http://www.pegi.eu/licence</issuer-licence>
    <last-change>2014-06-25</last-change>
    <country>
        <country-code>eu</country-code>
        <country-code>NO</country-code>
       <country-code>CH</country-code>
    </country>
    <customer-licence>USER-2938237</customer-licence>
    <custom>
        <custom-field class="PEGI-custom">PEGI-related custom
        field content</custom-field>
    </custom>
</issuer>
```

Block 2: Label metadata - <scope>

Depending on the existing form of age classification data (a central classification data base that can be queried from third parties vs. a specific age label attached to specific content), each piece of interoperable classification information has to provide information that identifies the specific content the given age information applies to („scope"). This scope might either aim at specific, isolated media content that –

sometimes in a nation- or region-specific version – has already been classified or it aims at a website or service, or parts of those.

The scope of application is depending on the context of its implementation:

a) Contrary to online labels another form of providing classification data is to offer *centralised databases with classification data*. Usually, existing rating bodies will opt for such forms of data provisions. The issue here is that for instance an online shop or a VOD service needs to query the database for valid age classification information. To get the correct output both the data provider as well as the demander will have to use unique identifiers or fuzzy approximations via an object title. As the label is detached from the content, the database query has to aim at getting the correct information back from the database. In practice, such UIDs are not being used coherently throughout all existing rating bodies – and even if a rating body uses UIDs, they aren't the same among different rating bodies. Hence, an alternative can be to base queries on the title of the media content (e.g. movie title, game title). This fuzzy approach might, however, lead to several results, as in many cases different versions of a game or film have been classified by a rating body (cinematic version, DVD version, Director's cut, uncut version, TV version etc.).

To remain flexible, the data model opts for offering all three fields: a scope URL, an UID field and a title field. These fields are not mandatory and can be used either alone or in combination. As each of the field can have one attribute, each issuing body will be able to attach system-specific UIDs to a dataset; e.g. country, rating system, producer, internal ERP or logistics numbers, EAN, GSDN, etc.).

b) For online labels, an URI-based scope is necessary to clarify the scope of the classification. The basic approach is to take an age label for the whole URI authority, usually the second level domain. However, MIRACLE provides possibilities for defining URI-based scopes (specific subdomains, files or folders); for FQDN and URLs, the following wildcards are defined:

> *(1) Subdomains:* *.example.com covers all subdomains including www.example.com, sub1.sub2.example.com, including example.com without a subdomain. It can also be used as *.sub1.example.com with similar rules. Wildcards for second level domains are not allowed (e.g. *.eu or *.de is not valid).
>
> *(2) Paths, i.e. URI-based scopes behind the first slash*:
> www.example.com only covers the main page, nothing else;
> www.example.com/* includes all URLs of www.example.com, including the main page and e.g. www.example.com/folder1/index.php;
> www.example.com/folder1/* covers all URLs inside this folder like www.example.com/folder1/pic1.jpg or www.example.com/folder1/index.php?par=1, including www.example.com/folder1.
>
> Different protocols do not lead to different scopes: https://www.example.com has the same scope as http://www.example.com (both: www.example.com or with wildcard *.example.com).

Temporary exemption for interpreting existing age-de.xml labels

These rules apply to all MIRACLE labels. For existing age-de.xml labels in Germany, please be advised that the interpretation of scopes currently follows these rules:

(1) Paths: age-de using www.example.com *apply to all URLs* of www.example.com, e.g. www.example.com/folder1/index.php. www.example.com/folder1 respectively covers all URLs inside this folder including the URL www.example.com/folder1 itself.

(2) The labelling of the main page with an age rating that differs from the rest of a website/subdomain is not possible in age-de.xml.

The age-de.xml label specification will be updated in mid-term according to the MIRACLE specification. To provide an age-de.xml label for a whole website that is fully processable by both age-de and MIRACLE based interpreters, please use

<scope>*.example.com/*<scope>
<scope>example.com/*</scope>
<scope>*.example.com</scope>

(Label regarding folders respectively.)

Examples for <scope-url>

Whole content of website at http://www.example.com (including all subdomains, example.com and the main page)	`<scope-url>`
	`*.example.com/*`
	`</scope-url>`

Whole content of website at http://www.example.com (but not any other subdomains than www.)	`<scope-url>`
	`www.example.com/*`
	`</scope-url>`

Whole content of subdomain http://sub1.example.com (including all further level subdomains like http://sub2.sub1.example.com but not www.example.com)	`<scope-url>`
	`*.sub1.example.com/*`
	`</scope-url>`

Content of website folder at http://www.example.com/folder1/	`<scope-url>`
	`www.example.com/folder1/*`
	`</scope-url>`

Content of website folder at http://subdomain.example.com/folder1/	`<scope-url>`
	`subdomain.example.com/folder1/*`
	`</scope-url>`

Content of a page within website folder at http://www.example.com/folder1/example.html	`<scope-url>`
	`www.example.com/folder1/example.html`
	`</scope-url>`

Content of the main page as well as one page within website folder at http://www.example.com/folder1/example.html	`<scope-url>`
	`www.example.com`
	`</scope-url>`
	`<scope-url>`
	`www.example.com/folder1/example.h`

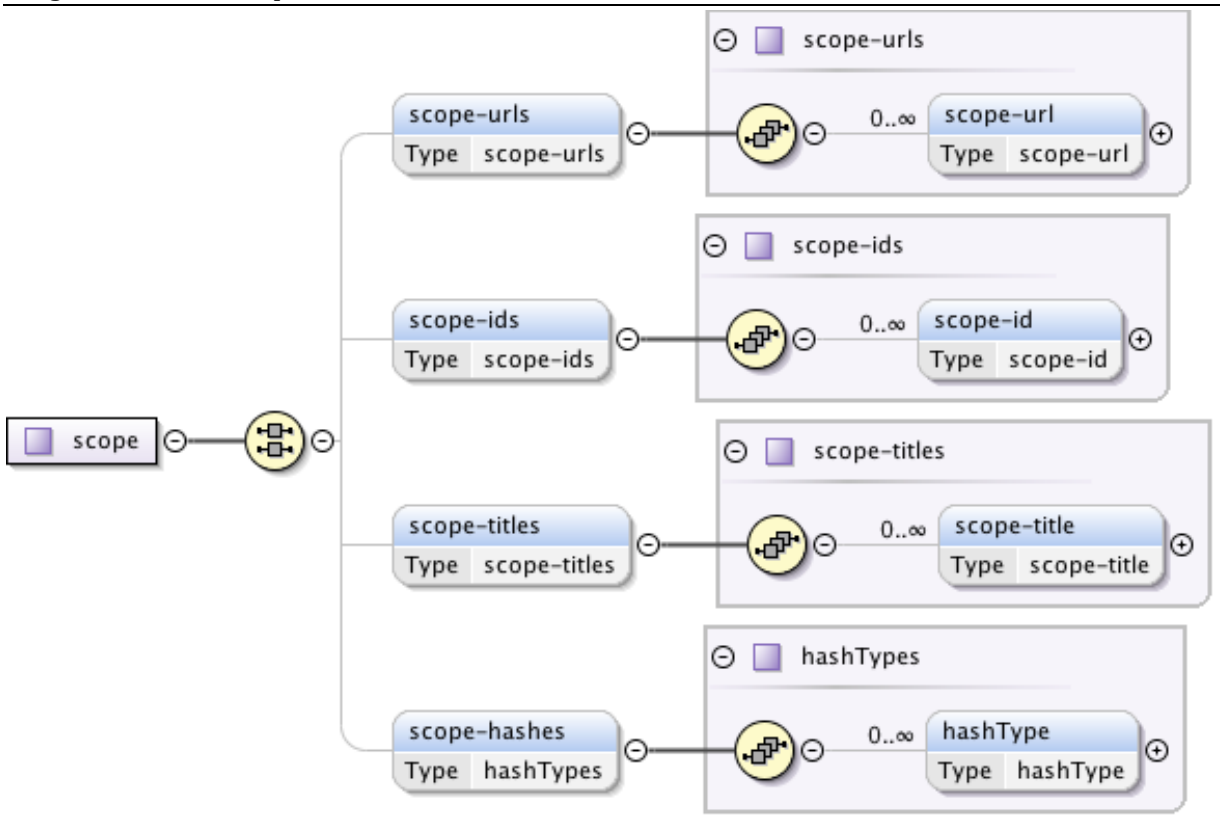
	tml </scope-url>
Only the content of the main page of the website at http://www.example.com	<scope-url> www.example.com </scope-url>

Elements of <scope> and children do not have to be provided in a fixed order [xs:all]. Depending on the context a data set regularly might only contain one of the available scope element types; however, if needed, several types can be used in parallel (e.g. scope-title and scope-id elements). In such cases, each type is defining the scope of the label independently of each other.

Name	**Possible values**	**Comments**	**Mandatory?**
<scope-urls> children: <scope-url class="">	FQDN, optionally making use of asterisks/"*", where *. is a wildcard for subdomains (third and higher level domains), covering complete second and higher level domains (*.example.com) "/*" on the right side means label applies to all URLs within that folder hierarchy [xs:string]	FQDN-/URL-based scope of application of a label; the class-attribute can be used both for internal reasons as well as for expressly pointing out the underlying scheme. URL covers all protocols (e.g. http/https). No protocol-information has to be given. More than one scope-url is allowed.	No
<scope-ids> children: <scope-id class="">	Alphanumeric [xs:string]	UID of classified content; the class-attribute can be used both for internal reasons as well as for expressly pointing out the underlying	No

		scheme. More than one scope-id element is allowed.	
<scope-titles> children: <scope-title class="">	shortText [xs:string]	Title of classified content (system-specific); the class-attribute can be used both for internal reasons as well as for expressly pointing out the underlying scheme. More than one scope-title element is allowed (e.g. to label TV episodes).	No
<scope-hashes> children: <scope-hash class="">	32-bit hash value	Sometimes an easy way of describing a scope is to refer to a file's hash value; this is the element that may contain the respective information; the class-attribute can be used both for internal reasons as well as for expressly pointing out the hash scheme.	No

Diagram – block <scope>



Snippet from an exemplary dataset in XML (Note: For exemplification purposes only, the example includes different scope types that might not make sense in real life.)

```
<scope>
  <scope-urls>
    <scope-url class="web-url">*.example.com/supergame/*</scope-url>
  </scope-urls>
   <scope-ids>
     <scope-id class="PEGI-classification-no">18423</scope-id>
     <scope-id class="EAN">9783125171341</scope-id>
   </scope-ids>
   <scope-titles>
     <scope-title class="PEGI-title-en">Supergame Title
     </scope-title>
      <scope-title class="title-de">Superspiel Titel</scope-title>
     <scope-title class="PEGI-version">1.2</scope-title>
     <scope-title class="PEGI-language">en</scope-title>
   <scope-titles>
</scope>
```

Block 3: Age label - <rating>

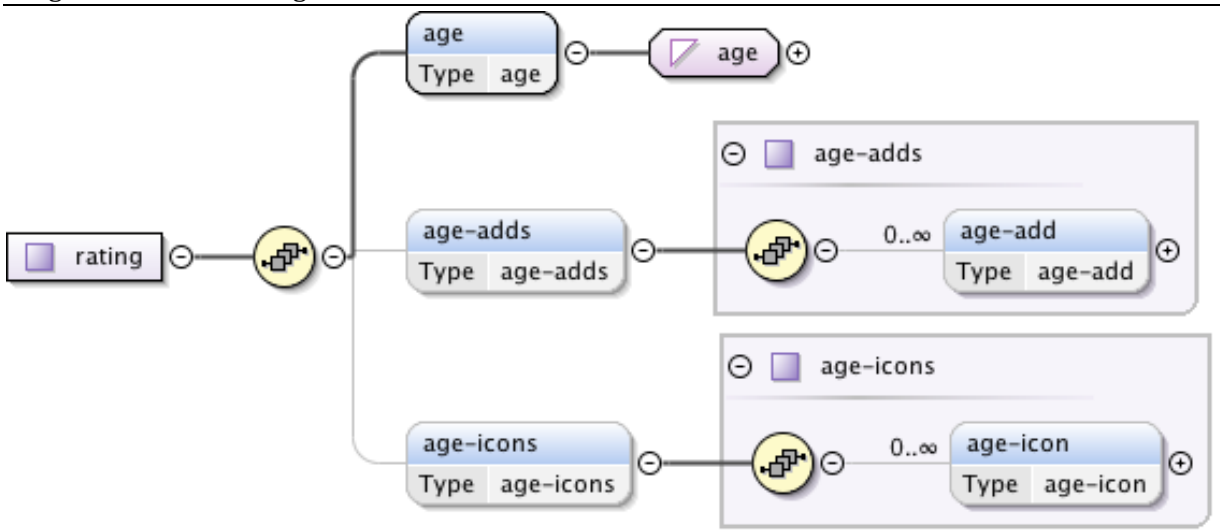
Age labels are a common approach in content rating systems worldwide – age ratings hence are the core aspect of interoperable rating information. However, there are different schemes of how to provide age-specific information (specific age, age group, age group description, or additional age information like parental guidance). As numeric values can be processed by machines in an optimal way compared to open text fields, it is deemed most feasible to translate textual age classification like „teens" or „mature" into numbers on side of the classification scheme (and its API) rather than to translate textual values in the data model into number on processor-/client-side. The data model will encompass these differences by making the numeric age field mandatory, but provides options to extend the data structure by additional age-related data fields.

Elements of <scope> and children do not have to be provided in a fixed order [xs:all].

Field	Possible values	Comments	Mandatory?
<age>	Numeric [xs:integer] min=0 max=99	Minimum age for that the content is deemed suitable. Most age classification systems already use numeric values. However, systems that do not will have to provide translation tables to provide numeric values here (e.g. US ESRB: E→0; E10+→10; T→13; M→17; A→18). Schemes that do not include any age classification (e.g. just content or feature descriptors) should use the attribute "class" with "na" and "-1" as a value. Each dataset must only have one <age> element.	Yes
<age-adds> children: <age-add class="">	shortText [xs:string]	If a system uses additional values to specify an age classification, this additional age information can be provided here, e.g. PG, R18, 12A, X etc. Systems that rely on non-numeric classification (e.g. ESRB) can provide their original rating in this field, too. By providing a class attribute the data structure is able to provide a scheme-consistent explanation. More than one <age-add> element is allowed.	No
<age-icons> children: <age-icon class="">	URL [xs:anyURI]	To provide trustful classifications, this field provides the URL to the original age rating icon that can be used for displaying an age rating icon if requested. Protocol (e.g. http://) has to be included. More than one <age-icon>	No

		element is allowed.	

Diagram – block <rating>



Snippet from an exemplary dataset in XML

```
<rating>
   <age>12</age>
   <age-adds>
    <age-add class="PEGI-age">12+</age-add>
   <age-adds>
   <age-icons>
      <age-icon class="PEGI-icon">http://pegi.eu/label/12.png
      </age-icon>
   <age-icons>
</rating>
```

Block 4: Content descriptors - <content-descriptors>

Several existing classification schemes use content descriptors to give additional information about the content and the reasons for a specific age rating. The categories of these descriptors are quite comparable worldwide. However, there are and always will be peculiarities of single systems or schemes, showing the need for a flexible approach. The data model provides a mixed data field model here, where common and agreed categories are predefined, while the specification permits additional content descriptor elements for the sake of flexibility. In this category, too, icons are commonly used as content descriptors. The data model hence provides URL references to these icons.

If a pre-defined content descriptor is applicable, the allowed values are true/false only. In this context it doesn't matter if the issuing body considers the respective relevant content as mild or strong, as long as it is relevant for the age classification decision. All content descriptor fields are optional, since some existing schemes do not provide content descriptors at all. Hence if a field is not provided by a data set, this does *not* default to "no".

Elements of <content-descriptors> do not have to be provided in a fixed order [xs:all]. The following categories are not slimming down the options of possible content descriptors; each scheme might introduce its own sets of descriptors. However, to achieve a small set of best practice categories, schemes using MIRACLE are strongly encouraged to map their descriptors to the following ones (and add their scheme-specific fields as additional ones where deemed necessary). This way MIRACLE will be able to fulfil an integrating function regarding different classification schemes.

Pre-defined data fields (standardised content descriptors)

Name	**Possible values**	**Comments**
<cd-sexuality>		
<cd-sexuality-exist>	true/false [xs:boolean]	Sex/eroticism/nudeness is a reason for the age classification
<cd-sexuality-desc>	shortText [xs:string]	Description of content
<cd-sexuality-icon>	URL [xs:anyURI]	Address of original sex/erotic/nudeness icon
<cd-violence>		
<cd-violence-exist>	true/false [xs:boolean]	Violence/weapons/ blood is a reason for the age classification
<cd-violence-desc>	shortText [xs:string]	Description of content
<cd-violence-icon>	URL [xs:anyURI]	Address of original violence/weapons/blood icon
<cd-discrimination>		
<cd-discrimination-exist>	true/false [xs:boolean]	Discrimination/ racism/ hate speech is a reason for the age classification
<cd-discrimination-desc>	shortText [xs:string]	Description of content
<cd-discrimination-icon>	URL [xs:anyURI]	Address of original discrimination/ racism/ hate speech icon
<cd-cursing>		
<cd-cursing-exist>	true/false [xs:boolean]	Obscene language/bad language/cursing is a reason for the age classification

<cd-cursing-desc>	shortText [xs:string]	Description of content
<cd-cursing-icon>	URL [xs:anyURI]	Address of original obscene/bad language/cursing icon
<cd-drugs>		
<cd-drugs-exist>	true/false [xs:boolean]	Drugs/tobacco/alcohol is a reason for the age classification
<cd-drugs-desc>	shortText [xs:string]	Description of content
<cd-drugs-icon>	URL [xs:anyURI]	Address of original drugs/tobacco/alcohol icon
<cd-fear>		
<cd-fear-exist>	true/false [xs:boolean]	Fear/shock is a reason for the age classification
<cd-fear-desc>	shortText [xs:string]	Description of content
<cd-fear-icon>	URL [xs:anyURI]	Address of original fear/shock icon
<cd-gambling>		
<cd-gambling-exist>	true/false [xs:boolean]	Gambling is a reason for the age classification
<cd-gambling-desc>	shortText [xs:string]	Description of content
<cd-gambling-icon>	URL [xs:anyURI]	Address of original gambling icon

If additional relevant content categories emerge or a scheme would like to provide additional content descriptors than the ones pre-defined, it might do so in the following way within the block <cd-other>.

Example for additional content descriptors

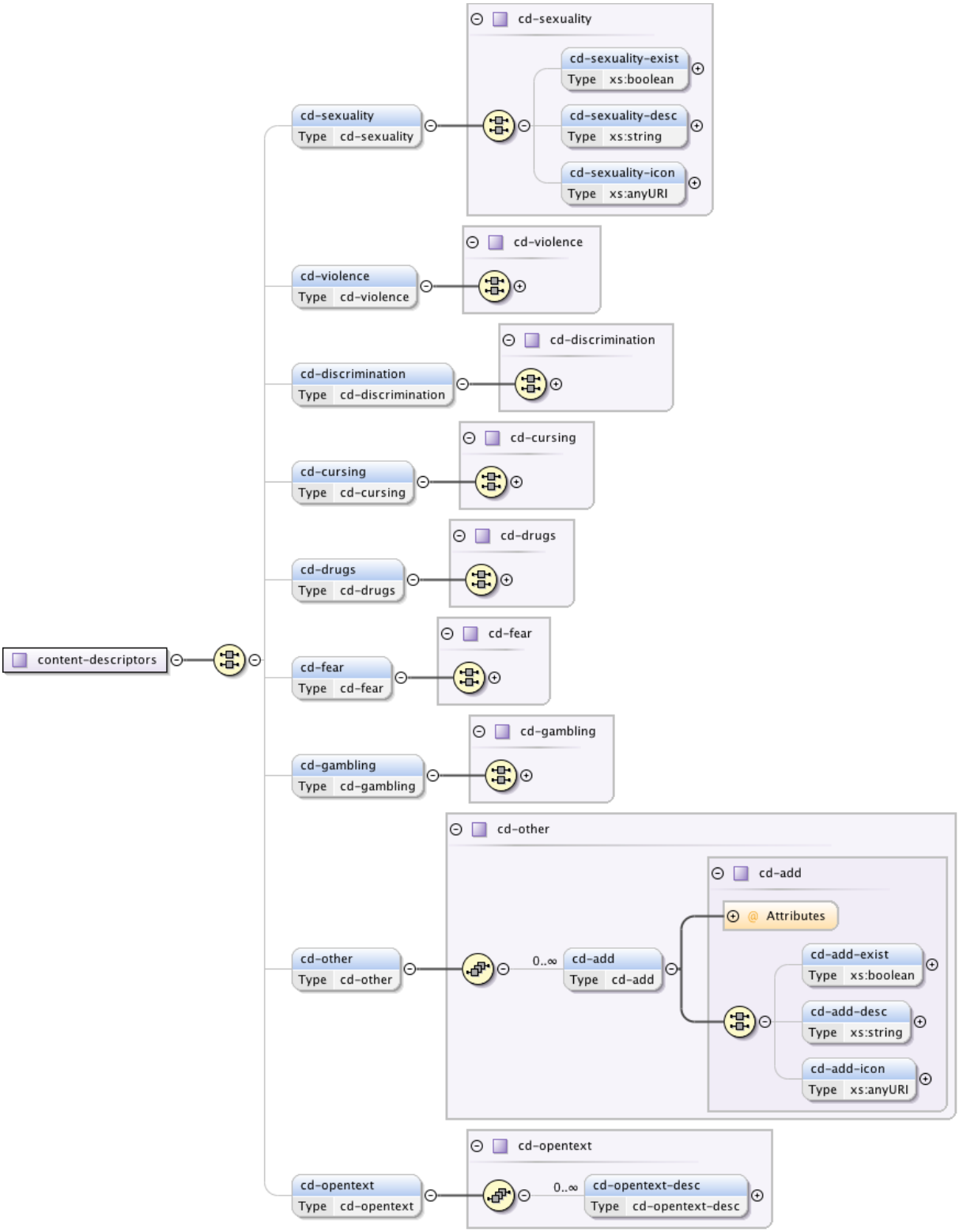
Field name	**Possible values**	**Comments**
<cd-add class="cd-xyz"> (e.g. class="cd-selfharm"; class="cd-antisocial")		The class-attribute is mandatory here, as it contains information about the additional descriptor.
<cd-add-exist>	true/false	

	[xs:boolean]	
<cd-add-desc>	shortText (short description of additional content descriptor) [xs:string]	
<cd-add-icon>	URL; address of original additional icon [xs:anyURI]	

Some systems do not provide systematic content descriptors, but offer additional information regarding the reasoning for a specific age rating in text form. Such information is harder to structure and to process technically but it still provides relevant information. Later on, systems might be able to process this data automatically, too.

Field name	**Possible values**
<cd-opentext>	
<cd-opentext-desc class="">	shortText [xs:string]

Diagram – block <content-descriptors>



Snippet from an examplary dataset in XML

```
<content-descriptors>
      <cd-violence>
          <cd-violence-exist>true</cd-violence-exist>
          <cd-violence-icon>http://pegi.eu/label/violence.png
          </cd-violence-icon>
      </cd-violence>
      <cd-fear>
```

```
            <cd-fear-exist>true</cd-fear-exist>
            <cd-fear-icon>http://pegi.eu/label/fear.png
            </cd-fear-icon>
        </cd-fear>
        <cd-other>
            <cd-add class="self-harm">
                <cd-add-exist>true</cd-add-exist>
                <cd-add-icon>http://pegi.eu/label/self-harm.png
                </cd-add-icon>
            </cd-add>
        </cd-other>
            <cd-opentext class="PEGIonline">
                <cd-opentext-desc>Online game</cd-opentext-desc>
            </cd-opentext>
    </content-descriptors>
```

Block 5: Feature descriptors - <feature-descriptors>

A different type of descriptors may relate to information about features or functionalities that the content (or better: the content-related service or platform) provides to the user. Depending on the scheme, this information sometimes results in a specific age classification result, sometimes it is only regarded as additional information for end users without an effect on the actual age rating decision. Information regarding such features are relevant for minors and other consumers, too; e. g. user generated content might contain relevant depictions that would change existing age classifications, chat functionalities result in unknown people approaching (underage) user in a harmful way or location-based services log and display the movement and/or other person-related information to third parties. For instance, the PEGI scheme already started to implement such descriptors. Hence a first step is to take those as predefined ones, while leaving the feature descriptor field open to new ones, too (see above). All feature descriptor fields are optional, since most existing schemes do not provide content descriptors. If a field is not provided by a data set, this does *not* default to "no".

Elements of <feature-descriptors> and children do not have to be provided in a fixed order [xs:all]. The following categories are not slimming down the options of possible feature descriptors; each scheme might introduce its own sets of descriptors here. However, to achieve a small set of best practice categories, schemes using MIRACLE are encouraged to map their descriptors to the following ones (and add their scheme-specific fields as additional ones, where deemed necessary; see above).

Pre-defined data fields regarding features (standardised feature descriptors)

Name	**Possible values**	**Comments**
<fd-inapppurchase>		
<fd-inapppurchase-exist>	true/false [xs:boolean]	The service contains elements enabling the consumer to purchase additional content or functionality, regardless of whether the app itself was acquired for free or

		not.
<fd-inapppurchase-desc>	shortText [xs:string]	Description of feature
<fd-inapppurchase-icon>	URL [xs:anyURI]	Address of original icon for in-app purchase features.
<fd-personaldatasharing>		
<fd-personaldatasharing-exist>	true/false [xs:boolean]	The service gives its provider (or a third party) access to personal data such as home address, contact details or bank account numbers.
<fd-personaldatasharing-desc>	shortText [xs:string]	Description of feature
<fd-personaldatasharing-icon>	URL [xs:anyURI]	Address of original icon for personal data sharing features.
<fd-locationdatasharing>		
<fd-locationdatasharing-exist>	true/false [xs:boolean]	The service contains the option to share exact location on a map when using the service. The location information may be shared publicly or with a specific network inside the service or beyond.
<fd-locationdatasharing-desc>	shortText [xs:string]	Description of feature
<fd-locationdatasharing-icon>	URL [xs:anyURI]	Address of original icon for location data sharing features.
<fd-chat>		
<fd-chat-exist>	true/false [xs:boolean]	The service includes an option for a user to chat with other users via the app. These users may operate under a pseudonym or anonymously.
<fd-chat-desc>	shortText	Description of content

	[xs:string]	
<fd-chat-icon>	URL [xs:anyURI]	Address of original icon for chat features.

Similarly to the content descriptors, additional feature descriptors will emerge during time. Hence, the data has to be open to additional or new descriptors, too.

Example for additional feature fields (additional feature descriptors)

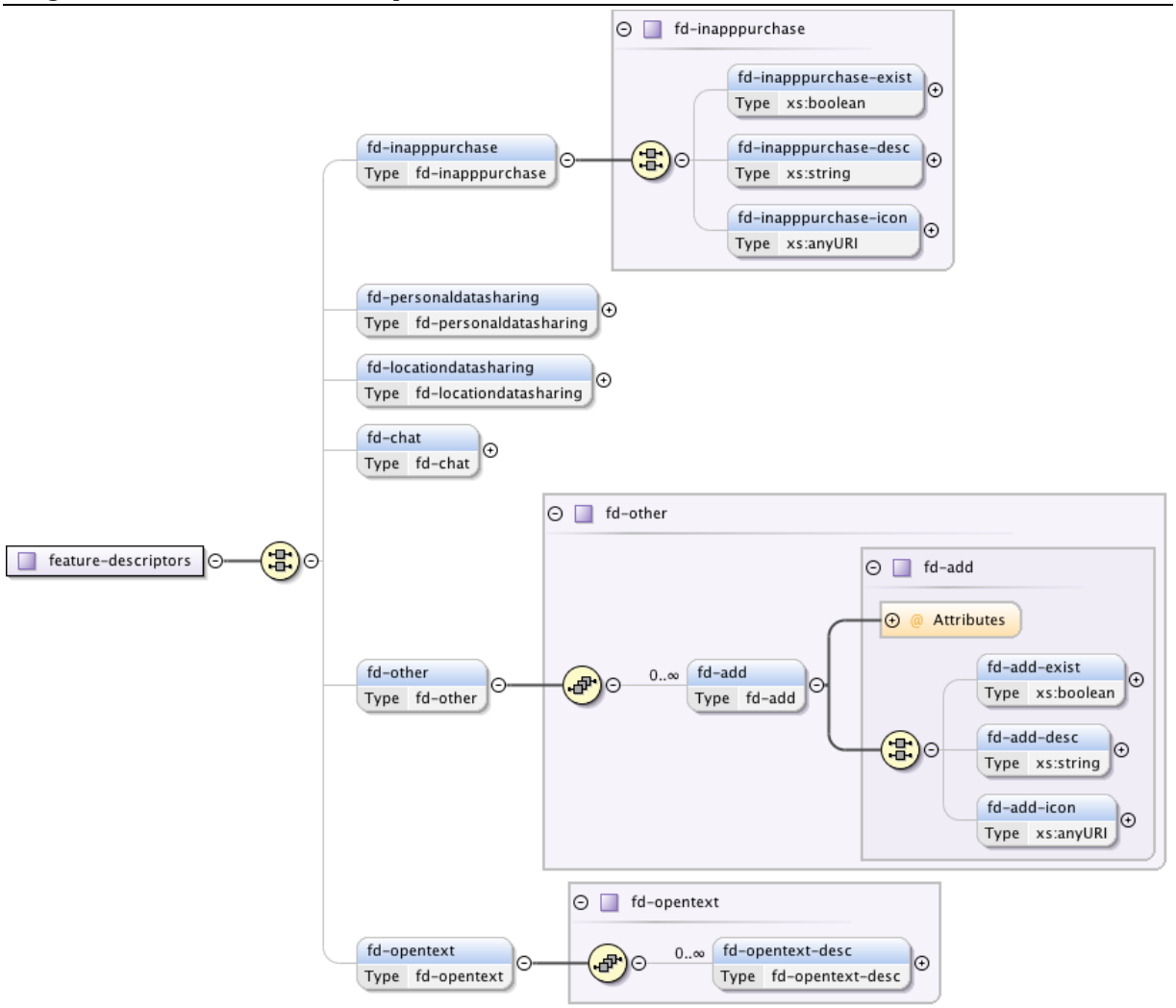
Field name	**Possible values**	**Comments**
<fd-add class="fd-xyz"> (e.g. class="fd-upload")		The class-attribute is mandatory here, as it contains information about the additional descriptor
<fd-add-exist>	true/false [xs:boolean]	
<fd-add-desc>	shortText (short description of additional feature descriptor) [xs:string]	
<fd-add-icon>	Address of original additional icon [xs:anyURI]	

Such additional fields might become pre-defined fields in later versions of the data model, when deemed necessary.

Some systems do not provide systematic feature descriptors but offer additional information in text form. Such information is harder to structure and to process technically but it still provides relevant information (see above).

Field name	**Possible values**
<fd-opentext>	
<fd-opentext-desc class="fd-xyz">	

Diagram – block <feature-descriptors>



Snippet from an exemplary dataset in XML

```
    <feature-descriptors>
        <fd-inapppurchase>
            <fd-inapppurchase-exist>true
            </fd-inapppurchase-exist>
            <fd-inapppurchase-icon>http://pegi.eu/label/iap.png
            </fd-inapppurchase-icon>
        </fd-inapppurchase>
        <fd-personaldatasharing>
            <fd-personaldatasharing-exist>true
            </fd-personaldatasharing-exist>
            <fd-personaldatasharing-icon>
            http://pegi.eu/label/pds.png
            </fd-personaldatasharing-icon>
        </fd-personaldatasharing>
        <fd-chat>
            <fd-chat-exist>1</fd-chat-exist>
            <fd-chat-icon>http://pegi.eu/label/chat.png
            </fd-chat-icon>
```

```
        </fd-chat>
           <fd-other>
            <fd-add class="self-harm">
                <fd-add-exist>true</fd-add-exist>
                <fd-add-icon>http://pegi.eu/label/self-harm.png
                </fd-add-icon>
            </fd-add>
        </fd-other>
           <fd-opentext>
                <fd-opentext-desc class="PEGI-fd-info">
                Text about additional features
                </fd-opentext-desc>
           </fd-opentext>
    </feature-descriptors>
```

Annex 1: Exemplary XML data sets in full

Disclaimer: This is a completely fabricated, hypothetical data set for demonstration purposes only. All content is for demonstration purposes and names of content titles or organisations are only used for practical clarification.

Example of shortest possible version of a MIRACLE data set in XML

```
<?xml version="1.0" encoding="UTF-8"?>
<age-declaration xmlns="http://www.miracle-label.eu/ns/”
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation=" http://www.miracle-label.eu/ns/ miracle-1-
0.xsd">
    <issuer>
       <age-issuer>myhomepage.cz</age-issuer>
    </issuer>
    <scope>
       <scope-urls>
          <scope-url>*.myhomepage.cz/*</scope-url>
       <scope-urls>
    </scope>
    <rating>
        <age>6</age>
    </rating>
</age-declaration>
```

Long version with different examples

(Note: for example reasons some information might not make sense in real life)

```
<?xml version="1.0" encoding="UTF-8"?>
<age-declaration xmlns="http://www.miracle-label.eu/ns/”
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation=" http://www.miracle-label.eu/ns/ miracle-1-
0.xsd">
    <issuer>
        <age-issuer>PEGI</age-issuer>
        <issuer-url>http://www.pegi.eu</issuer-url>
        <issuer-licence>C0-2124-23443</issuer-licence>
        <last-change>2014-06-25</last-change>
        <country>
            <country-code>eu</country-code>
            <country-code>NO</country-code>
           <country-code>CH</country-code>
        </country>
        <custom>
            <custom-field class="PEGI-custom">
            PEGI-related custom field content
            </custom-field>
        </custom>
    </issuer>
    <scope>
      <scope-urls>
        <scope-url class="web-url">*.example.com/supergame/*
        </scope-url>
```

```
      </scope-urls>
      <scope-ids>
        <scope-id class="PEGI-classification-no">18423</scope-id>
      </scope-ids>
      <scope-titles>
        <scope-title class="PEGI-title-en">Supergame Title
        </scope-title>
         <scope-title class="title-de">Superspiel Titel
        </scope-title>
        <scope-title class="PEGI-version">1.2</scope-title>
        <scope-title class="PEGI-language">en</scope-title>
      </scope-titles>
    </scope>
    <rating>
        <age>12</age>
      <age-adds>
        <age-add class="PEGI-age">pegi12+</age-add>
      </age-adds>
      <age-icons>
        <age-icon class="PEGI-icon">
        http://pegi.eu/label/12.png</age-icon>
      </age-icons>
    </rating>
    <content-descriptors>
        <cd-violence>
            <cd-violence-exist>true</cd-violence-exist>
            <cd-violence-icon>http://pegi.eu/label/violence.png
            </cd-violence-icon>
        </cd-violence>
        <cd-fear>
            <cd-fear-exist>true</cd-fear-exist>
            <cd-fear-icon>http://pegi.eu/label/fear.png
            </cd-fear-icon>
        </cd-fear>
        <cd-other>
            <cd-add class="self-harm">
                <cd-add-exist>true</cd-add-exist>
                <cd-add-icon>http://pegi.eu/label/self-harm.png
                </cd-add-icon>
            </cd-add>
        </cd-other>
            <cd-opentext>
                <cd-opentext-desc class="PEGIonline">
                Online game</cd-opentext-desc>
            </cd-opentext>
    </content-descriptors>
    <feature-descriptors>
        <fd-inapppurchase>
            <fd-inapppurchase-exist>true</fd-inapppurchase-exist>
            <fd-inapppurchase-icon>http://pegi.eu/label/iap.png
            </fd-inapppurchase-icon>
        </fd-inapppurchase>
        <fd-personaldatasharing>
            <fd-personaldatasharing-exist>true
            </fd-personaldatasharing-exist>
            <fd-personaldatasharing-icon>
            http://pegi.eu/label/pds.png
            </fd-personaldatasharing-icon>
```

```
        </fd-personaldatasharing>
        <fd-chat>
            <fd-chat-exist>true</fd-chat-exist>
            <fd-chat-icon>http://pegi.eu/label/chat.png
            </fd-chat-icon>
        </fd-chat>
           <fd-other>
            <fd-add class="self-harm">
                <fd-add-exist>true</fd-add-exist>
                <fd-add-icon>http://pegi.eu/label/self-harm.png
                </fd-add-icon>
            </fd-add>
        </fd-other>
           <fd-opentext>
               <fd-opentext-desc class="PEGI-fd-info">
               Text regarding additional features</fd-opentext-desc>
           </fd-opentext>
    </feature-descriptors>
</age-declaration>
```

Annex 2: XSD for MIRACLE XML data set

There will be provided an XML Schema Definition for the MIRACLE specification shortly. It will document all elements and types in detail and might be used to validate conformity of MIRACLE-compatible XML instances.

The XSD file and its technical documentation will available at
http://www.miracle-label.eu/data-model