

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
access-count	EL	The number of times the actor accessed the current run of the unit of learning. (datatype=integer.
act	EL	A play is modelled according to the metaphor of an theatrical performance. A play consists of a series of acts and an act consists of a series of role-part's. In an act each role may be present in only one role-part. An act represents a series of concurrent role-part's. There is at least one act in a play. When there is more then one act in a play these are interpreted in a sequenced manner: from first act to last act. Only one act in a play is the active act at any moment in time, starting with the first. When the first act is completed, the second act is made the active act. The first is still visible and accessible, but in the interface it is made clear that it this is only looking in history. When the second act is completed, the third act is made active, etc. Acts which are sequenced in row after the current active act are never visible. Conditions cannot overrule this, meaning that the act is of higher priority than conditions.
act-ref	EL	*EML new* Refers to an act (in method/play/act).
activities	EL	This element is renamed from 'content' to 'activities' and is a container for activity definitions, including related constructs as 'activity-structure'.
activity-description	EL	Alias: task. The activity-description is the actual cue given to the user (rendered in the user-interface) to describe the activity to be performed by the user. In most cases the activity-description is a text (of type webcontent or emlcontent). In other cases it can be an audio-file (webcontent), a video file or any other cue to the user. Activity-descriptions also define the environment for the activity. Every noun mentioned in the description refers to a resource in the environment. It is up to the author to have a strict representation of the nouns in the environment or a more open one (leaving nouns implicit). Example: activity-description: "Solve problem X, write a report and discuss this in the group." environment-resources: 'problem X' (type: knowledge-object), 'report' (type: property accessible in e.g. knowledge-object), 'group' (e.g. a conference-service). The EML 1.0 elements:what, how, with-hom can be modelled by defining different items with the respective item title's (take care that the title of the item is described IN the item). The top-level title is provided when more than one item is included. E.g.: title="study task" item/title="what" item/title="how" item/title="with-whom"
activity-progression	EL	Activity-progression provides information about the status of the progression of: - the run of the current unit of learning (one result per run at any moment, same result for all persons, so returned for one). - learning-activities (per person, default in own dossier, when in context of monitor/role-ref than for all persons). - support-activity (per person, default in own dossier, when in context of monitor/role-ref than for all persons). - activity-structure (per person, default in own dossier, when in context of monitor/role-ref than for all persons). - play (one result per run at any moment, same result for all persons, so returned for one). - act (one result per run at any moment, same result for all persons, so returned for one). It returns the value of the progression-status in a table when there is more than one result. The values per item can be any of these: (not started started completed). The table has to be designed by implementers. At least the following general information must be provided: - what type of object the table is about (unit-of-learning, act, etc.) - the date of production of the table - the dossier the table is derived from (self, role-ref: indicate which role). The table columns contain at least the user identification, the title of the object and the progression status.
activity-selection	EL	A selection is a structure where users may select and complete the activities contained in any order. Selections may be nested with other sequences or selections. When the attribute 'number-to-select' is set, the activity-selection is completed when the total number of activities selected are completed. The number-to-select must be similar or smaller than the number of activities (including unit-of-learning's) which are at the immediate child level. When the number-to-select isn't set, the activity-selection is completed when all the activities in the selection are completed. The attribute 'sort' determines the sort-order in relation to the visibility. Default the order in which activities are made visible is in the order specified in the activity-selection structure. When the value is set to 'visibility-order', activities are presented in the order they where made visible (this imitates a kind of inbox: new activities come available over time).
activity-selection-ref	EL	
activity-sequence	EL	A sequence is a structure of activities which must be completed in the specified order. When entering the sequence for the first time, a user can access the first activity in the structure. When the first activity is completed, the second activity is made visible at runtime, etc. for the whole structure. The activity-structure is completed when the last activity in the structure is completed. Sequences may be nested with other sequences or selections.
activity-sequence-ref	EL	
activity-structure-ref	EL	Reference to an activity-structure.
activity-title	EL	
and	EL	The element is an operand in an expression. This element determines whether two expressions both succeed. For example, it tests whether two properties both have a particular value.
and-answer	EL	Adds additional conditions (in multiple response formats).
answer-choice	EL	Answer-choice specifies a response option for a user in a test-item. Depending on the item-format, the answer-choice can contain an correct or an incorrect answer. This is set with the score attribute. For more information see item-format. The attribute 'score'. When this value isn't set, the interaction is just for data collection. In test situations, one or more items is correct and the others are incorrect. This attribute is used to identify correct or incorrect responses. In this case the score-value is set to true or false. The attribute isvisible. value: true (default), then the response is rendered. value: false, then the response is hidden with a show/hide control (a show/hide control is e.g. the + and - control in the windows explorer to open/close folders). This is used for question-answer item-formats, where the answer is hidden. In this case there is a standard title to be rendered in the interface (implementation dependet, e.g. '+ answer'). When a series of faq items is specified in a questionnaire object, the visibility doesn't have to be set on each answer-choice, but can be set in the context of the questionnaire object for all test-items (it overrules). The text is provided in ANY schema. In practice this means: xhtml.
answer-pair	EL	Needed to specify the item-pairs in a matching question. ##not needed for eml 1.1

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
answer-pattern	EL	compliance. A pattern to match open, constructed response questions. ##not needed for eml 1.1 compliance
answer-property	EL	Refers to property which value is set to the response-value of the user. Depending on the item-format the value is different. When answer-choices and answer-pairs are selected, the identifier of the response is stored (in string). When answer-patterns are selected, the concrete answer is stored (in text). With multiple response the answer identifiers are stored in a comma separated list.
calculate	NG	
calculate	EL	This is the container for the elements to perform calculations. This container is also used in expressions. If refers to a calculation-schema which evaluation results in exactly one value as a result from the calculation. The calculation-schema is kept as a separate schema with a different namespace (e.g. http://eml.ou.nl/eml11/expressions (calculations are part of expressions). In theory different calculation schemas may be used. However it is preferred to use the calculation-schema provided with eml. In authoring environments it is expected to be integrated at this place of the schema.
change-property-value	EL	This element is used to change values of properties after an event (e.g. completion of something). E.g. When the activity is completed, a property value may be changed to reflect this fact. In the dossier also an automated record of completed activities is kept, so it isn't necessary to record the completion as such, but to register (or change) other things.
class	EL	Inherited from HTML (related to CSS). In EML 1.0 this was called 'content-type'. It is used to identify classes of common objects in order to manipulate them once. A class attribute contains a CDATA string. Just as in HTML more than one class may be specified in one CDATA string, each separated with a blank space. The priority order for classes is the same as specified in the CSS specification (see www.w3.org/style/css). Note that the classes in eml can be used for style sheet like functions (e.g. set visibility), but they can also have a semantic classification purpose (just as in HTML) not connected to style sheets or automated processing at all.
complete	EL	The element is an operand in an expression. It evaluates to true when the referenced element is completed. Referenced elements can be: unit-of-learning-packages, learning-activities, support-activities, role-part's, act's or play's.
complete-act	EL	This container has elements to specify when an act is completed. When this element doesn't occur, the completed is set to 'unlimited'. Once an act is completed it stays completed even if the complete condition evaluates to false in a later stage. (completed = completed OR (completed condition) with completed being initially false). It is recommended that all references to properties only refer to one of the following types: - loc-property - locrole-property - glob-property In very specific cases references to personal (local or global) properties is usefull in this place.
complete-activity	EL	This container has elements to specify when an activity is completed. When this element doesn't occur, the completed is set to 'unlimited'. In EML 1.0 this element was called 'completed'.
complete-play	EL	This container has elements to specify when a play is completed. When this element doesn't occur, the completed is set to 'unlimited'. It is recommended that all references to properties only refer to one of the following types: - loc-property - locrole-property - glob-property In very specific cases references to personal (local or global) properties is usefull in this place.
complete-unit-of-learning	EL	This container has elements to specify when a unit-of-learning is completed. When this element doesn't occur, the completed is set to 'unlimited'. It is recommended that all references to properties only refer to one of the following types: - loc-property - locrole-property - glob-property In very specific cases references to personal (local or global) properties is usefull in this place.
components	EL	Is a container for the definition of the components which are used in the method.
conditions	EL	Conditions are used to personalize the presentation of the unit-of-learning. All conditions are pre-conditions and must be evaluated: - when entering the unit of learning (new session); - every time when the value of a property has been changed. This applies only to the following properties: a) properties where the person has access to in the context of the unit of learning, and b) the property has to be evaluated in one of the expressions in the unit of learning. These properties include properties, which are available in the expression, but are set automatically (e.g. time-unit-of-learning-started). An action is performed (fired) according to the success (true) or failure (false) of the condition. The action is to show, hide, change-property-value or notify a role. The show and hide actions set the visibility attribute (isvisible) of different objects: activities, environments, items, play's, activity-structures, units-of-learning and different classes of objects (set with the 'class' attribute).
conference	EL	In EML 1.0 this was three elements: announcement-object, asynchronous-conference, synchronous-conference. In the new conference element the structure of the three elements are harmonized and integrated. The distinction between the three categories are now specified with the attribute 'conference-type'. The elements participant, observer, conference-manager, moderator facilitate the setting of the user rights in the conferences. It depends on the implementation how this is managed: 1. when the conference system is an integral part of the runtime it is expected to be set automatically; 2. when the conference is external the user-rights can be set manually by the conference manager. The conference managers, must be able to get a list from the runtime agent about which conferences of what type, for what users with what rights must be set. 3. the latter can also be implemented by a developing a legacy interface to the rights management system of the conferencing system. In all instances the runtime system must be able to provide this information in a structured way. The item element refers to the resource where the conferencing system is to be found or identified. External conferencing systems can be of any kind accessible through the internet (resource type is webcontent). Examples: netmeeting, placeware (synchronous), first-class, lotus notes, news groups (asynchronous). An announcement object sets the rights: creator of announcement = participant. Reader of announcements = observer.
conference-information	EL	This element refers to resource(s) of type webcontent or emlcontent, where the information about the conferences are to be found. This information may be of any

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
conference-manager	EL	kind. Typically it is information about the conference, passwords and timetables for synchronous conferences. The conference manager is allowed to create new sub conferences and delete conferences he/she created. The new conferences are children of the existing base conference. The conference manager may not delete the base conference. It is deleted by system management when deleting the information of the (completed) run of the unit of learning. The conference manager has all the rights of observer, participant.
creator	EL	*Dublin Core* Inherited from the Dublin Core Metadata Initiative (dublincore.org). Definition: An entity primarily responsible for making the content of the resource. Examples of a Creator include a person, an organisation, or a service. Typically, the name of a Creator should be used to indicate the entity. The role attribute determines what the role of the creator was/is. E.g.: author, contributor, etc. This role is rendered in the user-interface. When there is no role specified, the term 'creator' is used.
current-datetime	EL	The element can be used as an operand in an expression. Represents the current datetime. This is a datetime specification of the form: CCYY-MM-DDThh:mm:ss CC is the century; YY is the year (year 0000 is prohibited); MM is the month; dd is the day. T is the date/time separator. hh are the hours; mm are the minutes; ss are the seconds. (see ISO 8601)
datatype	EL	The following types are supported: boolean, integer, real, string, datetime, text, file, uri. These are also predefined in the attribute datatype. For extensions use the 'other' value and specify the content in the element self. boolean: represents binary logic, true or false (aliases: yes/no; 1/0). NB: just as any other data types, booleans can also have <no-value>. integer: is the standard mathematical concept of integer numbers, representing whole positive and negative numbers (including zero), ranging from: -9223372036854775808 to 9223372036854775807 (alias: longinteger). real: standard mathematical concept representing arbitrary precision decimal numbers, and must be capable of handling a number to 18 decimal places at least. string: represents any legal character strings. The minimal maximum number of characters is 2000. datetime: This is a datetime specification of the form: CCYY-MM-DDThh:mm:ss CC is the century; YY is the year (year 0000 is prohibited); MM is the month; dd is the day. T is the date/time separator. hh are the hours; mm are the minutes; ss are the seconds. (see ISO 8601). There is also an optional timezone separator. Partial productions of the lexical expression are not allowed. duration: specifies an amount of time: the duration of an event in relative terms (e.g the duration given the start datetime of the run of a unit-of-learning. The format - also used in the W3C XML schema specification - is: PnYnMnDTnHnMnS where: P is the designator that must always be present. nY is a variable where an integer is filled in. nY represents the number of years nM represents the number of month nD represents the number of days T is the date/time separator which must always be present when representing time. nH is the number of hours nM is the number of minutes nS is the number of seconds. Example: P2Y0M1DT20H10M55S Meaning that the duration is: 2 years and 0 month and 1 day and 20 hours and 10 minutes and 55 seconds. Limited forms of lexical production are also allowed E.g. a duration of 40 minutes is expressed: PT40M. A duration of 30 days is: P30D text: represents any legal character strings. The minimal maximum number of characters is 64000 (about 10 pages of A4 text). file: represents any binary file as datatype. The property stores this file. uri: represents a URI according to the IETF's RFC 2396 Note: according to the w3c only the word URI should be used in future and not URL or URN. (see: http://www.w3.org/TR/2001/NOTE-uri-clarification-20
date	EL	*Dublin Core* Inherited from the Dublin Core Metadata Initiative (dublincore.org). Definition: A date associated with an event in the life cycle of the resource. Typically, date will be associated with the creation or availability of the resource. Recommended best practice for encoding the date value is defined in a profile of ISO 8601 [W3CDTF] and follows the YYYY-MM-DD format.
datetime-activity-started	EL	The element can be used as an operand in an expression. It refers to: - the current run of the unit-of-learning - a learning-activity - a support-activity - an activity-structure This element represents the datetime that a person first accessed the activity-description content for an activity. This value has to be added automatically at runtime. This is a datetime specification of the form: CCYY-MM-DDThh:mm:ss CC is the century; YY is the year (year 0000 is prohibited); MM is the month; dd is the day. T is the date/time separator. hh are the hours; mm are the minutes; ss are the seconds. (see ISO 8601)
dependency	EL	*IMS CP* Inherited from IMS Content Packaging (including the attributes). So for a description and function of this element the IMS Content Packaging Specs (version 1.3).
description	EL	*Dublin Core* Inherited from the Dublin Core Metadata Initiative (dublincore.org). Definition: An account of the content of the resource. Description may include but is not limited to: an abstract, or a free-text account of the content. A description of the resource for human readers. The description rendered in the user-interface.
design	EL	This element specifies the design (alias: instructional design) of the unit of learning. Any unit-of-learning can have zero or one design. A unit-of-learning with zero designs are to support content-updates for an existing run of a unit-of-learning. For new runs a design is always expected to be present. Designs can be re-used in different units-of-learning, with different content. In an existing run of an unit-of-learning, the design may never be adapted for consistency reasons, however the href's of the resources may be adapted during the run as well as the content of the resources where the hrefs point to. For local resources which are available in the unit-of-learning package, this means that a new package may be loaded during the existing run, to update the content of the run (but not the logic, which is defined in the design). Note on authoring templates: In order to implement design templates in the authoring environment, it is expected that a meta did will describe how a specific design may or may not be adapted (e.g. an author may include extra learning activities at certain points, but not on other points). RULE: In order to distinguish between references (IDREF) within the design model and references to resources, the following rule applies: Attribute name 'ref' (IDREF) refers to an element with an identifier within the design. Example: <act-ref ref=""/> refers to an act element within design. Elements with the 'identifierref' attribute, refer to resources. Example: <item identifierref=""/>

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
divide	EL	refers to a resource. The attribute name 'uri' is used for URI's which are ID's and the attribute 'href' is used for URI's which refer to uri's as ID's. The element is an operand in an expression. This element calculates the division of the first child numerical operand and the second child numerical operand. For example, it divides the values of two properties.
element-name	EL	The name of the element, in conformance with the ISO/IEC 11179 [ISO11179] standard for the description of data elements. The name is defined as: Name - The label assigned to the data element. There is also an identifier for the data element: Identifier - The unique identifier assigned to the data element In most cases these are the same. The element-name in XML schema's is the name as in: <!ELEMENT name EMPTY>
element-title	EL	
element-value	EL	The value can be: a) a concrete value (PCDATA); b) one or more langstring's (paragraphs) c) one or more other elements. In this case the element is a container for these elements. d) external resource referred to via item.
else	EL	The property-syntax for if-then-else is not changed. So the EML1.0 manual is the reference for the if-then-else structure.
email-data	EL	This element refers to the property resources where the relevant e-mail data can be found for the connected role. This is used for send-mail purposes (as a service in the environment, or in notifications). This element has two attributes: - email-property-ref: this attribute contains a reference to the property containing the email address of the users being notified - username-property-ref: this optional attribute contains a reference to the property containing the user name of the users being notified Both properties (email, username) should be available for all persons assigned to the role and the sending party.
environment	EL	The container 'environment' is a container for resources which are used during the performance of activities. References to the environment are only done within the 'design' container (with environment-ref).
environment-ref	EL	This element refers to an environment somewhere in this package.
environment-title	EL	
environments	EL	A container for the definition of the set of environments used in this unit of learning.
existing	EL	Refers to a property already declared (e.g. in another unit-of-learning, or in the global dossier) to the knowledge of the author (see 'global-definition' what happens if the the author defines a new global property which in practice already exists). The property is referred to with href, specifying an absolute URI. NB: when validating this unit-of-learning, the URI doesn't have to be present. The declaration of the URI by an external unit-of-learning can happen at any time. So this is only under the control of the author.
expression	NG	COLOPHON This is the XML binding v. 1.1 of EML (Educational Modelling Language) v. 1.1, used by the Edubox 3.0 system, and authorised by OUNL: January, 2002. Annotation-version: 1.0 (is kept per EML version and binding).The version number is in format [binding]/[eml]/[annotation] and is: "Edubox-EML/XML binding 1.1/1.1/1.0". Author: Rob Koper (see version 1.0 for previous contributors) Contributors: Hubert Vogten, Marc Verhooren, Harrie Martens. Previous version: Edubox-EML 1.0 at http://eml.ou.nl (C) Copyright 2002. Open University of the Netherlands (OUNL). All rights reserved. The copyright owner makes no representation about the suitability of EML nor its authorised variants for any purpose. EML and all its authorised variants are provided "as is" without any expressed or implied warranty. Permission to use, copy, communicate to the public, and make available variants of EML is subject to the terms of the General Licence agreement to which any user of this DTD is required to subscribe. The terms of the General Licence can be downloaded from http://www.ou.nl/eml/licence_XML-binding.pdf CHANGE SUMMARY: - Semantic Domain model: minor changes: roles can have direct access to environments (implicit activities); Introduction of global role properties. - Information model: Main change: A two layer approach is introduced to discriminate between the design and the resources. The design doesn't include any tagged content anymore, only references to resources which contain the content. In order to support dynamic content, the dynamic eml content elements are made global elements to be included in XHTML sources (through the use of namespaces). Slight changes to cardinality of some elements and for some elements more consistent (and shorter) names. - Binding: the binding is an XML dtd, it has some improvements based on newer W3C XML related specifications. The dtd is annotated. The annotations describe the element or attribute lists that are stated after the annotation. PUBLIC IDENTIFIER The public identifier of this DTD is "-//OUNL/DTD Edubox-EML/XML binding 1.1/1.1/EN" targetnamespace="http://eml.ou.nl/eml11 TERMINOLOGY (in conformance with W3C definitions) ACTIVITY: Any assignment for a task to be performed by a role. There are different types of activity: 1) learning activities are activities defining the learning tasks, 2) support activities are activities describing the task to support the learning activities or the support activities of other roles. ACTIVITIES: A collective name used to refer to entities which have an activity character: learning-activities, support-activities, activity-structures, units-of-learning and environments which have an implicit activity (that is, an activity which is not specified). ACTIVITY-STRUCTURE: Any structured sequence or selection of two or more activities which could be assigned to be performed by a role. ANNOUNCEMENT CONFERENCE: An announcement is a message send to users to inform them about new events or relevant information. Announcements are declared in the environment/service/conference object with the conference-type set to 'announcement'. ASYNCHRONOUS CONFERENCE: Asynchronous conferences are group messaging systems which uses a store (inbox) for incoming messages. These are normally ordered in (nested) topics (conferences). The most primitive asynchronous conferencing system is internet news (ntp). ATTRIBUTE: An attribute is a parameter to an element declared in the DTD. An attribute's type and value range, including a possible default value, are defined in the DTD. DTD: A DTD, or document type definition, is a collection of XML declarations that, as a collection, defines the legal structure, elements, and attributes that are available for use in a document that complies to the DTD. DOCUMENT: A document is a stream of data that, after being combined with any other streams it references, is structured such that it holds information contained within elements that are organized as defined in the associated DTD. EML: Educational Modelling Language. The open specification

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
		<p>for the definition of units of learning, developed by the Open University of the Netherlands. EML DOCUMENT: An EML document is a document that is valid to the EML DTD contained in this text file. EMLCONTENT: Is a data type for a specific resource: it contains content which is tagged according to one of the XHTML dtds (preferably xhtml-strict), with included global EML elements. The XHTML may include extension modules (like MathML, Vectorgraphics and SMIL), according to the W3C specifications (included or imported in the schema using namespaces). emlcontent must be well-formed XML. ELEMENT: An element is a document structuring unit declared in the DTD. The element's content model is defined in the DTD, and additional semantics may be defined in the prose description of the element. FACILITIES: Functionality includes elements, attributes, and the semantics associated with those elements and attributes. An implementation supporting that functionality is said to provide the necessary facilities. IMPLEMENTATION: An implementation is a system that provides collection of facilities and services that supports this specification. DESIGN: a design is a part of the unit of learning which describes the learning process within a unit of learning, based on a pedagogical model. The same design can be used in different units of learning, referring to different resources. Also called: instructional design. The design forms the core structure within EML. LEARNING OBJECT: According to IEEE LTSC (2000), a learning object is any entity, digital or non-digital, that can be used, re-used, or referenced during technology-supported learning. In EML any object, which is dependent on a resource is considered to be a learning object. A fundamental idea is that a learning object can stand on its own and may be re-used. PARSING: Parsing is the act whereby a document is scanned, and the information contained within the document is filtered into the context of the elements in which the information is structured.</p> <p>PHYSICALOBJECT: Is a specific resource: any digital or non-digital resource which is not accessible throughor cannot be launched by a web browser. Only information about the object can be provided (e.g as emlcontent or webcontent). Example: a person, a book, a CD, a laboratory. PROPERTY: A property is a variable in the dossier of a person or a role. Properties may have a local or global scope relative to the run of a unit-of-learning. RENDERING: Rendering is the act whereby the information in a document is presented. This presentation is done in the form most appropriate to the environment (e.g. aurally, visually, in print).</p> <p>ROLE: represents a group of persons who are collectively assigned to the same activities. Roles are divided in learner-roles and staff-roles. Roles are bound to concrete persons in runtime. There are local and global roles. Global roles are defined by an organization and are referred to in the unit of learning with an href (absolute URI). Local roles are defined in the unit of learning with an identifier only. It is not possible to declare global roles in a unit of learning. This is just an organizational issue and is nothing more or less than providing absolute URI's for roles. RUNTIME: synonym of 'user agent'. RUN OF UNIT OF LEARNING: A unit-of-learning describes a class of possible instances. These instances of a unit of learning are called a 'run'. In a run concrete persons are bound to the roles defined in the unit of learning and a concrete start date of the learning process is defined. The same unit of learning with the same identifier can have an unlimited number of runs. When having the same identifier it is expected to have exactly the same structure and content. The identifier of the units of learning discriminates between versions of content and structure (design). SYNCHRONOUS CONFERENCE: Synchronous conferences are group communication systems which which enables groups to communicate and work with each other in real time. Mostly through a variety of media, but the most primitive once use one media type (e.g. chat and telephone conferences). More complex systems combine synchronous and asynchronous conferences. These are also classified here as synchronous conferencing systems. UNIT-OF-LEARNING: the root element in this dtd, describing a defined (marked out) piece of education, which as itself acts as a gestalt: its containing parts that themselves doesn't represent education perse. Units-of-learning can refer to other units-of-learning. Examples: a course, a curriculum, a workshop, a lesson, a practical, etcetera. It can be of any learning-time duration: from several minutes to several years.</p> <p>UNIT-OF-LEARNING-PACKAGE: Is a container that holds exactly one unit-of-learning together with all the physical resources which are locally referred to in the unit of learning (but not the absolute/global referred resources). URI: Unique Resource Identifier. Specification from IETF, annotated by W3C (see references). In this DTD URI's are used in the meaning of the W3C annotation. This annotation doesn't make a strict distinction between URL's and URN's. Every URI can be a URL and a URN depending on implementation specific conventions. URI's can be absolute or relative. Absolute URI's are global, relative URI's are local. For resources with local URI's it is expected that the resource is available in the unit-of-learning package when the resource is dependent on one or more files. USER AGENT: A user agent is an implementation that retrieves and processes EML documents. It is identical to a runtime system. It is indifferent where processing takes place: at the client or server side. VALIDATION: Validation is a process whereby documents are verified against the associated DTD, ensuring that the structure, use of elements, and the use of attributes are consistent with the definitions in the DTD. WEBCONTENT: Is a data type for a specific resource: any content which could be hosted in, or launched within a webbrowser, like html, xml, flash, applets, text processor/spreadsheet files, etcetera. Whether files are launched in the browser which cannot be hosted in the browser, depends on the user client. etcetera. Webcontent doesn't necessarily have to be well-formed XML (e.g. html isn't). WELL-FORMED: A document is well-formed when it is structured according to the rules defined in Section 2.1 of the XML 1.0 Recommendation (http://www.w3.org/TR/xhtml1/#sec-well-formed). Basically, this definition states that elements, delimited by their start and end tags, are nested properly within one another. REFERENCES - IETF http://www.ietf.org (for internet interoperability specifications) relevant specifications: URI, ftp, news, smtp, http. - W3C http://www.w3c.org (consortium for web related interoperability specifications) relevant specifications: HTML, XHTML, XML 1.0, XML schema, XML namespaces, XSLT. - Dublin Core http://dublincore.org (for general metadata specifications) relevant specification is the Dublin Core Metadata. - IEEE LTSC ltsc.ieee.org (for learning technology specifications) relevant specification: IEEE</p>

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
		<p>LTSC LOM (Learning Object Metadata). - IMS http://www.imsproject.org (consortium for interoperability of learning management systems) relevant specifications: IMS content packaging, IMS learner information packaging. - EML http://eml.ou.nl (laboratory for learning technology specifications) relevant specifications: EML 1.0 and Edubox-EML 1.0. USER AGENT COMPLIANCE EML expects runtime behaviour from a user agent: 0. A user agent is EML 1.1 compliant when it has facilities for all of the elements and attributes in this dtd (or the equivalent XML Schema) in conformance with the descriptions contained within this dtd, except those which are tagged '#not needed for eml 1.1 compliancy'. These elements are preferably supported by the runtime, but not necessarily. When an implementer comes across inconsistent definitions to his or her opinion, he or she is not free to make an own interpretation, but has to raise the issue to the EML owners (OUNL), which will make a decision and updates the dtd's and annotations when needed. This is needed to guarantee comparable interpretation of the same EML documents in different user agents. 1. A unit-of-learning-package has all the files needed to create one or more runs from this unit-of-learning. The URI of a unit-of-learning identifies the package uniquely, including the update versioning. During the run of a unit-of-learning-package the design substructure may not be updated, but the organization structure and the physical local files delivered in the package may be updated without affecting the run status for users. When a new version of the organization, the local files and/or the design has been created, a new URI has to be created for the unit-of-learning-package. Depending on the implementation several types of information can be stored in the URI (e.g. identifier+type+version). During the run, the new package with the new URI can be published over the run (updating resources and files) or a new run can be created. Facilities to import, publish, set startdates, manage users in roles, update existing runs and create new runs are expected to be provided with the runtime system. 2. When interpreting EML, the runtime reads the unit-of-learning/design /method/play element structure (called 'play'). When more than one play is specified these are interpreted concurrently. A play has one or more acts and an act has one or more role-parts. At every level there are explicit rules specified how an role-part, act, play and unit-of-learning is completed. It is expected that the runtime keeps record of the completion status of these different entities. The completion status is retrieved by some EML constructs. The role-part completion has to be derived from activity completions or unit-of-learning completion (when the role-part refers to a unit-of-learning). When no explicit completion rule is specified the completion is set to unlimited, meaning that it is always completed. 3. Per play the acts are interpreted in the order specified. Only one act per play can have the focus at any time, starting with the first act in the play. When the act is completed, the next act gets the focus, until all the acts in the play are completed. 4. The role-parts specify which roles should be able to access what activities. When there is more then one role-part in a play, the role-parts are accessible concurrently by the different role. 5. Every user that is in runtime bound to a role should have access to the activities that are made accessible and visible for that role. Users can be bound to roles at runtime (new users added, existing users deleted). 6. When a role is tagged to allow for the creation of new roles, the visibility rules and the users for the parent are applied to the children. Within these rules the creator of the new role is allowed to regroup the existing users in the parent role over the newly created roles. 7. Activities and environment objects that are once shown to a user cannot be made hidden (when shown they are 'owned' by the user. In the user-interface a control could be available for users to hide it). This also means that, when a new act has the focus, the activities which were shown in the previous act stays accessible and visible. However, a mechanism must be implemented to show the user which act has the focus and which acts are history. 8. Conditions work within the scope of an act that has the focus, plus the history acts. Conditions can never make activities visible, which are not specified in one of the role-parts of the current act in the current plays. So in priority the acts have a higher priority then the conditions. 9. Conditions are evaluated, any time the user accesses trees or content, when the system has an online connection. But only those conditions have to be evaluated which effect the selected content or trees. Also only those conditions have to be evaluated which refer to properties in the IF expression, which values have been changed after the last time the same property value was accessed by the user. 10. Notifications have a higher priority than acts and acts have an higher priority than conditions. When a notification is send, the connected activity is always made visible to the users in the role selected with the notification. When there is a role-part available in the act who has the focus, this activity is made visible within the structure (tree) of the current act. When the activity is not specified in one of the current roles or in an history act, the activity is added to the tree as a separate node directly below or above the activities from the current act. The user is expected to be alerted when a new notification has arrived. 11. When there are conflicting show and hide conditions, the rule is that show is of a higher priority then hide (and will overrule the hide). 12. The invisible attribute on elements are interpreted as the initial visibility value. Conditions can change this value to true (when show applies) or false (when hide applies). 13. Metadata of the unit-of-learning are always accessible in the user-interface. In the user-interface specification a format for all the elements has to be specified. When not specified it is shown in the format [element-name]: [element-value]. 14. Metadata of other elements can be made accessible and visible, depending on the user-interface design (implementation dependent). 15. The uol-title is always made visible somewhere in the user interface and can never be changed by templates. 16. Other *-title elements can be rendered in the interface when needed in the implementation. Templates can overrule the existing value. 17. The unit-of-learning/design/learning-objectives/item(s) and ./prerequisites/items(s) must be accessible for all the roles, at all times in the user-interface. They may require a user-action (e.g. opening a menu, clicking a button or link). They must semantically be presented as learning objectives respectively prerequisites. 18. A user must always be aware in which role he/she is in. When a user is assigned to more than one role, he/she must be able to see to which roles he/she is assigned and be able to switch roles at any time. The information-for-role/item(s) must always be assessable with the roles for a user. 19. The user-agent is expected to</p>

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
		<p>keep record of property-values and property-definitions for users and roles in a so-called 'dossier'. There are several types of properties. Local properties are stored with a scope local to the run of a unit of learning. They are defined and used in the unit-of-learning. Global properties are accessible outside the context of a unit of learning (e.g. by more than one unit of learning). They can be defined in one unit of learning and used in another one. In EML1.1 global properties can be defined. Runtimes are expected to control whether a defined global property URI already exists or not. Global properties - once defined - may never change definition. So when the property already exists the definition is ignored. Personal properties are owned by a person (local or global) and role properties are owned by a role (local or global). Dossier properties are defined and or declared (for already defined global properties) under design/roles/properties and operated upon with property-operation elements (view-property, set-property, conditions, etc.). When a property value changes a notification may be send to a role (depending on the eml declaration). User-agents are expected to operate on properties in a secure way and with a maximum performance (to be detailed by the implementer). 20. Learning-activities and support-activities are rendered in the user-interface in such a way that a user always knows which learning activity it is (by rendering the activity-title), where it fits in the sequence or selection of a series of activities, what the activity-description is, which learning-objectives and prerequisites are connected, which environment (and content of the environment) is connected, how to complete the activity (including controls when needed) and have access to the feedback-description after having completed the item (when available). 21. When activity-structures are presented to the user, the user-agent must interpret an activity-sequence in such a way that: the content of the 'information' element is made visible to the users, the connected environment (and its content) is available for the user and the activities are made accessible one by one in the specified sequence. The next comes available when the previous one is completed, even if a condition states otherwise. Sequences have a higher priority than conditions. An activity-selection is presented in such a way that the user sees the 'information', the connected environment and can access all the containing activities including a cue to inform the user when restrictions are set on the maximum number of items to be selected. The activity-sequence is completed when the last activity in the row is completed (including sub-sequences or sub-selections). Activity-selections are completed if the user has completed the number of items that should be completed (when the number-to-select isn't set, all the activities in the selection must be completed). In all situations a user must see what type of activity (learning, support, substructure or unit-of-learning) he/she can access at the moment he/she sees the link to the activity. 22. Support activities can be recurrent for every user in the supported role (when the role-ref is specified), the user-interface must represent this fact and makes explicit at any time for which user or group of users the support-activity is performed. The user may select one or more of the users in the supported role. 23. Environments are connected to activities, activity-structures or roles (in a role-part). When an activity-description is visible, always the connected environment (including the content structure of the environment) must be made visible. It must be possible to access and see the activity-description and the content of one of the objects or services within the environment at the same time. 24. Every object or service in the environment has its own design requirements. The implementer is free to implement the objects and services, taking care of the representation of all the elements and the functionality they represent as described with the elements in this dtd. 25. Resources are not shown directly in the user-interface. They define the collection of resources where the 'item' elements in the design refer to. Locally defined resources and files (by using relative URI's in the href) can be expected to be present in the unit-of-learning-package. Absolute URI's refer to resources outside of the package. A change of an external resource doesn't affect the unit-of-learning-package (the package isn't aware of the presence and change of external resources). 26. The element 'item' occurs at different places in the dtd. The items are referring to resources that contains webcontent or emlcontent (see terminology). This content must be rendered in the user-interface in such a way that the content itself is visible, including the item structure when there is more than one item. Also the item titles must be visualized. Providing a table of content and sectioned text can do the latter. The nesting level of the items must be made visible to the user (e.g. by providing nested headings). More concrete: emlcontent must always be directly rendered to the user-interface. At least the content of the first item/resource must be visible to the user. The user is not asked to perform a user-action (e.g. clicking a link) before this content becomes visible. This also applies for webcontent hosted in the webbrowser (html, xhtml, xml) and launched in the browser window (e.g. flash, applets, real media, etc.), but not for for webcontent which can not be hosted in the client, but is launched in a seperate window. For the latter implementation specific solutions may be provided.</p>
expression	EL	A global eml expression should be inserted here. Example: <users-in-role> <role-ref ref="student"/> <expression> <expression-schema xmlns="http://eml.ou.nl/eml11/expression"> <is.../> </expression-schema> </expression> </users-in-role>
expression-schema	EL	The container for the expression commands for use within the design element. It is used in the context of the elements: <expression/> and <if/>. It can be used global.
feedback	EL	The feedback given to a user, matching the previous condition (e.g. completion or certain responses from users). The text is provided in ANY schema. In practice this means: xhtml.
feedback-description	EL	The element points to a resource of type webcontent or emlcontent, where the feedback description can be found. After completion of an activity this text becomes visible.
file	EL	*IMS CP* Inherited from IMS Content Packaging (including the attributes). There is a change in occurrence. In IMSCP the file occurs 1 or more times. In the unit-of-learning it occurs zero or more times (in order to allow shorter descriptions of resources which are only dependent on URI's and not on files). So for a description and function of this element the IMS Content Packaging Specs (version 1.3).
first-access	EL	The datetime of first access to the current run of the unit of learning. (datatype=datetime).
first-category	EL	This element is used in matching questions, where pairs of items has to be

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
formal-title	EL	matched. The items are ordered in pairs of correct answers (and randomized for the presentation). This is the first-category of a pair (see second-category for the second item in the pair). ##not needed for eml 1.1 compliancy *Different metadata vocabularies* Definition: The formal title is the official full name of the resource, including subtitles. It has a mixed content model. When langstring is used, a maximum of three langstrings may be used. The first langstring describing the title, the second and third langstring are for the subtitle(s) (and may be formatted differently).
format	EL	*Dublin Core* This is a container for different elements, used to guide the publishing process and media selection. The element is inherited from the Dublin Core Metadata Specification, but adapted. The Dublin Core Specification is: <format>e.g. text/html</format> Here the same specification is done with: <format><source-format>e.g. text/html</source-format></format>
glob-property	EL	A global property is a global unique property, which stores one value, independent of user, units-of-learning and role. The identifier can be used to refer to the property in this unit-of-learning-package. Operations can refer to this identifier to operate on the value. ##not needed for eml 1.1 compliancy
global-definition	EL	NB: This element is only included for implementations which lack a tool to define global properties: it is an optional construction in EML1.1 and can be removed in systems which only refer to existing (elsewhere) decalared global properties. To force consistency, the next rule applies: RULE: Once a global property has been defined in whatever context, it can never be changed! This is also true for re-publications of the same unit-of-learning. So the definition is only used when the URI (href) doesn't exist yet. Otherwise it is ignored. The URI must be an identifier which identifies the global property global unique. It must be an absolute URI. When the URI is an URL, the URI doesn't need to point to the property location, but can be interpreted as an identifier.
global-elements	EL	The global EML elements can be inserted in other vocabularies, like xhtml (it must be well-formed XML). Use the namespace xmlns="http://eml.ou.nl/eml11", which is already predefined in all elements. Examples of use in xhtml: <html xmlns:eml="http://eml.ou.nl/eml11"> <head><title>example emlcontent</title></head> <body><p>This is an example of the use of emlcontent. This text is xhtml. The eml properties are included in a table.</p> <p>Provide your email address: <eml:set-property href="email" max-transactions="1"/></p> <table><tr><td><i>learning style</i></td><td><i>e-mail address</i></td></tr><tr><td><eml:view-property href="learning-style"/> </td><td><eml:view-property href="email"/> </td></tr></table></body> </html> In the example the namespace is declared with the prefix eml:. Of course the namespace could also have been defined in the element itself (xmlns="http://eml.ou.nl/eml11").
globpers-property	EL	global personal property, alias: portfolio-property. This property can have a different value for every user, independent of the different runs of units of learning (its specifies the portfolio of the user). The property is owned by the person. The identifier can be used to refer to the property in this unit-of-learning-package. Operations can refer to this identifier to operate on the value.
greater-than	EL	The element is an operand in an expression. This element determines the relationship between two operands. It determines whether the first child element represents a value greater than the second child element.
group-title	EL	NB: Hide learning-activity-ref, activity-structure-ref and support-activity-ref are included for backwards compatibility, but they are obsolete in EML 1.1 The general rule is: activities that are shown to a user can never be hidden again. In the show-model however they are present! This container contains what must be made hidden when the condition (if) is true. This effects the 'invisible' status of the entity (set to false).
hide	EL	
hint	EL	When a hint is specified, the user gets a anchor named 'hint' rendered with the interaction. When the anchor is selected, the text of the hint is shown to the user. RULE: there is always a user request needed to get to the content of the hint. The text is provided in ANY schema. In practice this means: xhtml.
if	EL	If refers to an expression-schema which evaluation results in the value: true or false. The expression-schema is kept as a seperate schema with a different namespace (e.g. http://eml.ou.nl/eml11/expressions. In theory different expression schema's may be used. However it is preferred to use the expression (and calculation) schema provided with eml). In authoring environments it is expected to be integrated at this place of the schema. When the expression resolves to 'true', the 'then' rule fires. When it resolves to 'false' the 'else' rule fires when it is present (otherwise nothing happens in this rule).
included-schemas	EL	These schemas are included in the unit-of-learning, but are kept separate for documentation, etc.
index	EL	This element is a wrapper for indexing aspects, used to set up a search service. The index is made in the background (not visible to users). The visibility is determined with the search element. The functionality of the index is dependent on the search element: - when search is free-text-search, then the index is made on the resource pointed at in the index (i.e. the underlying html texts). - when search is index-with/without-reference, than only an index is made of the elements which share the same class, including underlying items. This has the form of a table of content.
index-class	EL	This element selects the class to make the index on. Only one class item per element may be provided. Example: <index-class index-class="problemdescription"/> makes an index on all objects in the design which have one of the strings in the class attribute assigned to "problemdescription".
index-element	EL	This element selects the element to make the index on. The index attribute specifies the element to index-on (only one reference per index-element). This indexing only makes sense when there is a structure to index on, or underlying text to index for free-text-search.
index-search	EL	index-search
information	EL	The information element specifies the resource(s) of type webcontent or emlcontent, where the information can be found about the activity-sequence or the activity-selection. This information is ABOUT the structure and the selection and is no part of the activities itself. In user-interfaces it must be made visible in relationship with the sequence or selection (e.g. at the top of the

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
information-for-role	EL	sequence/selection tree). Refers to a resource of type webcontent or emlcontent with information (mostly text). This information is visible when roles are planned or selected by users. When the role is global, the information-for-role is local for the unit-of-learning (so the same global role can have different informations in different units of learning). The item of course can also point to a global URL, when an author or institute wants the same information for the global roles.
initial-value	EL	The initial value of the property is set to the value of this element when specified. When this value isn't specified the initial value is '<no value>'
intended-user-agent	EL	*new in EML* This element specifies what user-agents the developers had in mind when developing this resource. The user-agents are specified in priority order: - the first specified user-agent is the primary user-agent; - the second user-agent specified is the secondary user agent. - etcetera.
is	EL	The element is an operand in an expression. This element determines whether two operands are equal. For example, it tests whether a property has a particular value.
is-member-of-role	EL	This element is true when the person accessing this element is a member of the specified role.
is-not	EL	The element is an operand in an expression. This element determines whether two operands are not equal. For example, it tests whether a property does not have a particular value.
item	EL	*IMS CP* Inherited from IMS Content Packaging (including the attributes). So for a description and function of this element the IMS Content Packaging Specs (version 1.3). The identifier attribute is included to support the IMS CP specification, but is of no use in this dtd.
item-format	EL	Test-items can have several fixed formats, which are described in the testing literature. A main distinction can be made between test items based on constructed response, or test items based on selected response. In the first case learners have to produce the answer on questions, tasks or problems themselves, in the second case they have to select or manipulate answer-options provided in the test-item. Examples of the first category is: open answer questions, essay writing. Examples of the second category are multiple-choice or matching questions. Most test-items can be modelled using a generic, abstracted structure like the one provided in EML. However, the user agent has to know what the item-format is in order to be able to render the item in an appropriate way. For instance: in the item-format of true-false questions, there can be two questions which are interpreted and rendered as statements. For each item-format a separate rendering scheme has to be worked out when rendering EML. In the attribute some predefined formats are provided. For extensibility new types may be provided in the content of the object (the attribute 'other' must be provided then). Supported item formats are: 1. question-answer (alias: faq) 2. multiple-choice (alias: mc) 3. multiple-respons (alias: mr) 4. true-false (alias: tf) 5. prompt (alias: pr) For these items a definition is provided here: ad 1. question-answer format. Faq's are used for different purposes, not only for testing, but mainly for support functions. The format is: - one question - one answer-choice (isvisible set to 'true' or 'false'). When isvisible is true, the faq item is rendered in a sequence of question/answer where it is evident from the rendering what the question is and what the answer. When isvisible is false, the faq item is rendered as a link with the anchor 'answer'. The answer is only visible to the user after selection of the link (e.g. rendered as a collapse and expand control). When a collection of question-answer items is modelled for a faq, use a questionnaire-object. The visibility of the answer-choices can be set for all items in the questionnaire-object. ad 2. multiple-choice A mc test-item is defined as any item having two parts: - one question (a stem), describing a problem for a learner. - two or more (mostly four or five) answer-choices, one of which is correct (property score is set to 'correct') and the rest is wrong (score set to 'incorrect'). The student is asked to select the right
item-ref	EL	Refers to the identifier of an item in the design context.
item-selection	EL	
knowledge-object	EL	Definition: A knowledge-object is every object in an environment which is used to transfer knowledge to the learner. Examples: books, articles, CBT (interactive or linear). The object can contain resources of type 'webcontent' or of type 'emlcontent'.
knowledge-object-ref	EL	This element refers to a knowledge-object somewhere in this design. This element can have an own class, overruling the class specified in the referred object.
langstring	EL	This is identical to the XHTML <p> element. The binding comes from the IMS METADATA. The attribute xml:lang may be added to all elements according to the W3C specifications. It is specifically needed on this element, but deleted here in order to support XML schema notation. When using dtd's the xml:lang attribute can be added to the different elements.
language	EL	**Dublin Core* Inherited from the Dublin Core Metadata Initiative (dublincore.org). Definition: A language of the intellectual content of the resource. Recommended best practice for the values of the Language element is defined by RFC 1766 [RFC1766] which includes a two-letter Language Code (taken from the ISO 639 standard [ISO639]), followed optionally, by a two-letter Country Code (taken from the ISO 3166 standard [ISO3166]). For example, 'en' for English, 'fr' for French, or 'en-uk' for English used in the United Kingdom.
last-access	EL	The date and time of last access to the current run of the unit of learning is returned (datatype=datetime).
learner	EL	In every design there is at least one learner-role. Learners can be 'nested', meaning that a role may be divided in sub roles. E.g. in an educational game you can distinguish: <learner ref="student"> <title>student</title> <learner ref="chair"> <title>chair</title> </learner> </learner> .. <resources> <resource identifier="student" type="role" href="http://ou.nl/role/student.role"/> </resources>
learning-activity	EL	Definition: a learning activity is an instruction for a learner to perform an task within an environment. Examples: solve a problem, study a book, discuss a topic with peers, write a dissertation, etcetera. This is the core element in EML.
learning-activity-ref	EL	Refers to a learning-activity. The element can be used as an operand in an calculation or expression.
learning-objectives	EL	Learning-objectives are specified as a series of zero or more learning-objective elements. Every learning-objective in the root of this model is a single learning objective, pointing to the resource of type 'webcontent' where the prerequisite is

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
		described. The learning objective is rendered in the user-agent. There are two locations where learning-objectives are specified: - At the level of the unit-of-learning (in the root of design) - At the level of learning-activities (within learning-activities). The first ones are a more general description; the second ones are more concrete. When creating a template, it is advised to include only one learning-objective, referring to a resource where the learning objective is specified. This resource can then specify more than one learning objective in a free (x)html format. There are two types of learning-objectives: -1- human readable descriptions (the items point to text resources) -2- machine-readable specifications (in EML 1.0 the performance-properties). These are addressed through the href attribute of the resources pointed at. ##this latter is not supported. The learning-objectives schema's could be user-defined or fixed by an organization. In the latter case, the texts of the learning objectives are referred to (through href).
less-than	EL	The element is an operand in an expression. This element determines the relationship between two operands. It determines whether the first child element represents a value lower than the second child element.
loc-property	EL	Local property, alias: run-property Local property. This property has the same value in a run for every user. The property is owned by the run of the unit-of-learning. The identifier can be used to refer to the property in this unit-of-learning-package. Operations can refer to this identifier to operate on the value.
locpers-property	EL	Local personal property. This property can have a different value for every user in all the roles for a run of this unit-of-learning. The owner of the property is a person in a run. The scope of this property is the run. The identifier can be used to refer to the property in this unit-of-learning-package. Operations can refer to this identifier to operate on the value.
locrole-property	EL	alias: group-property Local role property. This property has the same value for every user in the specified role during the run of a unit-of-learning. The property is owned by the role in the run of the unit-of-learning. The identifier can be used to refer to the property in this unit-of-learning-package. Operations can refer to this identifier to operate on the value.
meta	EL	This element is used to model metadata schema's, like the IEEE LTSC LOM or the Dublin Core metadata schema. The functionality is that a user agent for human reading may render the title and the value of an element. When the invisible attribute is set to 'true' (default), then the data are rendered to the user by the user agent. What is presented is: a) When the title is available: title and value is presented b) When the title is not available: element-name and value are presented.
metadata	EL	This is the EML metadata model. Most of the elements are derived from the Dublin Core Metadata (http://dublincore.org). This set of elements is selected to enable automated processing. There is a core set of elements in the metadata which are processed differently, e.g. the title is always rendered in the user interface at a special place; rights are always accessible from the interface, etc. Metadata which are only shown (or even hidden) to the users are all modelled in the element 'meta'. Elements which have only the function of rendering information to users, or facilitate search processes are all contained in the 'meta' element. When a user scheme has a title, like lom:general/lom:title. Authors can decide what schema's to use (LOM, Dublin Core or proprietary). The complete LOM set can be modelled in the 'meta' element. However it would be advised to map the common elements to the base set (e.g. lom/general/title can be mapped to unit-of-learning/metadata/title (e.g. using XSLT). The metadata elements selected are a general set, used for all the different elements in the unit-of-learning model. This means that some elements are not useful in every context. E.g. the element 'format' makes no sense in the context of a role specification, but does in the context of a resource definition. The decision to use certain elements or not is up to the user. A change from the previous EML10 dtd is that this dtd specifies the metadata in a fixed order (so there is more control over the occurrences). User-agents can transform this order in any other order, but must be explicit on this point: they cannot expect that the author has created the right order of elements). The metadata of a unit-of-learning are fixed in any publication.
metadata-schema	EL	
metadataschema	NG	
method	EL	The method is a container for the definition of the dynamics of the learning process. It consists of two major parts: the play (which could be interpreted as the runscript for the unit of learning) and the conditions (which are used for personalization by showing, hiding, notifying and changing things dependent on properties).
moderator	EL	Specifies who the moderators are in the conference. Moderators are persons who have the right to control and change the contributions of participants before they are made visible to other participants or observers. When a moderator is specified it means that participants may not contribute directly to the conference, but via the moderator. The moderator can reject, adapt or accept a proposed contribution of a participant. In all cases the contributor is notified of the judgement of the moderator. When there are more users in the role connected to the moderator, all have the same rights, but always the first one who did the job decides. This element has an effect on the setting of the user rights in the conference.
monitor	EL	The monitor service provides a facility for users to look at their own properties or that of others in a structured way. The idea is that the author defines emlcontent (e.g. XHTML tables with global eml view-property elements) to view the properties. This emlcontent is referred to with the 'item' element. In the monitor object the author makes the forced choice to see the properties of the dossier of the user 'self' or of all the users in a certain, specified role. When self is selected it means that the all the property values viewed in the monitor service are from ones own dossier. As a consequence, every property has exactly one value. When a role is selected one is viewing properties of the dossier of all the users in the specified role. In this case one has to take care in the design of the emlcontent, because one specifies only one view-property, but the effect is recurrent for every user in the role. By convention this means that the list in the user interface must be extended automatically when parsing the content. This means: - when the view-property is in a text line without other view-properties on the same line (all

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
multiply	EL	outside tables), then a list of values is created, each separated with a linefeed and carriage return. - when there is more than one view-property in a text line (outside tables), then a list of values is created, each separated with a linefeed and carriage return and grouped per line in conformance with the grouping of the view-properties. - when the view-property is in a table, than for each user in the role a new row in the table is created. The same behaviour as for view-property is expected for the following global elements as well: -last-access -first-access -activity-progression -datetime-activity-started -view-property-group Note: In EML 1.0 the monitor object contained a fixed lay-out. This has changed because most users liked to design the monitor object themselves and 'miss used' the knowledge-object in doing so. The element is an operand in an expression. This element calculates the multiplication of two numerical operands. For example, it multiplies the values of two properties.
no-value	EL	The element is an operand in an expression. This element determines whether a particular property has a value. It succeeds if no value is specified. Note that all properties defined in the design specification may have an initial value.
not	EL	The element is an operand in an expression. This element negates the boolean value of an expression. For example, if a property has a particular value, the sub-expression succeeds; this element causes the complete expression to fail.
notification	EL	A notifications happens after an event which is known by the runtime environment. Such an event can be e.g.: the completion of an activity, an expression evaluates to true, or a property value is set. The notification makes a new learning or support activity active for a role. This notification is of the highest priority, meaning that an otherwise invisible item will be made visible and accessible to the user. Depending on the implementation an email message can be send to the user, notifying that a new activity has arrived (with a link to that activity in the message). For the mail header the subject field can be filled in to a specific value (otherwise a standard message will be send). A notification can be inserted in external vocabularies (after an event like set-property), however, then the content must be provided in the package (because it contains references to identifiers in the package). When the identifier cannot be resolved the notification is ignored (but doesn't stop the xhtml content to be presented).
number-of-items	EL	*EML new* States the restriction to the number of test-items which should be selected from the specified test-items to include in the test which is rendered for the user. When this option isn't provided, the number is unrestricted (all are provided). This selection is done when creating the run and is unchanged during the run for all persons. The selection is based on the set which is left after complying to any further restrictions (like the selection of test-item classes with another selection element).
observer	EL	Specifies who the observers are in the conference. Observers have only reading rights; they may not contribute. This element has an effect on setting the user rights in the conference.
operand or	NG EL	The element is an operand in an expression. This element determines whether one out of two expressions succeed. For example, it tests if property A or property B has a particular value.
participant	EL	Specifies who the participants are in the conference. Participants can read (listen/see) the information, and can contribute to the conference. This element has an effect on setting the user rights in the conference. At least one role must be specified to identify the participants in the conference.
play	EL	The play is the root element when interpreting the design. It represents the flow of activities during the learning process (the 'workflow' or better: the 'learningflow'). The play is modelled according to the metaphor of an theatrical performance. A play consists of a series of acts and an act consists of a series of role-part's. There is always at least one play in every design (and every unit-of-learning). In runtime the play is interpreted to show and hide activities, (other)units-of-learning, environments and resources to the users. When there is more than one play, these are interpreted concurrently and independent of each other. The same user can see the results of more than one play in the user-interface. Practical experience have shown that a lot of designs use multiple plays, to represent the flows of activities per role, e.g. a play for the learners and a play for staff. However this can only be done when the activities are independent of each other.
play-ref prerequisites	EL EL	*EML new* Refers to a play (in method/play). The prerequisites in the root describe a series of zero or more prerequisites. Every item in the root of this model is a single prerequisite, pointing to the resource of type 'webcontent' or 'emlcontent' where the prerequisite is described. The prerequisite is rendered in the user-agent. When creating a template, it is advised to include only one prerequisite, referring to a resource where the prerequisite is specified. This resource can then specify more than one prerequisite in a free (x)html format. There are two locations where the prerequisites are specified: -1- At the level of the unit-of-learning (in the root of design). -2- At the level of learning-activities (within learning-activities). The first ones are a more general description; the second ones are more concrete. The prerequisite schema's could be user-defined or fixed by an organization. In the latter case, the text of the prerequisites are referred to (through href). There are also two types of prerequisites: -1- Human readable descriptions (the items point to text resources). -2- Machine-readable specifications (in EML10 the prerequisite properties). These are addressed through the href attribute of the resources pointed at. ##the latter is not supported.
presentation-option	EL	There is a fixed set of presentation options provided in three attributes: - Feedback attribute: Default value: suppress means that feedback (in the feedback element) is suppressed when rendering. Value: 'immediately' then feedback is provided immediately after the user has provided the response. Value: 'delayed', then the feedback is given after completion of the total test (for all items in a list). - Hints attribute: default value: suppress-hints, this suppresses hints when rendering. value: provide-hints: the hints and hints anchors are available for the users. - Answers-visible: This option overrules invisible attributes set on answer-choice. This is used in the context of faq's (question-answer). When set to true (default) the answers are rendered in the user interface. When set to false, the answers are hidden under the standard link 'answer'. Only when the user selects the link the answer becomes visible. This can be implemented with a show/hide control (e.g.

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
process	EL	+ answer). For extensibility, new presentation-options may be defined. The attribute other-type can be used to specify the name of the new presentation-option. The value of the new presentation-option can be specified in the content of the element. Is a container to specify conditions for property manipulation after the test is completed. These The conditions are only applied after completion of the test. It is possible to refer to test-properties to comply to the older EML 1.0 specification.
properties	EL	Definition and declaration of new and existing properties. All properties used in the unit of learning are declared in this section. All properties can be addressed for property-operations (property-ref, view-property, view-property-group, etc.). See dossier specifications for more information about properties and operations.
property-group	EL	A definition of a group of properties which belong together (and are edited in e.g. a form). It can only contain properties of the same type. The identifier can be used to refer to the property-group in this unit-of-learning-package. Operations can refer to this identifier to operate on the value.
property-group-ref	EL	Refers to a property This can be any of the following property types: - local property - global property - local personal property - local role property - global personal property - local role property The ref refers to the property declaration in the design. The element can be used as an operand in an calculation or expression..
property-title	EL	
property-value	EL	The element can be used as an operand in an calculation or expression. This element specifies the value a property is set or compared to. This depends on the context. For instance within an If statement the property is compared to the value. In a change-property-value context, the property is set to this value. Depending on the property-type this value is of type PCDATA or langstring. Property-value's may be calculated from the values of other properties. It is also possible to take over the property value of another property (with property-ref).
publishing-format	EL	*new in EML* This element describes for which publishing format the resource was intended to be developed. That is the format that users will see. Different vocabularies are possible here. For now, use MIME types. The publishing formats are specified in priority order: - the first specified format is the primary format; - the second format specified is the secondary format - etcetera.
question	EL	A question is a resource (mostly xhtml text) displayed to the user in order to invoke a response. Depending on the item-format the question has another interpretation, e.g. a problem, a statement, a task, a cue. For example, In a true-false format, the prompt is the 'statement'. Depending on the item-format, one or more questions must be provided. The resource is provided in ANY schema. In practice this means: xhtml. Questions may be in different mediatypes, supported by webbrowsers (audio, video, etc.)
questionnaire-object	EL	Definition: questionnaire-object is a component within the environment which specifies a questionnaire. The questionnaire can be used in different functions: self-tests, survey, to set properties for personalization. The resources for services are of type: 'emlcontent'.
questionnaire-object-ref	EL	This element refers to a questionnaire-object somewhere in this package. This element can have an own class, overruling the class specified in the referred object.
randomize	EL	This element occurs in the context of an single test-item and in the questionnaire-object. When this option is selected, the test items or the answer-choices (depending on context) are randomized at the moment the run is created. During a run the order stays fixed. When the option isn't selected, the test items are provided in the order specified by the author.
resource	EL	*IMS CP* Inherited from IMS Content Packaging (including the attributes). So for a description and function of this element the IMS Content Packaging Specs(version 1.3). Different types are supported in EML 1.1 (they are provided as a fixed nametoken list at the type attribute. This is done for validation reasons, but isn't part of the IMSCP specification itself (there the type is of type PCDATA). - webcontent (see IMS spec): unparsed. The URI is interpreted as an URL. Webcontent is defined as: all content which can be hosted in, or launched by a web browser. This includes: html, xml, applets, and files that are connected to an application through their extension (e.g. *.doc launches a word processor), common web applications like flash, real media, etc. - emlcontent is xhtml mixed with eml global-elements and possible well-formed XML extension modules (e.g. mathml, smil, vectorgraphics). This is parsed element by element before it is rendered. emlcontent must be well-formed XML. The URI may be interpreted as ID's or as URL's depending on implementation. Emlcontent represents personalized content, like the content of a monitor object, or personalized knowledge-object content. In some cases the resource is a physical object, not accessible for the user agent. In that case the author can refer to a file (webcontent or emlcontent) referring to the description of the object.
resource-structure	NG	
resources	EL	*IMS CP* Inherited from IMS Content Packaging (including the attributes). So for a description and function of this element the IMS Content Packaging Specs (version 1.3).
restriction	EL	Zero or more restrictions of different type may be set on the property-values, meaning that the property-value is valid when it is of the specified data type and its value is within the specified restriction rules. Zero or more restrictions can be specified (these have the same format as specified in the W3C XML schema 1.0 specification) in the attribute: 'restriction-type'. However properties may not contain arrays (lists) of data, but can only contain a single value. So restrictions only apply to this single value. (Also the 'whitespace' restriction is not supported in eml). The restriction types are: length: constraints the length of the property value of a textual datatype (string, text or uri) in terms of the of characters that it can have. minLength: constraints the minimum number of characters that a property of textual datatype can have for its value. maxLength: constraints the maximum number of units of length that a property of textual datatype can have for its value. enumeration: constraints the value of a property to a specific value (use for value alternative lists). This was called 'value-list' in EML 1.0. maxInclusive:constraints the value of an ordered (integer, real, datetime) property to a specific inclusive upper bound. minInclusive: constraints the value of a ordered property to a

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
rights	EL	specific inclusive lower bound. maxExclusive: constrains the value of a ordered property to a specific exclusive upper bound. minExclusive: constrains the value of a ordered property to a specific exclusive lower bound. totalDigits:constrains the value of a decimal property to a specific number of digits it must contain. fractionDigits: constrains the value of a decimal property to the maximum number of digits it may have after the decimal point. pattern:constrains the literals comprising the value of a property to a pattern defined by a regular expression (##pattern is not needed for eml 1.1 compliancy). **Dublin Core* Inherited from the Dublin Core Metadata Initiative (dublincore.org). Definition: Information about rights held in and over the resource. Typically, a rights element will contain a rights management statement for the resource, or reference a service providing such information. Rights information often encompasses Intellectual Property Rights (IPR), Copyright, and various Property Rights. If the rights element is absent, no assumptions can be made about the status of these and other rights with respect to the resource.
role-part	EL	A play is modelled according to the metaphor of an theatrical performance. A play consists of a series of acts and an act consists of a series of role-part's (alias 'scene'). A role-part relates exactly one role to exactly one type of activity (including the performance of another unit-of-learning and activity-structures). Role-parts within one act, are performed concurrently. The definition of a role-part differs here somewhat from real role/parts's in a theatrical play (here the metaphor is not completely correct, like all metaphors). When an activity, item or environment's attribute invisible is set to 'false', the link in the activity-tree may be made visible when the role-part sets the activity for a role (implementation dependent) , but the content isn't accessible. First a condition has to specify that the activity is visible again (show condition).
role-part-ref	EL	*EML new* Refers to a role-part (in method/play/act/role-part).
role-ref	EL	role refers to the identifier of the resource of the role. The element can be used as an operand in an expression.
role-title	EL	
roles	EL	Roles is a container for the two general roles: learner & staff. A href can be provided when referring to a global role (e.g. a role defined by an institute). This is obligatory when specifying a global role and connected globrole-properties. Global roles are specified with the href attribute. The rest of the declaration, like information-for-role, is local. It is not possible to declare global roles in a unit of learning. This is just an organizational issue and is nothing more or less than providing absolute URI's for roles. The attribute 'identifier' on roles can be used to refer to the whole group of all roles within the learning-design (learners and staff). The attributes: - min-persons specifies the minimum number of persons which must be bound to the role in order to start a run. - max-persons specifies the maximum number of persons which can be bound to the role in order to start the run. In both cases: when the attributes are empty, there are no restrictions. - match-persons. This attribute is used when there are several sub roles (e.g. chair, secretary, member). Persons can be matched exclusively to the sub roles, meaning that a person who has the role of chair, may not be bound to one of the other roles at the same time. When it is not exclusive, persons may be bound to more than one sub role (this is the default situation). -create-new. This attribute indicates wether multiple occurrences of this role are may be created during run-time. When the attribute has the value "not-allowed" than there is always one and only one instance of the role. If the value is "allowed" a mechanism in the run-time is provided to create new instances of this role. If a new instance of a role is created, new instances for all sub-roles of that role are created as well. In every learning-design at least one learner role is specified. In institutional installations the role names are fixed. For instance in the OUNL role-identifiers for student is: 'student'. To address this, use the following declaration: This element specifies the roles distinguished in this unit of learning. NB: the role declarations don't contain properties anymore. They are specified in the method section and bound to roles through a 'role-ref'.
score-value	EL	This value refers to a property-resource. The user agent automatically sets this property-value to the score of the user-interaction when the response-score attribute on the element 'response' is set. The score is '1' when the user answered correct and '0' when the user answered incorrect. E.g. A MC question: Who's the queen of holland? a. Beatrix (correct) b. Maxima (incorrect) c. Juliana (correct) d. Wilhelmina (incorrect) When the user selects Juliana, the property score-value is set to '0'. When the user selects Beatrix, the property is set to '1'. When there are no score attributes set in the response element, the score-value is ignored (not set). The attribute 'calculate-score' regulates how the score is aggregated when more then one score-value refers to the same property. value: add: the new value is added to the existing value (this is default). value: clear: the property is set to the new value (the old value is cleared).
search	EL	This element specifies how a user can access the indexed entities. There are three possibilities: 1. the user gets a free text search dialog, where he can search the index in a free text format (this also means that the index has to be build for free text retrieval). The syntax for free text retrieval is implementation dependent, e.g. the format found in search engines like Google or Altavista. 2. the user is presented a text index (table of content) with (hyper-)linked (or on other media e.g. page numbers) references to the source. 3. the user is presented a text index (table of content) without (hyper-)linked references. This provides e.g. information about the structure of the unit of learning.
second-category	EL	This element is used in matching questions, where pairs of items has to be matched. The items are ordered in pairs of correct answers (and randomized for the presentation). This is the second-category of a pair (see first-category for the second item in the pair). ##not needed for eml 1.1 compliancy
self	EL	Means that the all the property values viewed in the monitor service are from ones own dossier. As a consequence, every property has exactly one value.
send-mail	EL	This service is used to send mail to users or to all users in a role. This is set with the attribute 'select'.
service	EL	Service is a container for a variety of services to the users (communication, index-search and monitor). The resources for services are of type: 'emlcontent'.
service-ref	EL	This element refers to a communication-service somewhere in this package. This element can have an own class, overruling the class specified in the referred object.

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
set-property	EL	With this element a specified property-value may be set by the user. It works outside of the context of a textline (e.g. outside <p>). The view attribute sets whether the value or the title+value should be delivered. The user gets a control in the user-interface to set the value of the property. The type of control is dependent on the property datatype and the restrictions. In the control the current value is shown and the datatype and restrictions are made explicit so that the user knows exactly what values are valid and which are not. This allows for client-side checking of the input (dependent on implementation this may also be dealt with at the serverside). The element refers to the property URI or identifier with a ref or href. In order to avoid confusions it is good practice to include the emlcontent with property-operations in the unit-of-learning-package. In case of personal properties, the property value of the user himself is set. Only in the context of a monitor object (with the role-ref specified) the values of all the users in the specified role can be set. The attribute max-transactions represents the number of times a property may be set by a user. Technical (upload) errors do not count as a trial, but only successful transactions. When the attribute sn't specified, the number of attempts is set to unlimited. The attribute transaction-type: future extension, e.g. for secure transactions
set-property-group	EL	With this element the values of the properties contained in a specified property-group may be set by the user. It works outside of the context of a textline (e.g. outside <p>). The view attribute determines whether the titles of the containing values should be shown. The group-title is always shown. The user gets a control in the user-interface to set the value of the property-group. The type of control per property is dependent on the property datatype and the restrictions set on the property. In the control the current value of the properties are shown and the datatype and restrictions are made explicit so that the user knows exactly what values are valid and which are not. This allows for client-side checking of the input (dependent on implementation this may also be dealt with at the serverside). All values of all properties in the group are set at by the user before updating. The transaction is always counted for the group of properties, not for single properties. The element refers to the property-group identifier or URI with a ref or href. In order to avoid confusions it is good practice to include the emlcontent with property-operations in the unit-of-learning-package. In case of personal properties, the property values of the user himself is set. Only in the context of a monitor object (with the role-ref specified) the values of all the users in the specified role can be set. The attribute max-transactions represents the number of times a property may be set by a user. Technical (upload) errors do not count as a trial, but only successful transactions. When the attribute sn't specified, the number of attempts is set to unlimited.
show	EL	This container contains what has to be shown when the condition (if) is true. This effects the 'isvisible' status of the entity (set to true).
source-format	EL	*Dublin Core (new in EML)* Inherited from Dublin Core Metadata (where it is named 'format') Definition: The physical or digital manifestation of the resource. For eml use: text/eml For html use: text/html for xhtml use: text/xhtml Typically, Format may include the media-type or dimensions of the resource. Format may be used to determine the software, hardware or other equipment needed to display or operate the resource. Examples of dimensions include size and duration. Recommended best practice is to select a value from a controlled vocabulary (for example, the list of Internet Media Types [MIME] defining computer media formats).
special	EL	A global element. Special is one of the most often used elements in EML 1.0. It identifies classes of content (formerly known as 'content-types' which can be manipulated by conditions (show or hide). In this function it is identical to the <DIV class="."/ > element in (x)html. These structures can be used interchangeable. The special element however adds some extra attributes: attribute 'position': Identifies where and how the content in the special element should be rendered, given the current flow of text. value: in-flow (default). The marked text is shown at the position where it is marked in the text flow (=current position). The fact that it is marked as special is invisible for the user. value: block-in-flow. The marked text is rendered as a text block (special background, different colouring,etc) at the current position in the flow. value: typed-block-in-flow. The marked text is rendered as a text block (special background, different colouring,etc) at the current position in the flow, but has a title which is provided in the title attribute of special. value: after-flow. The marked text is rendered at a position directly after the current flow. The fact that it is marked as special is invisible for the user. value: new-flow. The marked text is rendered as the start of a new flow (e.g. new page) in a position directly after the current flow. This fact makes the special is visible for the user. value: next-to-flow. The marked text is rendered at a position directly to the left of the current flow, where a new flow is created, in position connected to the current flow. This fact makes the special is visible for the user. Example (to be seen in the dtd source): this is the special text this is the current text and is in a new flow and stays connected to the special flow (special is marked exactly before the start of this text, before the word 'this'). attribute 'title': Used for typed-block-in-flow to provide the title of the block. In every design there are zero or more staff-members. Staff members can be 'nested', meaning that a role may be divided in sub roles. E.g. in an organization you can distinguish: <staff ref="tutor"><title>tutor</title> <staff ref="assessor"><title>assessor</title> </staff> </staff>
staff	EL	Statements are used in the item-format of true-false questions. Every statement may be true or false (which can be set with the score attribute). When the answer-options are not provided, the answer-options are automatically generated (implementation dependent text). E.g. when there are two statements, the generated answer-choices can be: a) statement I and II are both false, b) statement I and II are both true, c) statement I is true and II is false, d) statement I is false and II is true. When the answer-choices are explicitly defined, this overrules the automatic generation. The text is provided in ANY schema. In practice this means: xhtml.
statement	EL	It specifies the subject of a notification, to be presented to the notified actor when the notification is activated. E.g. in the mail-header (subject field).
subject	EL	The element is an operand in an expression. This element calculates the subtraction of two numerical operands. For example, it subtracts the values of two properties.
subtract	EL	

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
sum	EL	The element is an operand in an expression. This element calculates the sum of two or more numerical operands. For example, it adds the values of two properties.
support-activity	EL	Definition: a support activity is an instruction for a staff or learner role, to support a role (learner/staff) or users in a role in performing the activities. The support role uses an environment in order to perform the support activity. Most of the times, support activities are performed by staff members (e.g. tutors) to support learners. In some pedagogical models however learners can support learners (peer to peer teaching). It is also possible that staff members support staff members. When the optional role-ref element is set, it is expected that the support activity will act for every single user in the specified role(s). That is: the same support activity is repeated for every user in the role(s). When the role-ref is not available, the support activity is a single activity (like the learning-activity).
support-activity-ref	EL	Refers to a support-activity. The element can be used as an operand in an calculation or expression.
test	EL	Container for tests, meaning a collection of interactions. It can be used global.
test-item	EL	A test item is an item which is used in assessments. These are not secure assessments for examination purposes, but for self assessment, informal progress assessment or assessment of a certain user state (intake, preference, survey, etc.). This element is used globally to include test items into e.g. xhtml content. When it contains local (idref) references to property-resources it MUST be included into the unit-of-learning package. The resource href reference must then be relative. When used in global content (not packaged), then the property references are ignored, but the questions work.
test-item-class	EL	Specifies the restriction to the test item classes (attribute 'class' on test items) which should be selected in the test. When this element isn't specified, there are no restrictions (all classes are shown). This selection is done when creating the run and is unchanged during the run for all persons.
test-items	EL	Container for references to the test-items in a questionnaire. The testitem resources are of type 'emlcontent', more specific it is XHTML, mixed with one of the elements: 'test-item' OR 'test'. When referring to test-item, every test-item must be referenced separately with an item/resource pair. When using 'test' only one item/resource pair is needed to include a number of test-items. For the operations in the questionnaire-object, this is in both instances done on the underlying, individual test-items! E.g. when referring to two tests, each containing 10 items, then the randomize element in the questionnaire-object will randomize the 20 items (independent of the structure of the 'test').
test-property	EL	Test properties are automatically declared and set They are available for reading in the current run by referring to their identifier with property-ref & view-property referring to the identifier of the test-property. Some predefined values are included in the attribute test-property-type. Extension types could be stated in the element content (PCDATA). In that case the attribute test-property-type should be set to 'other'. Values of attribute 'test-property-type': - proportion-correct: the proportion (=percentage divided by 100 to get a proportion as a fraction of between 0 and 1). The proportion has 1 digit behind the decimal point. Valid values are e.g.: 0, 0.2, 0.9 and 1. This property has a standard automatic declaration: locpers-property: title="proportion correct of test [insert questionnaire-object/title]" datatype="real" - number-correct: Number of correct answer provided by the user at any moment during the test performance. This property has a standard automatic declaration: locpers-property: title="number correct of test [insert questionnaire-object/title]" datatype="integer" - number-incorrect: Number of incorrect answer provided by the user at any moment during the test performance. This property has a standard automatic declaration: locpers-property: title="number incorrect of test [insert questionnaire-object/title]" datatype="integer" - number-answered: Number of items answered by the user at any moment during the test performance. This property has a standard automatic declaration: locpers-property: title="number of items answered of test [insert questionnaire-object/title]" datatype="integer" - number-of-items-provided: Number of test-items provided to the user. This is a fixed value. This property has a standard automatic declaration: locpers-property: title="number of items in test [insert questionnaire-object/title]" datatype="integer" - other For extensions. The value of the extensiontype is specified in the content of the element.
testitemmodel	NG	
then	EL	The property-syntax for if-then-else is not changed. So the EML1.0 manual is the reference for the if-then-else structure.
thenmodel	NG	
time-limit	EL	The time limit specifies that it is completed when a certain amount of time have been passed, relative to the start of the run of the current unit of learning. The data type time is expressed in the 'duration' format which is also used in the W3C XML schema specification. The format is: PnYnMnDTnHnMnS where: P is the designator that must always be present, n is a variable where an integer is filled in. nY represents the number of years nM represents the number of month nD represents the number of days T is the date/time separator nH is the number of hours nM is the number of minutes nS is the number of seconds. Example: P2Y0M1DT20H10M55S Meaning that the duration is: 2 years and 0 month and 1 day and 20 hours and 10 minutes and 55 seconds. Limited forms of lexical production are also allowed E.g. a duration of 40 minutes is expressed: PT40M. A duration of 30 days is: P30D The time is always counted relative to the time when the run of the unit-of-learning has been started (see element: 'time-unit-of-learning-started'. The time-limit may be specified in a property (property-ref attribute). (of type loc-property, datatype=string, to be declared by the author). In that case an author may set controls (set-property) on this property for users to control the value of the property. When a property-ref is specified, content in the element is ignored: the property overrules. Authors have to take care that the time limits set on role-parts, acts and plays are logical. In runtime the time limit of the play overrules the time limit on act and that one on the role-parts.
title	EL	Definition: A short name given to the resource, suitable for rendering in user-agents. This title could be different from a 'formal-title' which is specified in the metadata. The title is rendered by the user agent, depending of the context where the title occurs. For instance: in the context of the metadata of the

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
tool-object	EL	unit-of-learning, the title is rendered as the title of the unit of learning. NB: in items the title comes always IN the element or item it applies to: <item><title/></item>. This applies for all the other elements: e.g. <learner><title/></learner> is the title of the learner. <learning-objective><title/></item><title/></item></learning-objective> The first title is of the learning-objective (-tree) and the second is the title of the item. Definition: a tool-object is every object in an environment, used/manipulated by the role in the performance of activities. It isn't a container for knowledge. Example: a hammer, an organizer, a general software application (productivity tools like MS-Office). The object can contain resources of type 'webcontent' or of type 'emlcontent'. Mostly it is actually a physicalobject, of which a description is provided.
tool-object-ref	EL	This element refers to a tool-object somewhere in this package. This element can have an own class, overruling the class specified in the referred object.
transaction	EL	The transaction element defines: a) the number of trials a user may perform before the answer is definite. Technical (upload) errors do not count as a trial, but only successful transactions. When the element isn't specified, the number of trials is set to unlimited. This number is set with the attribute 'max-transactions', datatype=integer. When the attribute isn't set, the maximum is unlimited. b) the transaction-type (future extension, e.g. for secure transactions).
typical-learning-time	EL	*IEEE LTSC LOM* Inherited from IEEE LTSC LOM 6.1 This is the typical learning time (study-load) for the target group the unit of learning is designed for. The learning-time is expressed in number of hours, e.g. <learning-time>6</learning-time>. In the future, this time can be used in calculations for the total sum of learning time for all the activities within a unit of learning. ##This is not supported in Edubox 2.05 and 3.0. The learning time is rendered by the user agent as information to the learner or staff.
unit-of-learning	EL	GENERAL INTRODUCTION The unit-of-learning (abbr. uol) is the root element of this dtd. A unit-of-learning is part of the unit-of-learning-package. A unit-of-learning-package consists of: 1) the unit-of-learning, which is an XML file named 'eml-unit-of-learning.xml'. The unit-of-learning consists of three subparts: metadata which describe the package; a design, which describes the learning process and the resources which are referred to in the design. 2) the physical, digital files which are distributed with the package. Actors in the learning process, dealing with the interpreted units of learning are: . Learner (the people who are supported with the unit of learning in their learning process, the key role). . Staff (the people who support learners during the learning process, e.g. tutors, assessors). Beside these primary actors which are using the units of learning, there are also other actors who are dealing with the creation and management of the units of learning itself. These roles are: . Designer (designs a unit of learning in EML, including emlcontent or additions to webcontent to produce emlcontent) . Author (authors webcontent and other non-EML resources the unit of learning is referring to) . Content manager (manages versions and learning objects, produces unit-of-learning-packages) . Legal roles (manages copy-rights) . Development manager (provides the authorization to develop or publish a unit of learning and the global URIs) All roles can be spit down further to an unlimited number of sub-roles and can have different names in different organizations. All actors expect from the e-learning system to get more effectiveness, more efficiency, more attractiveness and higher accessibility. All actors fill these aspects in from their own perspective. A learner wants more effective, efficient, attractive and accessible learning; a tutor wants to tutor in a more effective, efficient, attractive and accessible way, and so forth. The translation in general categories of requirements are as follows: An Educational Modelling Language (EML), which describes a unit of learning, must meet the following general requirements: R1. It must describe units of learning in a formal way, so that automatic processing is possible (formalisation). R2. It must be able to describe designs in the unit of learning, that are based on different theories and models of learning and instruction (pedagogical flexibility). R3. It must explicitly express the semantic meaning of the different learning objects within the context of a design within the unit of learning (semantic typed learning objects). R4. It must be able to fully describe a unit of learning, including the design, the resources which it is dependent on and the metadata for all objects. The design should define all the learning objects, the relationship between the learning objects and the workflow of all learners and staff members with the learning objects, regardless of whether these aspects are represented digital or non-digital (completeness). R5. It must describe a design of a unit of learning independent of its concrete content, so that the same design can be used in different units of learning (content independence). R6. It must describe the units of learning so that repeated execution is possible, with the same content and the same design (reproducibility). R7. It must be able to describe personalization aspects within units of learning, so that the content and activities within units of learning can be adapted based on the preferences, prior knowledge, educational needs and situational circumstances of users. In addition, control over the personalization must be able to be given, as desired, to the student, a staff member, the computer or the designer (personalization). R8. The notation of the (external) content components, where possible, must be medium neutral, so that it can be used in different publication formats, like the web, paper, e-books, mobile, etc. (medium neutrality). R9. It must provide an open and technology independent interface between educational content development and the different technical user agents which interpret the units of learning. Through this, investments in educational development will become resistant to technical changes and conversion problems (interoperability and sustainability). R10. It must fit in with current open standards and specifications (compatibility). R11. It must be possible to identify, isolate, decontextualize and exchange learning objects, and to re-use these in other contexts (reusability). R12. It must make it possible to produce, mutate, preserve, distribute and archive units of learning and all of its containing learning objects (life cycle). COMMENTS ON THE ELEMENT UNIT-OF-LEARNING The identifier attribute is used to refer locally (i.e. inside this unit-of-learning-package) to the unit-of-learning. The identifier must be unique within this EML document. The URI attribute identifies the unit-of-learning globally and be absolute. The URI is only an identifier, e.g. http://ou.nl/coursecatalog/psychology/P345216 . This URI doesn't necessarily point to a concrete resource.

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Parameter Entity Comment

Name	Type	Comment
unit-of-learning-href	EL	The element can be used as an operand in an calculation or expression. This element refers to the resource of a unit-of-learning (uol). That can be the resource of this current unit-of-learning, the resource of a unit-of-learning present in the package or a resource pointing to a unit-of-learning outside of the package (absolute URD). Formally called: unit-of-study-ref. Because -ref extensions point to elements within design, this name has changed.
uol-title	EL	
user-choice	EL	This element is used in the completed element of activities and specifies that the user may decide himself when the activity is completed. This means that a control must be available in the user-interface to set the activity status to 'completed'. A user can do this once (no undo). When he/she selected the activity to be completed, than this activity stays completed in the run.
users-in-role	EL	The element is an operand in an expression. This element refer to all the users in a certain role. It means that: for all (100%) of the users in the specified role the expression stated in this element must be true. In EML 1.0 also a proportion of users could be specified. This is deleted in EML 1.1 because the value was impossible to identify. In practice it is always: 100% of the users. Example: A learning-activity is made visible (show) when all the users in the role id='student' have completed the introductory activity: if(users-in-role(role-ref [ref='student']) ,complete(activity-ref))
value	EL	In this element an (element-) value may be specified for the newly defined meta element in the metadata. The value specified must match the value constraints of the newly defined element, but this is not controlled by the system (but by the author).
view-property	EL	With this element a specified property-value may be viewed. It works outside of the context of a textline (e.g. outside the context of a <p> element. The view attribute sets whether the value or the title+value should be delivered. It refers to the property with an ref or href. In order to avoid confusions it is good practice to include the emcontent with property-operations in the unit-of-learning-package. In case of personal properties, the property value of the user himself is returned. Only in the context of a monitor object (with the role-ref specified) the values of all the users in the specified role(s) are returned.
view-property-group	EL	With this element the values of the properties in a specified property-group may be viewed. It works outside of the context of a textline (e.g. outside <p>). The view attribute determines whether the titles of the containing values should be shown. The group-title is always shown. It refers to the property-group identifier or URI with a ref or href. In order to avoid confusions it is good practice to include the emcontent with property-operations in the unit-of-learning-package. In case of personal properties, the property value of the user himself is returned. Only in the context of a monitor object (with the role-ref specified) the values of all the users in the specified role(s) are returned.
when-answer	EL	Specifies when a feedback must be given or a property value must be set. The ref refers to the answer-choice or the answer pattern. The score refers to the fact whether the answer provided was evaluated as correct or incorrect by the system.
when-condition-true	EL	Simple expression for a condition. This condition applies to all the individual users mentioned in the containing role-ref! When the contained expression is true for all users in the specified roles, this condition is true. (NB: in EML 1.0 the element 'users-in-role' was explicitly defined in this context. A proportion of users could be identified. In EML 1.1 this proportion is always 100% of the users in the role!
when-last-act-completed	EL	This element states that a play is completed when the last act is completed.
when-play-completed	EL	This element states that an unit-of-learning is completed when the referenced play(s) is (are) completed. More than one play can be selected, meaning that all the referenced play's must be completed before the unit-of-learning is completed. When a unit-of-learning is completed this should be made aware in the runtime environment to the managers of the system.
when-property-value-is-set	EL	Simple expression, containing two child elements: a property and an optional property-value. The condition evaluates to true when: 1) the property is set to the specified property-value; 2) the property is not NULL and the property-value is omitted; NB: in EML1.1 the occurrence has been changed to 1 or more instead of 1
when-role-part-completed	EL	This element states that an act is completed when the referenced role-part(s) is(are) completed. A role part is completed when all members of the role have completed the particular role-part. If a role-part is associated with an environment, this environment is considered to be completed for all persons assigned to that role. Once a role-part is completed it stays completed even if the complete condition evaluates to false in a later stage, (completed = completed OR (completed condition) with completed being initially false) More than one role-part can be selected, meaning that all the referenced role-parts must be completed before the act is completed. NB: all role-parts references must be specified in the current act!

Attribute Comment

Name	Type	Comment
answers-visible	Name Token Group	
class	Character Data	
conference-type	Name Token Group	
create-new	Name Token Group	
datatype	Name Token Group	
email-property-ref	Identifier Ref	
eml-version	Character Data	
feedback	Name Token Group	
hints	Name Token Group	
href	Character Data	
href	Character Data	
identifier	Identifier Value	
identifier	Identifier Value	
identifierref	Identifier Ref	

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Attribute Comment

Name	Type	Comment
identifierref	Identifier Ref	
index	Identifier Ref	
interaction-type	Name Token Group	
invisible	Name Token Group	
match-persons	Name Token Group	
max-persons	Character Data	
max-transactions	Character Data	
min-persons	Character Data	
number-to-select	Character Data	
other	Character Data	
parameters	Character Data	
position	Name Token Group	
property-identifier	Identifier Value	
property-of	Name Token Group	
property-ref	Identifier Ref	
ref	Identifier Ref	
ref	Identifier Ref	
restriction-type	Name Token Group	
role	Character Data	
score	Name Token Group	
score	Name Token Group	
search-type	Name Token Group	
select	Name Token Group	
sort	Name Token Group	
test-property-type	Name Token Group	
title	Character Data	
track	Name Token Group	
transaction-type	Character Data	
type	Name Token Group	
uri	Character Data	
uri	Character Data	
username-property-ref	Identifier Value	
version	Character Data	
view	Name Token Group	
with-control	Character Data	
xmlns	Character Data	
xmlns	Character Data	
xmlns	Character Data	
xmlns	Character Data	
xmlns	Character Data	

Element / Attribute Comment

Name	Type	Comment
access-count	Element	The number of times the actor accessed the current run of the unit of learning. (datatype=integer.
xmlns	Attribute List Attribute	
act	Element	A play is modelled according to the metaphor of an theatrical performance. A play consists of a series of acts and an act consists of a series of role-part's. In an act each role may be present in only one role-part. An act represents a series of concurrent role-part's. There is at least one act in a play. When there is more then one act in a play these are interpreted in a sequenced manner: from first act to last act. Only one act in a play is the active act at any moment in time, starting with the first. When the first act is completed, the second act is made the active act. The first is still visible and accessible, but in the interface it is made clear that it this is only looking in history. When the second act is completed, the third act is made active, etc. Acts which are sequenced in row after the current active act are never visible. Conditions cannot overrule this, meaning that the act is of higher priority than conditions.
identifier	Attribute List Attribute	
act-ref	Element	*EML new* Refers to an act (in method/play/act).
ref	Attribute List Attribute	
activity-progression	Element	Activity-progression provides information about the status of the progression of: - the run of the current unit of learning (one result per run at any moment, same result for all persons, so returned for one). - learning-activities (per person, default in own dossier, when in context of monitor/role-ref than for all persons). - support-activity (per person, default in own dossier, when in context of monitor/role-ref than for all persons). - activity-structure (per person, default in own dossier, when in context of monitor/role-ref than for all persons). - play (one result per run at any moment, same result for all persons, so returned for one). - act (one result per run at any moment, same result for all persons, so returned for one). It returns the value of the progression-status in a table when there is more than one result. The values per item can be any of these: (not started started completed). The table has to be designed by implementers. At least the following general information must be provided: - what type of object the table is about (unit-of-learning, act, etc.) - the date of production of the table - the dossier the table is derived from (self, role-ref: indicate which role). The table columns contain at least the user identification, the title of the object and the progression status.
track	Attribute List Attribute	
xmlns	Attribute	

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Attribute Comment

Name	Type	Comment
xmlns	Attribute	
activity-selection identifier isvisible number-to-select sort	Element Attribute List Attribute Attribute Attribute	A selection is a structure where users may select and complete the activities contained in any order. Selections may be nested with other sequences or selections. When the attribute 'number-to-select' is set, the activity-selection is completed when the total number of activities selected are completed. The number-to-select must be similar or smaller than the number of activities (including unit-of-learning's) which are at the immediate child level. When the number-to-select isn't set, the activity-selection is completed when all the activities in the selection are completed. The attribute 'sort' determines the sort-order in relation to the visibility. Default the order in which activities are made visible is in the order specified in the activity-selection structure. When the value is set to 'visibility-order', activities are presented in the order they were made visible (this imitates a kind of inbox: new activities come available over time).
activity-selection-ref ref	Element Attribute List Attribute	
activity-sequence identifier isvisible	Element Attribute List Attribute Attribute	A sequence is a structure of activities which must be completed in the specified order. When entering the sequence for the first time, a user can access the first activity in the structure. When the first activity is completed, the second activity is made visible at runtime, etc. for the whole structure. The activity-structure is completed when the last activity in the structure is completed. Sequences may be nested with other sequences or selections.
activity-sequence-ref ref	Element Attribute List Attribute	
activity-structure-ref ref	Element Attribute List Attribute	Reference to an activity-structure.
and-answer ref	Element Attribute List Attribute	Adds additional conditions (in multiple response formats).
answer-choice identifier isvisible score	Element Attribute List Attribute Attribute	Answer-choice specifies a response option for a user in a test-item. Depending on the item-format, the answer-choice can contain a correct or an incorrect answer. This is set with the score attribute. For more information see item-format. The attribute 'score'. When this value isn't set, the interaction is just for data collection. In test situations, one or more items is correct and the others are incorrect. This attribute is used to identify correct or incorrect responses. In this case the score-value is set to true or false. The attribute isvisible. value: true (default), then the response is rendered. value: false, then the response is hidden with a show/hide control (a show/hide control is e.g. the + and - control in the windows explorer to open/close folders). This is used for question-answer item-formats, where the answer is hidden. In this case there is a standard title to be rendered in the interface (implementation dependent, e.g. '+ answer'). When a series of faq items is specified in a questionnaire object, the visibility doesn't have to be set on each answer-choice, but can be set in the context of the questionnaire object for all test-items (it overrules). The text is provided in ANY schema. In practice this means: xhtml.
answer-pattern identifier score	Element Attribute List Attribute Attribute	A pattern to match open, constructed response questions. ##not needed for eml 1.1 compliancy
answer-property property-identifier	Element Attribute List Attribute	Refers to property which value is set to the response-value of the user. Depending on the item-format the value is different. When answer-choices and answer-pairs are selected, the identifier of the response is stored (in string). When answer-patterns are selected, the concrete answer is stored (in text). With multiple response the answer identifiers are stored in a comma separated list.
class class	Element Attribute List Attribute	Inherited from HTML (related to CSS). In EML 1.0 this was called 'content-type'. It is used to identify classes of common objects in order to manipulate them once. A class attribute contains a CDATA string. Just as in HTML more than one class may be specified in one CDATA string, each separated with a blank space. The priority order for classes is the same as specified in the CSS specification (see www.w3.org/style/css). Note that the classes in eml can be used for style sheet like functions (e.g. set visibility), but they can also have a semantic classification purpose (just as in HTML) not connected to style sheets or automated processing at all.

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Attribute Comment

Name	Type	Comment
title with-control	Attribute Attribute	
conference	Element	In EML 1.0 this was three elements: announcement-object, asynchronous-conference, synchronous-conference. In the new conference element the structure of the three elements are harmonized and integrated. The distinction between the three categories are now specified with the attribute 'conference-type'. The elements participant, observer, conference-manager, moderator facilitate the setting of the user rights in the conferences. It depends on the implementation how this is managed: 1. when the conference system is an integral part of the runtime it is expected to be set automatically; 2. when the conference is external the user-rights can be set manually by the conference manager. The conference managers, must be able to get a list from the runtime agent about which conferences of what type, for what users with what rights must be set. 3. the latter can also be implemented by a developing a legacy interface to the rights management system of the conferencing system. In all instances the runtime system must be able to provide this information in a structured way. The item element refers to the resource where the conferencing system is to be found or identified. External conferencing systems can be of any kind accessible through the internet (resource type is webcontent). Examples: netmeeting, placeware (synchronous), first-class, lotus notes, news groups (asynchronous). An announcement object sets the rights: creator of announcement = participant. Reader of announcements = observer.
conference-type	Attribute List Attribute	
creator	Element	*Dublin Core* Inherited from the Dublin Core Metadata Initiative (dublincore.org). Definition: An entity primarily responsible for making the content of the resource. Examples of a Creator include a person, an organisation, or a service. Typically, the name of a Creator should be used to indicate the entity. The role attribute determines what the role of the creator was/is. E.g.: author, contributor, etc. This role is rendered in the user-interface. When there is no role specified, the term 'creator' is used.
role	Attribute List Attribute	
current-datetime	Element	The element can be used as an operand in an expression. Represents the current datetime. This is a datetime specification of the form: CCYY-MM-DDThh:mm:ss CC is the century; YY is the year (year 0000 is prohibited); MM is the month; dd is the day. T is the date/time separator. hh are the hours; mm are the minutes; ss are the seconds. (see ISO 8601)
xmlns	Attribute List Attribute	
datatype	Element	The following types are supported: boolean, integer, real, string, datetime, text, file, uri. These are also predefined in the attribute datatype. For extensions use the 'other' value and specify the content in the element self. boolean: represents binary logic, true or false (aliases: yes/no; 1/0). NB: just as any other data types, booleans can also have <no-value>. integer: is the standard mathematical concept of integer numbers, representing whole positive and negative numbers (including zero), ranging from: -9223372036854775898 to 922372036854775807 (alias: longinteger). real: standard mathematical concept representing arbitrary precision decimal numbers, and must be capable of handling a number to 18 decimal places at least. string: represents any legal character strings. The minimal maximum number of characters is 2000. datetime: This is a datetime specification of the form: CCYY-MM-DDThh:mm:ss CC is the century; YY is the year (year 0000 is prohibited); MM is the month; dd is the day. T is the date/time separator. hh are the hours; mm are the minutes; ss are the seconds. (see ISO 8601). There is also an optional timezone separator. Partial productions of the lexical expression are not allowed. duration: specifies an amount of time: the duration of an event in relative terms (e.g the duration given the start datetime of the run of a unit-of-learning. The format - also used in the W3C XML schema specification - is: PnYnMnDnHnMnS where: P is the designator that must always be present. nY is a variable where an integer is filled in. nY represents the number of years nM represents the number of month nD represents the number of days T is the date/time separator which must always be present when representing time. nH is the number of hours nM is the number of minutes nS is the number of seconds. Example: P2Y0M1DT20H10M55S Meaning that the duration is: 2 years and 0 month and 1 day and 20 hours and 10 minutes and 55 seconds. Limited forms of lexical production are also allowed E.g. a duration of 40 minutes is expressed: PT40M. A duration of 30 days is: P30D text: represents any legal character strings. The minimal maximum number of characters is 64000 (about 10 pages of A4 text). file: represents any binary file as datatype. The property stores this file. uri: represents a URI according to the IETF's RFC 2396 Note: according to the w3c only the word URI should be used in future and not URL or URN. (see: http://www.w3.org/TR/2001/NOTE-uri-clarification-20
datatype	Attribute List Attribute	
datetime-activity-started	Element	The element can be used as an operand in an expression. It refers to: - the current run of the unit-of-learning - a learning-activity - a support-activity - an activity-structure This element represents the datetime that a person first accessed the activity-description content for an activity. This value has to be added automatically at runtime. This is a datetime specification of the form: CCYY-MM-DDThh:mm:ss CC is the century; YY is the year (year 0000 is prohibited); MM is the month; dd is the day. T is the date/time separator. hh are the hours; mm are the minutes; ss are the seconds. (see ISO 8601)
ref xmlns	Attribute List Attribute Attribute	
dependency	Element	*IMS CP* Inherited from IMS Content Packaging (including the attributes). So

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Attribute Comment

Name	Type	Comment
identifieref	Attribute List Attribute	for a description and function of this element the IMS Content Packaging Specs (version 1.3).
design	Element	This element specifies the design (alias: instructional design) of the unit of learning. Any unit-of-learning can have zero or one design. A unit-of-learning with zero designs are to support content-updates for an existing run of a unit-of-learning. For new runs a design is always expected to be present. Designs can be re-used in different units-of-learning, with different content. In an existing run of an unit-of-learning, the design may never be adapted for consistency reasons, however the href's of the resources may be adapted during the run as well as the content of the resources where the hrefs point to. For local resources which are available in the unit-of-learning package, this means that a new package may be loaded during the existing run, to update the content of the run (but not the logic, which is defined in the design). Note on authoring templates: In order to implement design templates in the authoring environment, it is expected that a meta dtd will describe how a specific design may or may not be adapted (e.g. an author may include extra learning activities at certain points, but not on other points). RULE: In order to distinguish between references (IDREF) within the design model and references to resources, the following rule applies: Attribute name 'ref' (IDREF) refers to an element with an identifier within the design. Example: <act-ref ref=""> refers to an act element within design. Elements with the 'identifieref' attribute, refer to resources. Example: <item identifieref=""> refers to a resource. The attribute name 'uri' is used for URI's which are ID's and the attribute 'href' is used for URI's which refer to uri's as ID's.
uri xmlns	Attribute List Attribute Attribute	
email-data	Element	This element refers to the property resources where the relevant e-mail data can be found for the connected role. This is used for send-mail purposes (as a service in the environment, or in notifications). This element has two attributes: - email-property-ref: this attribute contains a reference to the property containing the email address of the users being notified - username-property-ref: this optional attribute contains a reference to the property containing the user name of the users being notified. Both properties (email, username) should be available for all persons assigned to the role and the sending party.
email-property-ref username-property-ref	Attribute List Attribute Attribute	
environment	Element	The container 'environment' is a container for resources which are used during the performance of activities. References to the environment are only done within the 'design' container (with environment-ref).
identifier isvisible	Attribute List Attribute Attribute	
environment-ref	Element	This element refers to an environment somewhere in this package.
ref	Attribute List Attribute	
existing	Element	Refers to a property already declared (e.g. in another unit-of-learning, or in the global dossier) to the knowledge of the author (see 'global-definition' what happens if the the author defines a new global property which in practice already exists). The property is referred to with href, specifying an absolute URI. NB: when validating this unit-of-learning, the URI doesn't have to be present. The declaration of the URI by an external unit-of-learning can happen at any time. So this is only under the control of the author.
href	Attribute List Attribute	
expression-schema	Element	The container for the expression commands for use within the design element. It is used in the context of the ements: <expression/> and <if/>. It can be used global.
xmlns	Attribute List Attribute	
file	Element	*IMS CP* Inherited from IMS Content Packaging (including the attributes). There is a change in occurrence. In IMSCP the file occurs 1 or more times. In the unit-of-learning it occurs zero or more times (in order to allow shorter descriptions of resources which are only dependent on URI's and not on files). So for a description and function of this element the IMS Content Packaging Specs (version 1.3).
href	Attribute List Attribute	
first-access	Element	The datetime of first access to the current run of the unit of learning. (datatype= datetime).
xmlns	Attribute List Attribute	
glob-property	Element	A global property is a global unique property, which stores one value, independent of user, units-of-learning and role. The identifier can be used to refer to the property in this unit-of-learning-package. Operations can refer to this identifier to operate on the value. ##not needed for eml 1.1 compliancy
identifier	Attribute List Attribute	
global-definition	Element	NB: This element is only included for implementations which lack a tool to define global properties: it is an optional construction in EML1.1 and can be removed in systems which only refer to existing (elsewhere) decalared global properties. To

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Attribute Comment

Name	Type	Comment
uri	Attribute List Attribute	force consistency, the next rule applies: RULE: Once a global property has been defined in whatever context, it can never be changed! This is also true for re-publications of the same unit-of-learning. So the definition is only used when the URI (href) doesn't exist yet. Otherwise it is ignored. The URI must be an identifier which identifies the global property global unique. It must be an absolute URI. When the URI is an URL, the URI doesn't need to point to the property location, but can be interpreted as an identifier.
global-elements	Element	The global EML elements can be inserted in other vocabularies, like xhtml (it must be well-formed XML). Use the namespace xmlns="http://eml.ou.nl/eml11", which is already predefined in all elements. Examples of use in xhtml: <html xmlns:eml="http://eml.ou.nl/eml11"> <head><title>example emlcontent</title></head> <body><p>This is an example of the use of emlcontent. This text is xhtml. The eml properties are included in a table.</p> <p>Provide your email address: <eml:set-property href="email" max-transactions="1"/></p> <table><tr><td><i>learning style</i></td> <td><i>e-mail address</i></td></tr> <tr><td><eml:view-property href="learning-style"/> </td> <td><eml:view-property href="email"/> </td></tr> </table></body> </html> In the example the namespace is declared with the prefix eml. Of course the namespace could also have been defined in the element itself (xmlns="http://eml.ou.nl/eml11").
xmlns	Attribute List Attribute	
globpers-property	Element	global personal property, alias: portfolio-property. This property can have a different value for every user, independent of the different runs of units of learning (its specifies the portfolio of the user). The property is owned by the person. The identifier can be used to refer to the property in this unit-of-learning-package. Operations can refer to this identifier to operate on the value.
identifier	Attribute List Attribute	
if	Element	If refers to an expression-schema which evaluation results in the value: true or false. The expression-schema is kept as a separate schema with a different namespace (e.g. http://eml.ou.nl/eml11/expressions. In theory different expression schema's may be used. However it is preferred to use the expression (and calculation) schema provided with eml). In authoring environments it is expected to be integrated at this place of the schema. When the expression resolves to 'true', the 'then' rule fires. When it resolves to 'false' the 'else' rule fires when it is present (otherwise nothing happens in this rule).
xmlns	Attribute List Attribute	
index-element	Element	This element selects the element to make the index on. The index attribute specifies the element to index-on (only one reference per index-element). This indexing only makes sense when there is a structure to index on, or underlying text to index for free-text-search.
index	Attribute List Attribute	
is-member-of-role	Element	This element is true when the person accessing this element is a member of the specified role.
ref	Attribute List Attribute	
item	Element	*IMS CP* Inherited from IMS Content Packaging (including the attributes). So for a description and function of this element the IMS Content Packaging Specs (version 1.3). The identifier attribute is included to support the IMS CP specification, but is of no use in this dtd.
identifier identifierref invisible parameters	Attribute List Attribute Attribute Attribute	
item-format	Element	Test-items can have several fixed formats, which are described in the testing literature. A main distinction can be made between test items based on constructed response, or test items based on selected response. In the first case learners have to produce the answer on questions, tasks or problems themselves, in the second case they have to select or manipulate answer-options provided in the test-item. Examples of the first category is: open answer questions, essay writing. Examples of the second category are multiple-choice or matching questions. Most test-items can be modelled using a generic, abstracted structure like the one provided in EML. However, the user agent has to know what the item-format is in order to be able to render the item in an appropriate way. For instance: in the item-format of true-false questions, there can be two questions which are interpreted and rendered as statements. For each item-format a separate rendering scheme has to be worked out when rendering EML. In the attribute some predefined formats are provided. For extensibility new types may be provided in the content of the object (the attribute 'other' must be provided then). Supported item formats are: 1. question-answer (alias: faq) 2. multiple-choice (alias: mc) 3. multiple-respons (alias: mr) 4. true-false (alias: tf) 5. prompt (alias: pr) For these items a definition is provided here: ad 1. question-answer format. Faq's are used for different purposes, not only for testing, but mainly for support functions. The format is: - one question - one answer-choice (invisible set to 'true' or 'false'). When invisible is true, the faq item is rendered in a sequence of question/answer where it is evident from the rendering what the question is and what the answer. When invisible is false, the faq item is rendered as a link with the anchor 'answer'. The answer is only visible to the user after selection of the link (e.g. rendered as a collapse and expand

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Attribute Comment

Name	Type	Comment
interaction-type	Attribute List Attribute	control). When a collection of question-answer items is modelled for a faq, use a questionnaire-object. The visibility of the answer-choices can be set for all items in the questionnaire-object. ad 2. multiple-choice A mc test-item is defined as any item having two parts: - one question (a stem), describing a problem for a learner. - two or more (mostly four or five) answer-choices, one of which is correct (property score is set to 'correct') and the rest is wrong (score set to 'incorrect'). The student is asked to select the righ
item-ref ref	Element Attribute List Attribute	Refers to the identifier of an item in the design context.
knowledge-object class identifier isvisible parameters	Element Attribute List Attribute Attribute Attribute Attribute	Definition: A knowledge-object is every object in an environment which is used to transfer knowledge to the learner. Examples: books, articles, CBT (interactive or linear). The object can contain resources of type 'webcontent' or of type 'emlcontent'.
knowledge-object-ref class ref	Element Attribute List Attribute Attribute	This element refers to a knowledge-object somewhere in this design. This element can have an own class, overruling the class specified in the referred object.
last-access xmlns	Element Attribute List Attribute	The date and time of last access to the current run of the unit of learning is returned (datatype=datetime).
learner create-new href identifier match-persons max-persons min-persons	Element Attribute List Attribute Attribute Attribute Attribute Attribute Attribute	In every design there is at least one learner-role. Learners can be 'nested', meaning that a role may be divided in sub roles. E.g. in an educational game you can distinguish: <learner ref="student"> <title>student</title> <learner ref="chair"> <title>chair</title> </learner> </learner> .. <resources> <resource identifier="student" type="role" href="http://ou.nl/role/student.role"/> </resources>
learning-activity identifier isvisible	Element Attribute List Attribute Attribute	Definition: a learning activity is an instruction for a learner to perform an task within an environment. Examples: solve a problem, study a book, discuss a topic with peers, write a dissertation, etcetera. This is the core element in EML.
learning-activity-ref ref	Element Attribute List Attribute	Refers to a learning-activity. The element can be used as an operand in an calculation or expression.
loc-property identifier	Element Attribute List Attribute	Local property, alias: run-property Local property. This property has the same value in a run for every user. The property is owned by the run of the unit-of-learning. The identifier can be used to refer to the property in this unit-of-learning-package. Operations can refer to this identifier to operate on the value.
locpers-property identifier	Element Attribute List Attribute	Local personal property. This property can have a different value for every user in all the roles for a run of this unit-of-learning. The owner of the property is a person in a run. The scope of this property is the run. The identifier can be used to refer to the property in this unit-of-learning-package. Operations can refer to this identifier to operate on the value.
locrole-property identifier	Element Attribute List Attribute	alias: group-property Local role property. This property has the same value for every user in the specified role during the run of a unit-of-learning. The property is owned by the role in the run of the unit-of-learning. The identifier can be used to refer to the property in this unit-of-learning-package. Operations can refer to this identifier to operate on the value.
meta	Element Attribute List	This element is used to model metadata schema's, like the IEEE LTSC LOM or the Dublin Core metadata schema. The functionality is that a user agent for human reading may render the title and the value of an element. When the isvisible attribute is set to 'true' (default), then the data are rendered to the user by the user agent. What is presented is: a) When the title is available: title and value is presented b) When the title is not available: element-name and value are presented.

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Attribute Comment

Name	Type	Comment
isvisible	Attribute	
metadata	Element	This is the EML metadata model. Most of the elements are derived from the Dublin Core Metadata (http://dublincore.org). This set of elements is selected to enable automated processing. There is a core set of elements in the metadata which are processed differently, e.g. the title is always rendered in the user interface at a special place; rights are always accessible from the interface, etc. Metadata which are only shown (or even hidden) to the users are all modelled in the element 'meta'. Elements which have only the function of rendering information to users, or facilitate search processes are all contained in the 'meta' element. When a user scheme has a title, like <code>lom:general/lom:title</code> . Authors can decide what schema's to use (LOM, Dublin Core or proprietary). The complete LOM set can be modelled in the 'meta' element. However it would be advised to map the common elements to the base set (e.g. <code>lom/general/title</code> can be mapped to <code>unit-of-learning/metadata/title</code> (e.g. using XSLT). The metadata elements selected are a general set, used for all the different elements in the unit-of-learning model. This means that some elements are not useful in every context. E.g. the element 'format' makes no sense in the context of a role specification, but does in the context of a resource definition. The decision to use certain elements or not is up to the user. A change from the previous EML10 dtd is that this dtd specifies the metadata in a fixed order (so there is more control over the occurrences). User-agents can transform this order in any other order, but must be explicit on this point: they cannot expect that the author has created the right order of elements). The metadata of a unit-of-learning are fixed in any publication.
xmlns	Attribute List Attribute	
metadata-schema	Element Attribute List	
xmlns	Attribute	
play	Element	The play is the root element when interpreting the design. It represents the flow of activities during the learning process (the 'workflow' or better: the 'learningflow'). The play is modelled according to the metaphor of an theatrical performance. A play consists of a series of acts and an act consists of a series of role-parts. There is always at least one play in every design (and every unit-of-learning). In runtime the play is interpreted to show and hide activities, (other)units-of-learning, environments and resources to the users. When there is more than one play, these are interpreted concurrently and independent of each other. The same user can see the results of more than one play in the user-interface. Practical experience have shown that a lot of designs use multiple plays, to represent the flows of activities per role, e.g. a play for the learners and a play for staff. However this can only be done when the activities are independent of each other.
identifier	Attribute List	
isvisible	Attribute Attribute	
play-ref	Element	*EML new* Refers to a play (in method/play).
ref	Attribute List Attribute	
presentation-option	Element	There is a fixed set of presentation options provided in three attributes: - Feedback attribute: Default value: suppress means that feedback (in the feedback element) is suppressed when rendering. Value: 'immediately' then feedback is provided immediately after the user has provided the response. Value: 'delayed', then the feedback is given after completion of the total test (for all items in a list). - Hints attribute: default value: suppress-hints, this suppresses hints when rendering. value: provide-hints: the hints and hints anchors are available for the users. - Answers-visible: This option overrules isvisible attributes set on answer-choice. This is used in the context of faq's (question-answer). When set to true (default) the answers are rendered in the user interface. When set to false, the answers are hidden under the standard link 'answer'. Only when the user selects the link the answer becomes visible. This can be implemented with a show/hide control (e.g. + answer). For extensibility, new presentation-options may be defined. The attribute other-type can be used to specify the name of the new presentation-option. The value of the new presentation-option can be specified in the content of the element.
answers-visible	Attribute List	
feedback	Attribute	
hints	Attribute	
other	Attribute	
property-group	Element	A definition of a group of properties which belong together (and are edited in e.g. a form). It can only contain properties of the same type. The identifier can be used to refer to the property-group in this unit-of-learning-package. Operations can refer to this identifier to operate on the value.
identifier	Attribute List Attribute	
property-group-ref	Element	
ref	Attribute List Attribute	
property-ref	Element	Refers to a property This can be any of the following property types: - local property - global property - local personal property - local role property - global personal property - local role property The ref refers to the property declaration in the design. The element can be used as an operand in an calculation or expression..
ref	Attribute List Attribute	

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Attribute Comment

Name	Type	Comment
questionnaire-object class identifier invisible parameters	Element Attribute List Attribute Attribute Attribute	Definition: questionnaire-object is a component within the environment which specifies a questionnaire. The questionnaire can be used in different functions: self-tests, survey, to set properties for personalization. The resources for services are of type: 'emlcontent'.
questionnaire-object-ref class ref	Element Attribute List Attribute Attribute	This element refers to a questionnaire-object somewhere in this package. This element can have an own class, overruling the class specified in the referred object.
resource href identifier type	Element Attribute List Attribute Attribute Attribute	*IMS CP* Inherited from IMS Content Packaging (including the attributes). So for a description and function of this element the IMS Content Packaging Specs(version 1.3). Different types are supported in EML 1.1 (they are provided as a fixed nametoken list at the type attribute. This is done for validation reasons, but isn't part of the IMSCP specification itself (there the type is of type PCDATA). - webcontent (see IMS spec): unparsed. The URI is interpreted as an URL. Webcontent is defined as: all content which can be hosted in, or launched by a web browser. This includes: html, xml, applets, and files that are connected to an application through their extension (e.g. *.doc launches a word processor), common web applications like flash, real media, etc. - emlcontent is xhtml mixed with eml global-elements and possible well-formed XML extension modules (e.g. mathml, smil, vectorgraphics). This is parsed element by element before it is rendered. emlcontent must be well-formed XML. The URI may be interpreted as ID's or as URL's depending on implementation. Emlcontent represents personalized content, like the content of a monitor object, or personalized knowledge-object content. In some cases the resource is a physical object, not accessible for the user agent. In that case the author can refer to a file (webcontent or emlcontent) referring to the description of the object.
resources xmlns	Element Attribute List Attribute	*IMS CP* Inherited from IMS Content Packaging (including the attributes). So for a description and function of this element the IMS Content Packaging Specs (version 1.3).
restriction restriction-type	Element Attribute List Attribute	Zero or more restrictions of different type may be set on the property-values, meaning that the property-value is valid when it is of the specified data type and its value is within the specified restriction rules. Zero or more restrictions can be specified (these have the same format as specified in the W3C XML schema 1.0 specification) in the attribute: 'restriction-type'. However properties may not contain arrays (lists) of data, but can only contain a single value. So restrictions only apply to this single value. (Also the 'whitespace' restriction is not supported in eml). The restriction types are: length: constrains the length of the property value of a textual datatype (string, text or uri) in terms of the of characters that it can have. minLength: constrains the minimum number of characters that a property of textual datatype can have for its value. maxLength: constrains the maximum number of units of length that a property of textual datatype can have for its value. enumeration: constrains the value of a property to a specific value (use for value alternative lists). This was called 'value-list' in EML 1.0. maxInclusive:constrains the value of an ordered (integer, real, datetime) property to a specific inclusive upper bound. minInclusive: constrains the value of a ordered property to a specific inclusive lower bound. maxExclusive: constrains the value of a ordered property to a specific exclusive upper bound. minExclusive: constrains the value of a ordered property to a specific exclusive lower bound. totalDigits:constrains the value of a decimal property to a specific number of digits it must contain. fractionDigits: constrains the value of a decimal property to the maximum number of digits it may have after the decimal point. pattern:constrains the literals comprising the value of a property to a pattern defined by a regular expression (##pattern is not needed for eml 1.1 compliancy).
role-part identifier	Element Attribute List Attribute	A play is modelled according to the metaphor of an theatrical performance. A play consists of a series of acts and an act consists of a series of role-part's (alias 'scene'). A role-part relates exactly one role to exactly one type of activity (including the performance of another unit-of-learning and activity-structures). Role-parts within one act, are performed concurrently. The definition of a role-part differs here somewhat from real role/parts's in a theatrical play (here the metaphor is not completely correct, like all metaphors). When an activity, item or environment's attribute invisible is set to 'false', the link in the activity-tree may be made visible when the role-part sets the activity for a role (implementation dependent), but the content isn't accessible. First a condition has to specify that the activity is visible again (show condition).
role-part-ref ref	Element Attribute List Attribute	*EML new* Refers to a role-part (in method/play/act/role-part).
role-ref	Element	role refers to the identifier of the resource of the role. The element can be used as

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Attribute Comment

Name	Type	Comment
ref	Attribute List Attribute	an operand in an expression.
roles	Element	Roles is a container for the two general roles: learner & staff. A href can be provided when referring to a global role (e.g. a role defined by an institute). This is obligatory when specifying a global role and connected globrole-properties. Global roles are specified with the href attribute. The rest of the declaration, like information-for-role, is local. It is not possible to declare global roles in a unit of learning. This is just an organizational issue and is nothing more or less than providing absolute URI's for roles. The attribute 'identifier' on roles can be used to refer to the whole group of all roles within the learning-design (learners and staff). The attributes: - min-persons specifies the minimum number of persons which must be bound to the role in order to start a run. - max-persons specifies the maximum number of persons which can be bound to the role in order to start the run. In both cases: when the attributes are empty, there are no restrictions. - match-persons. This attribute is used when there are several sub roles (e.g. chair, secretary, member). Persons can be matched exclusively to the sub roles, meaning that a person who has the role of chair, may not be bound to one of the other roles at the same time. When it is not exclusive, persons may be bound to more than one sub role (this is the default situation). -create-new. This attribute indicates wether multiple occurrences of this role are may be created during run-time. When the attribute has the value "not-allowed" than there is always one and only one instance of the role. If the value is "allowed" a mechanisme in the run-time is provided to create new instances of this role. If a new instance of a role is created, new instances for all sub-roles of that role are created as well. In every learning-design at least one learner role is specified. In institutional installations the role names are fixed. For instance in the OUNL role-identifiers for student is: 'student'. To address this, use the following declaration: This element specifies the roles distinguished in this unit of learning. NB: the role declarations don't contain properties anymore. They are specified in the method section and bound to roles through a 'role-ref'.
identifier	Attribute List Attribute	
score-value	Element	This value refers to a property-resource. The user agent automatically sets this property-value to the score of the user-interaction when the response-score attribute on the element 'response' is set. The score is '1' when the user answered correct and '0' when the user answered incorrect. E.g. A MC question: Who's the queen of holland? a. Beatrix (correct) b. Maxima (incorrect) c. Juliana (correct) d. Wilhelmina (incorrect) When the user selects Juliana, the property score-value is set to '0'. When the user selects Beatrix, the property is set to '1'. When there are no score attributes set in the response element, the score-value is ignored (not set). The attribute 'calculate-score' regulates how the score is aggregated when more then one score-value refers to the same property. value: add: the new value is added to the existing value (this is default). value: clear: the property is set to the new value (the old value is cleared).
ref	Attribute List Attribute	
search	Element	This element specifies how a user can access the indexed entities. There are three possibilities: 1. the user gets a free text search dialog, where he can search the index in a free text format (this also means that the index has to be build for free text retrieval). The syntax for free text retrieval is implementation dependent, e.g. the format found in search engines like Google or Altavista. 2. the user is presented a text index (table of content) with (hyper-)linked (or on other media e.g. page numbers) references to the source. 3. the user is presented a text index (table of content) without (hyper-)linked references. This provides e.g. information about the structure of the unit of learning.
search-type	Attribute List Attribute	
send-mail	Element	This service is used to send mail to users or to all users in a role. This is set with the attribute 'select'.
select	Attribute List Attribute	
service	Element	Service is a container for a variety of services to the users (communication, index-search and monitor). The resources for services are of type: 'emlcontent'.
class identifier invisible parameters	Attribute List Attribute Attribute Attribute	
service-ref	Element	This element refers to a communication-service somewhere in this package. This element can have an own class, overruling the class specified in the referred object.
class ref	Attribute List Attribute Attribute	
set-property	Element	With this element a specified property-value may be set by the user. It works outside of the context of a textline (e.g. outside <p>). The view attribute sets whether the value or the title+value should be delivered. The user gets a control in the user-interface to set the value of the property. The type of control is dependent on the property datatype and the restrictions. In the control the current value is shown and the datatype and restrictions are made explicit so that the user knows exactly what values are valid and which are not. This allows for client-side checking of the input (dependent on implementation this may also be dealt with at the serverside). The element refers to the property URI or identifier with a ref or href. In order to avoid confusions it is good practice to include the emlcontent

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Attribute Comment

Name	Type	Comment
score	Attribute List Attribute	(implementation dependent text). E.g. when there are two statements, the generated answer-choices can be: . a) statement I and II are both false, b) statement I and II are both true, c) statement I is true and II is false, d) statement I is false and II is true. When the answer-choices are explicitly defined, this overrules the automatic generation. The text is provided in ANY schema. In practice this means: xhtml.
support-activity	Element	Definition: a support activity is an instruction for a staff or learner role, to support a role (learner/staff) or users in a role in performing the activities. The support role uses an environment in order to perform the support activity. Most of the times, support activities are performed by staff members (e.g. tutors) to support learners. In some pedagogical models however learners can support learners (peer to peer teaching). It is also possible that staff members support staff members. When the optional role-ref element is set, it is expected that the support activity will act for every single user in the specified role(s). That is: the same support activity is repeated for every user in the role(s). When the role-ref is not available, the support activity is a single activity (like the learning-activity).
identifier invisible	Attribute List Attribute Attribute	
support-activity-ref	Element	Refers to a support-activity. The element can be used as an operand in an calculation or expression.
ref	Attribute List Attribute	
test	Element	Container for tests, meaning a collection of interactions. It can be used global.
uri xmlns	Attribute List Attribute Attribute	
test-item	Element	A test item is an item which is used in assessments. These are not secure assessments for examination purposes, but for self assessment, informal progress assessment or assessment of a certain user state (intake, preference, survey, etc.). This element is used globally to include test items into e.g. xhtml content. When it contains local (idref) references to property-resources it MUST be included into the unit-of-learning package. The resource href reference must then be relative. When used in global content (not packaged), then the property references are ignored, but the questions work.
class uri xmlns	Attribute List Attribute Attribute Attribute	
test-property	Element	Test properties are automatically declared and set They are available for reading in the current run by referring to their identifier with property-ref & view-property referring to the identifier of the test-property. Some predefined values are included in the attribute test-property-type. Extension types could be stated in the element content (PCDATA). In that case the attribute test-property-type should be set to 'other'. Values of attribute 'test-property-type': - proportion-correct: the proportion (=percentage divided by 100 to get a proportion as a fraction of between 0 and 1). The proportion has 1 digit behind the decimal point. Valid values are e.g.: 0, 0.2, 0.9 and 1. This property has a standard automatic declaration: locpers-property: title="proportion correct of test [insert questionnaire-object/title]" datatype="real" - number-correct: Number of correct answer provided by the user at any moment during the test performance. This property has a standard automatic declaration: locpers-property: title="number correct of test [insert questionnaire-object/title]" datatype="integer" - number-incorrect: Number of incorrect answer provided by the user at any moment during the test performance. This property has a standard automatic declaration: locpers-property: title="number incorrect of test [insert questionnaire-object/title]" datatype="integer" - number-answered: Number of items answered by the user at any moment during the test performance. This property has a standard automatic declaration: locpers-property: title="number of items answered of test [insert questionnaire-object/title]" datatype="integer" - number-of-items-provided: Number of test-items provided to the user. This is a fixed value. This property has a standard automatic declaration: locpers-property: title="number of items in test [insert questionnaire-object/title]" datatype="integer" - other For extensions. The value of the extensiontype is specified in the content of the element.
identifier test-property-type	Attribute List Attribute Attribute	
time-limit	Element	The time limit specifies that it is completed when a certain amount of time have been passed, relative to the start of the run of the current unit of learning. The data type time is expressed in the 'duration' format which is also used in the W3C XML schema specification. The format is: PnYnMnDnHnMnS where: P is the designator that must always be present. n is a variable where an integer is filled in. nY represents the number of years nM represents the number of month nD represents the number of days T is the date/time separator nH is the number of hours nM is the number of minutes nS is the number of seconds. Example: P2Y0M1DT20H10M55S Meaning that the duration is: 2 years and 0 month and 1 day and 20 hours and 10 minutes and 55 seconds. Limited forms of lexical production are also allowed E.g. a duration of 40 minutes is expressed: PT40M. A duration of 30 days is: P30D The time is always counted relative to the time when the run of the unit-of-learning has been started (see element: 'time-unit-of-learning-started'). The time-limit may be specified in a property (property-ref attribute). (of type loc-property, datatype=string, to be declared by

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Attribute Comment

Name	Type	Comment
property-ref	Attribute List Attribute	the author). In that case an author may set controls (set-property) on this property for users to control the value of the property. When a property-ref is specified, content in the element is ignored: the property overrules. Authors have to take care that the time limits set on role-parts, acts and plays are logical. In runtime the time limit of the play overrules the time limit on act and that one on the role-parts.
tool-object class identifier invisible parameters	Element Attribute List Attribute Attribute Attribute	Definition: a tool-object is every object in an environment, used/manipulated by the role in the performance of activities. It isn't a container for knowledge. Example: a hammer, an organizer, a general software application (productivity tools like MS-Office). The object can contain resources of type 'webcontent' or of type 'emlcontent'. Mostly it is actually a physicalobject, of which a description is provided.
tool-object-ref class ref	Element Attribute List Attribute Attribute	This element refers to a tool-object somewhere in this package. This element can have an own class, overruling the class specified in the referred object.
transaction max-transactions transaction-type	Element Attribute List Attribute Attribute	The transaction element defines: a) the number of trials a user may perform before the answer is definite. Technical (upload) errors do not count as a trial, but only successful transactions. When the element isn't specified, the number of trials is set to unlimited. This number is set with the attribute 'max-transactions', datatype=integer. When the attribute isn't set, the maximum is unlimited. b) the transaction-type (future extension, e.g. for secure transactions).
unit-of-learning	Element	GENERAL INTRODUCTION The unit-of-learning (abbr. uol) is the root element of this dtd. A unit-of-learning is part of the unit-of-learning-package. A unit-of-learning-package consists of: 1) the unit-of-learning, which is an XML file named 'eml-unit-of-learning.xml'. The unit-of-learning consists of three subparts: metadata which describe the package; a design, which describes the learning process and the resources which are referred to in the design. 2) the physical, digital files which are distributed with the package. Actors in the learning process, dealing with the interpreted units of learning are: . Learner (the people who are supported with the unit of learning in their learning process, the key role). . Staff (the people who support learners during the learning process, e.g. tutors, assessors). Beside these primary actors which are using the units of learning, there are also other actors who are dealing with the creation and management of the units of learning itself. These roles are: . Designer (designs a unit of learning in EML, including emlcontent or additions to webcontent to produce emlcontent) . Author (authors webcontent and other non-EML resources the unit of learning is referring to) . Content manager (manages versions and learning objects, produces unit-of-learning-packages) . Legal roles (manages copy-rights) . Development manager (provides the authorization to develop or publish a unit of learning and the global URI's) All roles can be spit down further to an unlimited number of sub-roles and can have different names in different organizations. All actors expect from the e-learning system to get more effectiveness, more efficiency, more attractiveness and higher accessibility. All actors fill these aspects in from their own perspective. A learner wants more effective, efficient, attractive and accessible learning; a tutor wants to tutor in a more effective, efficient, attractive and accessible way, and so forth. The translation in general categories of requirements are as follows: An Educational Modelling Language (EML), which describes a unit of learning, must meet the following general requirements: R1. It must describe units of learning in a formal way, so that automatic processing is possible (formalisation). R2. It must be able to describe designs in the unit of learning, that are based on different theories and models of learning and instruction (pedagogical flexibility). R3. It must explicitly express the semantic meaning of the different learning objects within the context of a design within the unit of learning (semantic typed learning objects). R4. It must be able to fully describe a unit of learning, including the design, the resources which it is dependent on and the metadata for all objects. The design should define all the learning objects, the relationship between the learning objects and the workflow of all learners and staff members with the learning objects, regardless of whether these aspects are represented digital or non-digital (completeness). R5. It must describe a design of a unit of learning independent of its concrete content, so that the same design can be used in different units of learning (content independence). R6. It must describe the units of learning so that repeated execution is possible, with the same content and the same design (reproducibility). R7. It must be able to describe personalization aspects within units of learning, so that the content and activities within units of learning can be adapted based on the preferences, prior knowledge, educational needs and situational circumstances of users. In addition, control over the personalization must be able to be given, as desired, to the student, a staff member, the computer or the designer (personalization). R8. The notation of the (external) content components, where possible, must be medium neutral, so that it can be used in different publication formats, like the web, paper, e-books, mobile, etc. (medium neutrality). R9. It must provide an open and technology independent interface between educational content development and the different technical user agents which interpret the units of learning. Through this, investments in educational development will become resistant to technical changes and conversion problems (interoperability and sustainability). R10. It must fit in with current open standards and specifications (compatibility). R11. It

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Element / Attribute Comment

Name	Type	Comment
eml-version identifier uri version xmlns	Attribute List Attribute Attribute Attribute Attribute	must be possible to identify, isolate, decontextualize and exchange learning objects, and to re-use these in other contexts (reusability). R12. It must make it possible to produce, mutate, preserve, distribute and archive units of learning and all of its containing learning objects (life cycle). COMMENTS ON THE ELEMENT UNIT-OF-LEARNING The identifier attribute is used to refer locally (i.e. inside this unit-of-learning-package) to the unit-of-learning. The identifier must be unique within this EML document. The URI attribute identifies the unit-of-learning globally and be absolute. The URI is only an identifier, e.g. http://ou.nl/coursecatalog/psychology/P345216 . This URI doesn't necessarily point to a concrete resource.
unit-of-learning-href href	Element Attribute List Attribute	The element can be used as an operand in an calculation or expression. This element refers to the resource of a unit-of-learning (uol). That can be the resource of this current unit-of-learning, the resource of a unit-of-learning present in the package or a resource pointing to a unit-of-learning outside of the package (absolute URI). Formally called: unit-of-study-ref. Because -ref extensions point to elements within design, this name has changed.
view-property href property-of ref view xmlns	Element Attribute List Attribute Attribute Attribute Attribute	With this element a specified property-value may be viewed. It works outside of the context of a textline (e.g. outside the context of a <p> element. The view attribute sets whether the value or the title+value should be delivered. It refers to the property with an ref or href. In order to avoid confusions it is good practice to include the emlcontent with property-operations in the unit-of-learning-package. In case of personal properties, the property value of the user himself is returned. Only in the context of a monitor object (with the role-ref specified) the values of all the users in the specified role(s) are returned.
view-property-group href property-of ref view xmlns	Element Attribute List Attribute Attribute Attribute Attribute	With this element the values of the properties in a specified property-group may be viewed. It works outside of the context of a textline (e.g. outside <p>). The view attribute determines whether the titles of the containing values should be shown. The group-title is always shown. It refers to the property-group identifier or URI with a ref or href. In order to avoid confusions it is good practice to include the emlcontent with property-operations in the unit-of-learning-package. In case of personal properties, the property value of the user himself is returned. Only in the context of a monitor object (with the role-ref specified) the values of all the users in the specified role(s) are returned.
when-answer ref score	Element Attribute List Attribute Attribute	Specifies when a feedback must be given or a property value must be set. The ref refers to the answer-choice or the answer pattern. The score refers to the fact whether the answer provided was evaluated as correct or incorrect by the system.
when-play-completed ref	Element Attribute List Attribute	This element states that an unit-of-learning is completed when the referenced play(s) is (are) completed. More than one play can be selected, meaning that all the referenced play's must be completed before the unit-of-learning is completed. When a unit-of-learning is completed this should be made aware in the runtime environment to the managers of the system.
when-role-part-completed ref	Element Attribute List Attribute	This element states that an act is completed when the referenced role-part(s) is(are) completed. A role part is completed when all members of the role have completed the particular role-part. If a role-part is associated with an environment, this environment is considered to be completed for all persons assigned to that role. Once a role-part is completed it stays completed even if the complete condition evaluates to false in a later stage, (completed = completed OR (completed condition) with completed being initially false) More than one role-part can be selected, meaning that all the referenced role-parts must be completed before the act is completed. NB: all role-parts references must be specified in the current act!

Notation Comment

Name	Comment

Comment for all Objects

Model Root: unit-of-learning

Thursday, January 31, 2002 5:41:00 PM



Notation Comment

Name	Comment

External Parameter Entity Comment

Name	Comment

Local Entity Comment

Name	Type	Comment

External Entity

Name	Type	Comment