**Estudi sobre el seqüenciament i la navegació en els projectes relacionats amb el e-learning**

**Autor: Eduard Céspedes i Borràs**

**Data: 18-07-2007**

**Índex**

**1.1.- IMS Simple Sequencing: què és?** pag. 4

**1.2.- L’arbre d’activitats** pag. 5

**1.2.1.- L’arbre d’activitats i l’especificació** pag. 5

**1.2.2.- L’arbre d’activitats i seqüenciament** pag. 5

**1.3.- El Seqüenciament i el manifest.** pag. 7

**1.4.- Apunts sobre *Simple Sequencing*** pag. 7

**1.5.- Empaquetar dintre del Content Packaging** pag. 7

**1.6.- Tracking Model** pag. 9

**2.0.- Els elements de Seqüenciament** pag. 11

**2.1.- <sequencing>** pag. 11

**2.2.- <controlMode>** pag. 11

**2.3.- <sequencingRules>** pag. 12

**2.3.1.- <preConditionRule>** pag. 13

**2.3.1.1.- <ruleConditions>** pag. 13

**2.3.1.1.1.- <ruleCondition>** pag. 13

**2.3.1.2.- <ruleAction>** pag. 14

**2.3.2.- <postConditionRule>** pag. 15

**2.3.3.- <exitConditionRule>** pag. 15

**2.4.- <limitConditions>** pag. 15

**2.5.- <auxiliaryResources>** pag. 15

**2.6.- <rollupRules>** pag. 16

**2.6.1.- <rollupRule>** pag. 16

**2.6.1.1.- <rollupConditions>** pag. 17

**2.6.1.1.1.- <rollupCondition>** pag. 17

**2.6.1.2.- <rollupAction>** pag. 18

**2.7.- <objectives>** pag. 18

**2.7.1.- <primaryObjective>** pag. 20

**2.7.1.1.- <minNormalizedMeasure>** pag. 20

**2.7.1.2.- <mapInfo>** pag. 20

**2.7.2.- <objective>** pag. 21

**2.8.- <randomizationControls>** pag. 21

**2.9.- <deliveryControls>** pag. 21

**2.10.- <adlseq:constrainedChoiceConsiderations>** pag. 21

**2.11.- <adlseq:rollupConsiderations>** pag. 23

**2.12.- <sequencingCollection>** pag. 23

**3.1.- Estratègies de seqüenciament** pag. 25

**3.1.1.- No sequencing** pag. 25

**3.1.2.- Linear** pag. 25

**3.1.3.- Linear with Controls** pag. 26

**3.1.4.- Linear Choice** pag. 26

**3.1.5.- Constrained Choice** pag. 27

**3.1.6.- Remediation** pag. 27

**3.1.7.- Competenci Assessment** pag. 28

**4.- Conclusions i proves pròpies de l’autor** pag. 29

**4.1.- Proves pròpies** pag. 29

**4.1.1.- Modificació del *Linear Controls*** pag. 29

**4.2.- Conclusions: Elements més importants** pag. 31

**4.3.- Conclusions: Estratègies de seqüenciament més importats** pag. 32

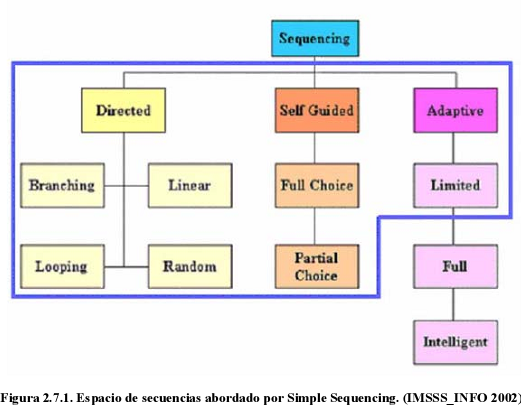
**5.- Bibliografia** pag. 34

**1.1.- IMS Simple Sequencing: què és?**

El seqüenciament que ofereix l’especificació de l’SCORM 2004 deriva de l’especificació IMS Simple Sequencing amb molt poques diferències. Per això durant aquest estudi parlarem indistintament tan d’una especificació com de l’altra, tot i que quan entrem en aspectes més específics i tècnics, com pot ser la semàntica a utilitzar en el manifest, ens centrarem exclusivament en el que explica l’especificació d’SCORM 2004 (ja sigui el CAM –*Content Agregation Model* , com el S&N- *Sequencing and Navigation*).

L’IMS *Simple Sequencing* descriu les regles necessàries per controlar el flux de les activitats educatives basant-se en els resultats obtinguts pels alumnes en les seves interaccions amb els continguts educatius.

*Simple Secuencing* està orientada exclusivament a l’alumne. No defineix seqüències per cap dels altres implicats en el procés d’ensenyament com podrien ser els professors, tutors, etc..



En aquesta imatge es pot veure la forma en que podem classificar l’ensenyament, i en blau veiem la part que està inclosa en l’especificació.

* **Seqüències dirigides**: Són aquelles en els que no es permet cap interacció amb l’usuari. Els continguts es presenten d’una forma fixa, és a dir, l’acció de l’alumne no modifica la forma de mostrar els continguts.
* **Seqüències guiades per l’alumne**: Permet decidir a l’alumne els continguts que vol visualitzar. Aquesta elecció pot ser total (*Full Choice*), on l’usuari podria elegir quin contingut vol veure en cada moment, o parcial (*Partial choice*), on s’imposen algunes condicions com poden ser prerequisits.
* **Seqüències adaptatives**: Aquest és el grau més gran a l’hora de mostrar continguts amb l’ordinador. En aquest tipus de seqüències, el sistema és capaç de decidir la manera de seqüènciar els continguts basant-se en les característiques i preferències de l’alumne.

Totes aquestes formes de veure l’ensenyament les classificarem més tard segons la forma d’implementar-les amb el seqüenciament de l’SCORM 2004.

**1.2.- L’arbre d’activitats**

(*2.1.1 Deriving and Activity Tree form a Content Package, SCORM-Sequencing&Navigation versió 1.0*)

Es pot definir amb tres punts principals:

1. Un arbre d’activitat representa l’estructura conceptual del contingut, que resulta del disseny del contingut. Finalment l’arbre d’activitat és representat com un *Content Organization* (organitzador de contingut) que és representat amb l’element *<organization>* dintre d’un paquet de continguts scorm.
2. Un LMS de SCORM traduirà el *Content Organization* en un arbre d’activitats. L’arbre d’activitats representarà l’estructura de dades que el LMS implementarà per reflexar l’estructura jeràrquica d’aquests.
3. Quan un alumne interactua amb el contingut representat per un arbre d’activitat, el LMS avaluarà la informació que vagi rebent per determinar la posició relativa de l’alumne en les activitats d’aprenentatge. En aquest context, cada alumne tindria una estructura diferent, ja que aquesta dependrà de les condicions i del que hagi interactuat amb arbre d’activitats.

**1.2.1.- L’arbre d’activitats i l’especificació**

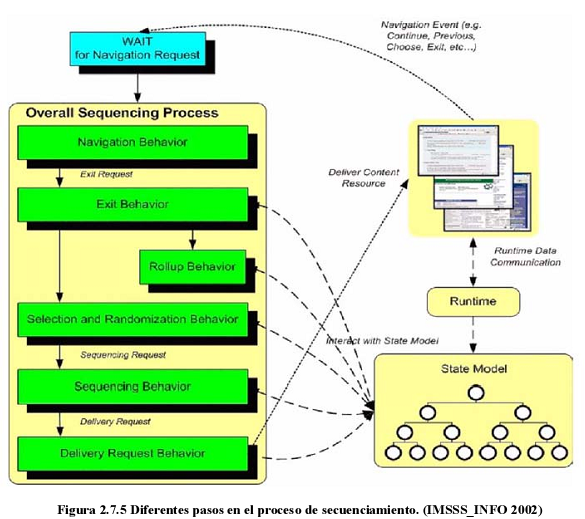
Totes les activitats que administra l’IMS Simple Sequencing s’organitzen amb l’ajuda de l’arbre d’activitats. L’especificació defineix la forma de recórrer l’arbre,i és en profunditat.

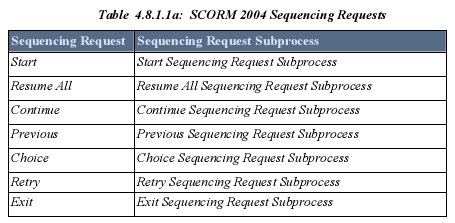
Com s’ha explicat en el punt anterior una forma de representar l’arbre d’activitats és com un element *<organization>* dintre d’un manifest. Així, l’element arrel serà *<organization>*, i cada un dels nodes serà un *<item>* que portarà una referència al contingut corresponent, amb el que aconseguim que la forma de recórrer l’arbre coincideixi amb la forma de presentar els continguts a l’usuari.

**1.2.2.- L’arbre d’activitats i seqüenciament**

Suposant que un arbre d’activitats ha estat creat i inicialitzat, anem a fer un repàs de totes les passes que s’han de fer durant el procés de seqüenciament:

1. L’alumne accedeix al LMS i estableix un context dintre d’una unitat d’aprenentatge.
2. El sistema inicia el procés de seqüenciament emetent una petició de navegació (normalment del tipus “start” o “reume all”).
3. Amb la informació de seguiment de l’alumne i la petició de seqüenciament es recorre l’arbre en busca de l’activitat adequada per presentar-la l’alumne (que correspondrà amb un node de l’arbre).
4. Una vegada seleccionada una activitat es determina si aquesta pot ser mostrada (permisos i prerequisits). En cas afirmatiu es preparen tots els recursos associats a l’activitat. En cas contrari, tot el procés de seqüenciament s’atura a l’espera d’una altra petició de navegació.
5. Una vegada rebut el recurs d’aprenentatge, l’alumne interactua amb ell i tots els processos encarregats del seqüenciament resten a l’espera de noves peticions.
6. L’alumne, el sistema o l’activitat determinada invoquen un event de navegació (continuar, anterior, sortir, etc). En aquest moment, el sistema informa al procés de seqüenciament que s’ha produït aquest event.
7. La petició és traduïda per una petició de sortida que farà que s’abandoni l’activitat actual, i una de seqüenciament amb l’objectiu de demanar la següent activitat. En cas que la petició de sortida indiqués que l’alumne vol abandonar la sessió, la petició de seqüenciament s’eliminaria.
8. S’actualitza la informació de seguiment de l’alumne i finalitza l’activitat.
9. El procés es repeteix des del pas 3.



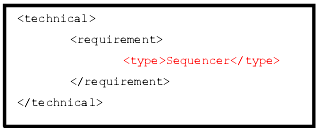


Durant tot el procés de seqüenciament els models d’estat (arbre d’activitat) i el model de seguiment resideixen en memòria, fet que obliga que s’accedeixi a ells mitjançant interfícies en temps d’execució.

**1.3.- El Seqüenciament i el manifest.**

Els elements de *IMS Simple Sequencing* s’inclouen dintre dels elements *<organization>*. L’element principal d’aquesta especificació es *<sequencing>*. Per tant quan ens trobem amb aquest element dintre d’un manifest sabrem que hem de considerar el seqüenciament i per tant que hem d’aplicar les mesures necessàries per controlar-ho en temps d’execució.

Una solució per no tindre que analitzar el manifest en la seva totalitat per veure si és necessari activar la seqüenciació podria ser utilitzar metadades en la capçalera del manifest indicant si s’utilitzarà seqüenciació.



**1.4.- Apunts sobre *Simple Sequencing***

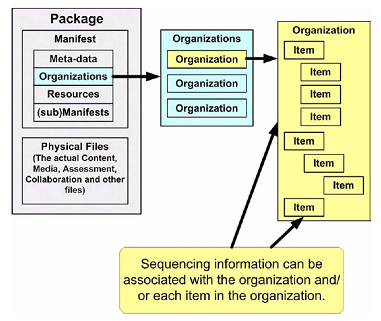
El principal problema amb el que ens trobem és la falta d’informació sobre la renovació del seqüenciament que ha sigut interromput. És més que probable que ens trobem amb el cas en que un alumne abandoni la sessió sense haver acabat la unitat que se li estava mostrant. En aquest cas, l’única sortida que ens ofereix IMS és la de reiniciar el procés de seqüenciament a la pròxima connexió de l’usuari.

Es tracta d’una solució inacceptable, especialment considerant que l’extensió de les unitats pot resultar variable, pel que un alumne es podria veure obligat a recorre una unitat molt àmplia si en la seva anterior connexió no va aconseguir finalitzar-la.

La solució, proposada per *Borja Manera Iglesias* en el seu estudi, és guardar en el fitxer personalitzat de l’alumne (descrit en el IMS LIP) la informació de seqüenciament en temps real. D’aquesta manera es pot recuperar l’entorn de seqüenciament d’una unitat en el moment de la seva interrupció.

**1.5.- Empaquetar dintre del Content Packaging**

Com a últim punt abans d’entrar de ple en la semàntica del seqüenciament definida en l’especificació concreta de l’scorm 2004 només fixar-nos com és de fàcil empaquetar el seqüenciament dintre del manifest. Per fer-ho només hem de tindre en compte que segons la posició en la que es trobi afectarà no només al node (item) actual sinó també a tots els seus fills.



Com es pot veure en aquesta figura, es pot afegir comportaments de seqüenciament tan en els *items* com en els elements *organization*, sense més problemes que afegir la informació de seqüenciament definida en l’element *<sequencing>*.

**1.6.- Tracking Model**

(*4.2 Tracking Model (pag 86) – 4.2.1.7 Tracking Behavior (pag 99) , SCORM-Sequencing & Navigation 3rd Edition- versió 1.0*)

Per cada objectiu guardarem la següent informació:

* **Objective Progress Information**
  + **Objective Progress Status** (boolean, *default False*): Indica quan l’actual objectiu te un valor satisfactori.
  + **Objective Satisfied Status** (boolean, *default False*): Indica si l’objectiu està satisfet.
  + **Objective Measure Status** (boolean, *default False*): Indica que l’objectiu te un valor vàlid per la mesura.
  + **Objective Normalized Measure** (real [-1.0.. 1.0], *default 0.0*): La mesura de l’objectiu.

Per cada activitat guardarem:

* **Activity Progress Information**
  + **Activity Progress Status** (boolean, *default False*): Indica si l’informació de progrès de l’activitat és significativa.
  + **Activity Absolute Duration** (duration accuracy 0.1 second, *default 0.0*): La duració acumulada de tots els intents que s’han fet a una activitat.
  + **Activity Experienced Duration** (duration accuracy 0.1 second, *default 0.0*): La duració acumulada de tots els intents que s’han fet a una activitat. Aquest valor no és enregistrat pels nodes (SCO o Assets).
  + **Activity Attempt Count** (unsigned int, *default 0*): El número d’accesos a una activitat incloent l’accés actual. (Nota! Aquest valor és despreciat si *Activity Progress Status* no està a *true*).
* **Attempt Progress Information**
  + **Attempt Progress Status** (boolean, *default False*): Ens indica si l’informació sobre els accesos és significativa.
  + **Attempt Completion Amount** (Real [0..1] presició de 4 decimals, *default 0.0000*). La mesura per saber quin percentatge d’accedit està una activitat, normalitzat entre 0 i 1.
  + **Attempt Completion Status** (boolean, *default False*): Indica si l’accés a una activitat està completada (s’ha accedit a tots els apartats en cas de ser un cluster).
  + **Attempt Absolute Duration** (duration accuracy 0.1 seconds, *default 0.0*): El temps entre que comença i que finalitza l’accés a una activitat.
  + **Attempt Experienced** Duration (duration accuracy 0.1 seconds, *default 0.0*): El temps entre que comença i que finalitza l’accés a una activitat, sense contar el temps en el que l’activitat ha estat suspesa.
* **Activity State Information**
  + **Activity is Active** (boolean, *default False*): Ens indica si l’activitat està en progrés en aquest moment, això vol dir, si l’usuari ha accedit però encara no ha finalitzat o si algun dels seus fills és l’activitat actual (*Current Activity*).
    - Només hi pot haver un sol “camí actiu” en un arbre d’activitats en un moment donat. El “camí actiu” començarà en el root de l’arbre i finalitzarà en el *Current Activity.*
    - Només un node fill (*leaf*) pot tindre el *Activity is Active* com a *true* en un moment donat.
    - L’Activitat actual (*Current Activity*) només tindrà activat el *Activity is Active* si encara no ha finalitzat.
  + **Activity is Suspended** (boolean, *default False*). Indica si l’activitat actual està suspesa. *Ens servirà per marcar si una activitat ja ha estat accedida i per tant si no és pot crear noves dades perquè simplement es recuperaran les dades que ja han estat gaurdades*.
    - Si l’activitat és un cluster, el seqüenciament del LMS el farà suspendre si algun dels fills ha estat suspès.
  + **Available Children** (ordered list of activities, *default all children*): Contindrà una llista ordenada que l’LMS utilitzarà pel seqüenciament. Aquest element només és reprensenta en activitats que siguin clusters i serà implícitament utilitzat durant la petició de navegació, la petició de terminació i la petició de rollup i la petició de seqüenciament (*ruleConditions*) com el conjunt a ser considerat pel seqüenciament. Nota! Aquesta llista també s’utilitzarà (i és variarà) quan s’utilitzi el imss:RandomizationControls!.

Informació genèrica a guardar per una organització:

* **Global State Information**
  + **Current Activity** (Activity, *default null*): Si l’usuari està accedint a una activitat, aquesta activitat serà la *Current Activity*.
  + **Suspended Activity** (Activity, *default null*): Indica l’activitat des de la que s’ha llançat un *Suspend All*.

**2.0.- Els elements de Seqüenciament**

(*5.1 Sequencing and Presentation (pag 199) – 5.3 Relationship to Content Packaging (pag 243) , SCORM-Content Aggregation Model- versió 1.3.2 Draft*)

En aquest apartat analitzaré els elements concrets del seqüenciament en SCORM 2004 a mode de resum estructurat o de guia. De cada element donaré una breu descripció i tot seguit llistaré els atributs que conté i els elements que en pengen.

Entrant ja en matèria podem dir que hi ha dues vies possibles per crear informació de seqüenciament:

* L’element **<sequencing>** : Aquest element, com ja s’ha comentat anteriorment, encapsula tota la informació necessària sobre el seqüenciament d’una activitat.
* L’element **<sequencingCollection>**: Aquest element pot ser utilitzat per emmagatzemar informació de seqüenciament de manera que després sigui reutilitzable des de diversos <*sequencing>.*

***Nota****!: En tot aquest resum s’utilitza molt el concepte activitat. Una activitat serà per nosaltres una manera genèrica de referir-nos tan a un <item> com a un <organization>, ja que des del punt de vista del seqüenciament els tractarem igual.*

**2.1.- <sequencing>**

Es l’arrel del seqüenciament.

**Atributs:**

* **ID** : Només s’ha d’especificar quan formi part d’un <sequencingCollection>.
* **IDRef** : Fa referència a un identificador únic (aquest identificador normalment està dintre d’un <sequencingCollection>).

**Elements:**

* <controlMode>
* <sequencingRules>
* <limitConditions>
* <auxiliaryResources>
* <rollupRules>
* <objectives>
* <randomizationControls>
* <deliveryControls>
* <adlseq:constrainedChoiceConsiderations>
* <adlseq:rollupConsiderations>

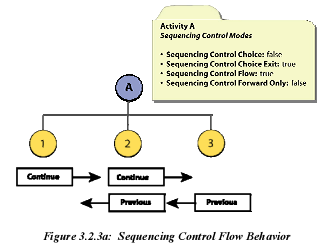
A l’apartat de conclusions analitzaré quins elements son fàcilment implementables i quins es poden descartar, tot i així els elements i atributs descartables els barraré per així ja anar-ho indicant.

**2.2.- <controlMode>**

Aquest element és el contenidor del mode de control del seqüenciament de la informació incloent descripcions dels tipus de comportament de seqüenciament que especifica una activitat. Aquest element també captura informació repartida dels tipus de resposta de seqüenciament que són permeses.

**Atributs:**

* **choice** (opcional, default *true*) : Indica que fer una tria del seqüenciament està permès (o prohibit).
* **choiceExit** (opcional, default *true*) : Indica si un fill d’aquesta activitat pot finalitzar.
* **flow** (optional, default *false*): Si el *Sequencing Control Flow* (o sigui, aquesta opció) està a *true* aleshores l’LMS ha de proporcionar mecanismes perquè l’usuari pugui “Continuar” o “Retrocedir” (*Continue, Previous*).



* **forwardOnly** (opcional, default *false*): Indica si un fill d’aquesta activitat pot anar només endavant.
* **useCurrentAttemptObjectiveInfo** (opcional, default *true*): És per marcar si es pot (o no es pot) utilitzar la informació sobre el progrés dels objectius.
* **useCurrentAttemptProgressInfo**(opcional, default *true*): Indica si es guardarà la informació sobre els intents.

***Nota!*** *Si es desactiva el flow l’arbre d’activitats es bloqueja, per tant es imprescindible que estigui activat.*

**2.3.- <sequencingRules>**

Aquest element és el contenidor per les regles que descriuen el seqüenciament. Cada regla descriurà el comportament d’una seqüència d’una activitat. Aquestes regles i condicions seran les que finalment crearan un seqüenciament diferent que s’adaptarà més o menys a l’alumne depenent de les circumstancies que avaluem.

Dintre d’una *sequencingRule* s’hi pot ficar tantes condicions del tipo pre/post i/o exit com que es vulgui, ja que cadascuna serà avaluada en el seu moment.

**Atributs:** Cap.

**Elements:**

* <preConditionRule>
* <exitConditionRule>
* <postConditionRule>

**2.3.1.- <preConditionRule>**

Conté regles que inclouen accions que són utilitzades per determinar si una activitat ha de ser accedida.

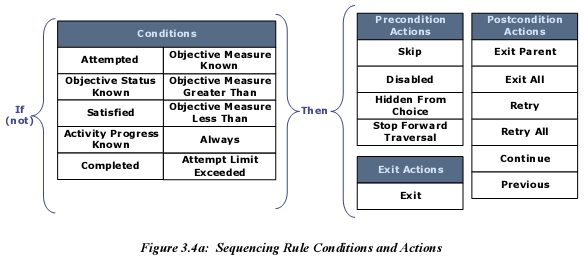
**Atributs:** Cap.

**Elements:**

* <ruleConditions>
* <ruleAction>

**2.3.1.1.- <ruleConditions>**

És el contenidor de la *llista* de condicions que seran aplicades a les pre-condicions, post-condicions i a les condicions per sortir (<preConditionRule>, <postConditionRule> i <exitConditionRule> respectivament).



**Atributs:**

* **conditionCombination** (optional, default *all)*: Indica com es combinen les regles entre elles. Valors que pot tindre: **{ all | any }**

**Elements:**

* <ruleCondition>

**2.3.1.1.1.- <ruleCondition>**

Aquest element representa la condició que és avaluada.

**Atributs:**

* **referencedObjective** (optional*)*: Indica si la condició fa referència a un altre element durant l’avaluació de la condició.
* **mesureThreshold** (optional, default *0.0*): Aquest valor és utilitzat com a llindar durant la condició d’avaluació.
* **operator** (optional, default *noOp)*: Serveix per negar una condició o deixar-la tal qual: **{ not | noOp }.**
* **condition** (**required**, default *always*): Aquest atribut representa la condició actual per aquesta condició : {
  + - **satisfied :** S’avalua a *true* si l’estat de progressió i l’estat de satisfacció associats a l’activitat estan a *true*.
    - **objectiveStatusKnown :** S’avalua a *true* si l’estat de progressió està a *true*.
    - **objectiveMeasureKnown :** S’avalua a *true* si l’estat de la mesura està a *true*.
    - **objectiveMeasureGreaterThan:** S’avalua a *true* si l’estat de la mesura està a *true*  i si la mesurà de normalització supera el llindar (el *Threshold*).
    - **objectiveMesureLessThan:** S’avalua a *true* si l’estat de la mesura està a *true*  i si la mesurà de normalització es inferior al llindar (el *Threshold*).
    - **completed:** S’avalua a *true* si l’estat dels intents (l’*Attempted*) és *true* i si l’estat de completat també està a *true*.
      * ***Nota!*** *Això a la pràctica NO és cert!*
    - **activityProgressKnown:** S’avalua a cert si l’estat de progressió s’avalua a *true* i si l’estat dels accessos (*attempted*) també s’avalua a *true*.
    - **attempted:** S’avalua a cert si l’estat de la progressió és *true* i si el contador d’una activitat es positiu (o sigui, si s’ha accedit).
    - **attemptLimitExceeded:** S’avalua a cert si l’estat del progrés per l’activitat és *true* i el contador d’accessos es igual o superior a la condició límit d’accessos per una activitat.
    - **always :** Sempre s’avalua a cert.

**}**

**2.3.1.2.- <ruleAction>**

És la seqüència desitjada si la regla és avaluada a *true*. El conjunt de les regles d’acció varia depenent del tipus de condició ( <preConditionRule>, <postConditionRule> o <exitConditionRule>).

**Atributs:**

* Si <preConditionRule>
  + **skip**
  + **disabled**
  + **hiddenFromChoice**
  + **stopForwardTraversal:** L’activitat evitarà que s’accedeixin als continguts que vinguin a continuació. No es pot utilitzar si utilitzem un sistema *flow* (*ControlMode*, paràmetre *flow=true*).
* Si <postConditionRule>
  + **exitParent**
  + **exitAll**
  + **retry**
  + **retryAll**
  + **continue**
  + **previous**
* Si <exitConditionRule>
  + **exit**

**2.3.2.- <postConditionRule>**

Igual que el <preConditionRule> l’únic que les seves accions s’aplicaran quan l’activitat intenti finalitzar.

***Nota!:****L’experiència ens ha demostrat que aquesta condició no dona bons resultats quan la provem amb el LMS d’SCORM, el Sample RunTime Environtment de ADL.*

**2.3.3.- <exitConditionRule>**

Igual que el <{pre | post}ConditionRule> l’únic que les seves accions s’aplicaran quan algun descendent de l’activitat intenti finalitzar.

**2.4.- <limitConditions>**

“*Ara per ara, ADL recomana utilitzar aquest element amb precaució. ADL te planejat continuar avaluant aquest element i el seu impacte en el seqüenciament, en les competències i les estratègies*.”

**Atributs:**

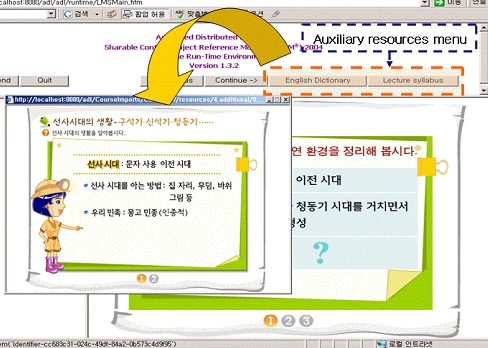
* **attemptLimit** (opcional, default 0): Aquest valor indica el màxim nombre d’intents per l’activitat.
* **attemptAbsoluteDurationLimit** (opcional, default 0.0): S’especifica la duració màxima que se li permet a un alumne perdre en un intent d’una activitat. Aquest límit només s’aplica mentres l’alumne està interactuant amb l’activitat, no mentres l’activitat està suspesa.

**2.5.- <~~auxiliaryResources~~>**

Una activitat pot tindre recursos auxiliars associats amb ella que li proporcionin recursos i serveis addicionals per l’alumne. IMS SS no defineix cap semàntica o cap significat per aquests recursos auxiliars. Tampoc no defineix quins recursos han de ser accessibles o com s’han d’utilitzar els recursos. L’únic que defineix l’IMS SS és el “concepte” de que pot haver-hi recursos auxiliars associats a una activitat.

SCORM no requereix que un LMS doni suport als *Auxiliary Resources*, i fins i tot si un LMS tria d’implementar-ho no hi ha cap garantia de interoperabilitat.

***Nota!:****Tot i així hi ha universitats o centres que si que han implementat versions pròpies del AuxiliaryResources. Un exemple es el projecte de KERIS (Korea Education and Research Information Service) que ho implementen (o això diuen) i així aconsegueixen afegir funcionalitats extres com diccionaris online, etc.*



<item identifier="item\_id\_01" identifierref="resource\_id\_01">

<title>Content with Aux. Resources</title>

<imsss:sequencing>

**<keris:auxResources>**

**<keris:auxResource title="English Dictionary" identifier="aR-01" identifierref="aR-01-Resource"/>**

**<keris:auxResource title="Syllabus" identifier="aR-02" identifierref="aR-02-Resource"/>**

**</keris:auxResources>**

</imsss:sequencing>

</item>

**2.6.- <rollupRules>**

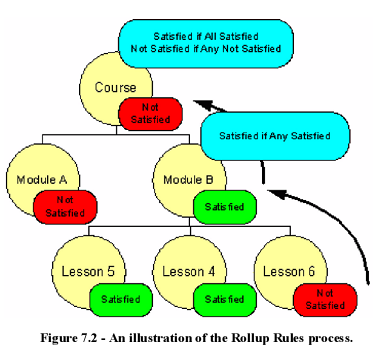
Fer un *rollup* significa “pujar” un valor/variable cap al nivell immediatament superior. Per tant totes les regles i accions que venen a continuació ens seran especialment útils per fer les *RulesCondition*s, ja que des dels nivells superiors sempre anirem a mirar si les activitats estan “satisfied”, “completed” o “attempted” per prendre una acció o altre (ja sigui pre, post o exitCondition).

**Atributs:**

* **rollupObjectiveSatisfied** (optional, default *true*):
* **rollupProgressCompletion** (optional, default *true*):
* **objectiveMeasureWeight** (optional, default *1.0*):

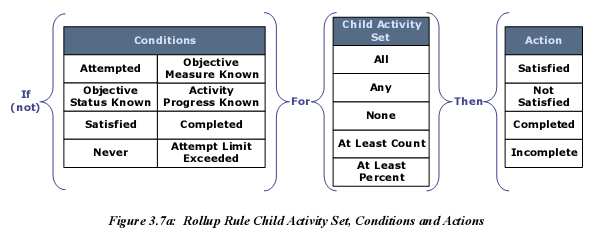
**Elements:**

* <rollupRule>



**2.6.1.- <rollupRule>**

Aquest element és el contenidor de cada regla “rollup” que és aplicada a una activitat. El format general per una regla pot ser definida de manera informal com: “Si *childActivitySet* , *conditiont* Aleshores *action*”.



**Atributs:**

* **childActivitySet** (optional, default *all*): Aquest atribut indica quins valors de les dades seran utilitzats per avaluar la condició de *rollup*. Valors { **all | any | none | atLeastCount | atLeastPercent }**.
* **minimumCount** (optional, default *0*): Aquest valor s’ha de concretar si amb el *childActivitySet* hem marcat *atLeastCount*.
* **minimumPercent** (optional, default *0.0*). Aquest valor s’ha de concretar si amb el *childActivitySet* hem marcat *atLeastPercent*. Valors [ **0.0000, 1.0000** ].

**Elements:**

* <rollupConditions>
* <rollupAction>

**2.6.1.1.- <rollupConditions>**

Conté la llista de condicions que seran aplicades a una sola regla de *rollup*.

**Atributs:**

* **conditionCombination** (optional, default *any*): Indica com les condicions de *rollup* seran combinades. Valors: { **all | any** }

**Elements:**

* <rollupCondition>

**2.6.1.1.1.- <rollupCondition>**

Identifica una condició que serà aplicada a una regla *rollup*.

**Atributs:**

* **operator** (optional, default *noOp)*: Serveix per negar una condició o deixar-la tal qual: **{ not | noOp }.**
* **condition** (required):Indica la condició de la regla. Valors: {
  + - **satisfied:** S’avalua a *true* si el estat de progressió dels objectius per els valors que ens arriben dels objectius associats amb les activitats dels fills estan a *true*.
    - **objectiveStatusKnown:** La definició es igual que per satisfied!
    - **objectiveMeasureKnown;** La condició s’avalua a *true* si la mesura de progressió de l’objectiu per els objectius associats als valors que ens arriben dels fills (*rollup*) es *true*.
    - **completed:** S’avalua a cert si el *Attempt Progress Status* per els fills de l’activitat es *true* i l’*Attempt Completion Status* pels mateixos fills també es *true*.
    - **activityProgressKnown:** S’avalua a cert si *l’Activity Progress Status* per els fills de l’activitat es *true* i l’*Attempt Completion Status* pels mateixos fills també es *true*.
    - **attempted :** S’avalua a cert si *l’Activity Progress Status* per els fills de l’activitat es *true* i l’*Activity Attempt Count*  es positiu.
    - **attemptLimitExceeded:** La condició s’avalua a certa si l’*Activity Progress Status* pels fills de l’activitat es true i el *Limit Condition Attempt Limit Control* pels mateixos fills també es *true* i si l’*Activity Attempt Count* es més gran o igual que el *Limit Condition Attempt Limit*.

**}**

**2.6.1.2.- <rollupAction>**

Identifica l’acció que serà aplicada a la regla de *rollup*. L’acció serà fer un *rollup* de la constant que vulguem.

**Atributs:**

* **action** (obligatori): Indica el valor que pujarem al nivell superior si la regla s’avalua a *true*. Valors: **{** 
  + **satisfied:** Posem les variables
    - *Objective Progress Status* *= true*
    - *Objective Satisfied Status = true*
  + **notSatisfied:** Activem les variables
    - *Objective Progress Status = true*
    - *Objective Satisfied Status = false*
  + **completed:** Posem les variables
    - *Attempt Progress Status = true*
    - *Attemp Completion Status = true*
  + **incomplete :** Posem les variables
    - *Attempt Progress Status = true*
    - *Attempt Completion Status = false*

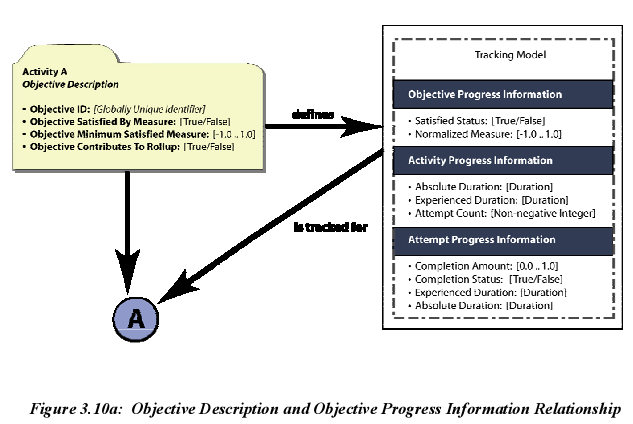
**}**

**2.7.- <objectives>**

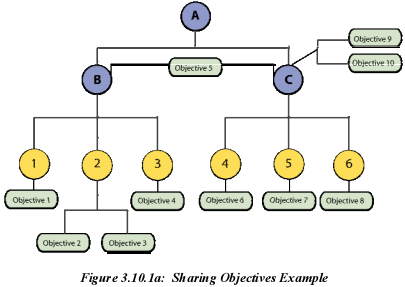
És el contenidor dels objectius que seran associats a una activitat. Cada activitat ha de tindre com a mínim un objectiu principal i després tants *objectives* com es vulguin.

Amb l’ introducció de l’IMS SS en l’SCORM, ara hi ha un mecanisme per associar els objectius d’aprenentatge amb les activitats.

SCORM no descriu com es defineix, s’interpreta o s’utilitza un objectiu d’aprenentatge, però per a propòsits de seqüenciament cada objectiu d’aprenentatge associat amb una activitat tindrà un conjunt d’estats que s’aniran enregistrant a mida que l’alumne vagi interactuant amb l’LMS, de manera que amb aquests estats podrem prendre decisions.



**Objectius Locals vs Objectius Globals Compartits:** Un objectiu local serà aquell que només serà accessible per l’activitat associada. Si una activitat vol compartir el seu objectiu local el que ha de fer es *mapejar* aquest contingut a un objectiu global compartit. Una activitat podrà tindre associats un o més objectius locals, tot i que només un serà primari, això vol dir que només un servirà per fer les condicions de *rollup*. D’igual manera, una activitat pot *mapejar* a un o diversos objectius globals. Un arbre d’activitat tindrà com a mínim un objecte global compartit.



**Elements:**

* <primaryObjective>
* <objective>

**2.7.1.- <primaryObjective>**

Aquest element identifica l’objectiu que contribueix al *rollup* associat amb l’activitat.

**Atributs:**

* **satisfiedByMeasure** (optional, default *false*): Aquest atribut indica que <minNormalizedMeasure> s’ha d’utilitzar en comptes d’un altre mètode per determinar si l’objecte associat amb l’activitat es satisfà. També conegut dintre del *Tracking Model* com a *Objective Satisfied Measure*.
* **objectiveID** (optional): És l’identificador associat amb una activitat i serà únic. Si el <primaryObjective> conté un mapInfo, aleshores aquest atribut és obligatori.

**Elements:**

* <minNormalizedMeasure>
* <mapInfo>

**2.7.1.1.- <minNormalizedMeasure>**

Aquest element identifica la mesura mínima per satisfer l’objectiu. El valor es normalitzat entre -1 i 1. Si el <primaryObjective> té com atribut *satisfiedByMeasure* aleshores l’LMS utilitzarà aquest valor per inicialitzar el cmi.scaled\_passing\_score.

En el *Tracking Model* rep el nom de *Objective Normalized Measure*.

**2.7.1.2.- <mapInfo>**

És el contenidor de la descripció mapejada de l’objectiu. Això defineix el mapeix de la informació de l’objectiu local d’una activitat de i per a un objectiu global compartit.

**Atributs:**

* **targetObjectiveID** (obligatori): L’identificador de l’objecte global compartit.
* **readSatisfiedStatus** (optional, default *true*). Es llegirà la informació de l’objecte global compartit i se li assignarà a l’objecte local. En aquest cas es llegirà l’estat de progrés.
* **readNormalizedMeasure** (optional, default *true*): Es llegirà la informació de l’objecte global compartit i se li assignarà a l’objecte local.
* **writeSatisfiedStatus** (optional, default *false*): Es transferirà l’estat de progrés de l’objecte local a l’objecte global compartit.
* **writeNormalizedMeasure** (optional, default *false*): Igual però amb el valor de la mesura normalitzada.

Per llegir Objectives Maps (*readSatisfiedStatus* and *readNormalizedMeasure*):

* Si existeixen múltiples elements <mapInfo> per a un objectiu, aleshores només un <mapInfo> pot tindre el *readSatisfiedStatus* a *true*. El mateix succeeix per a *readNormalizedMeasure*.

Per escriure Objectives Maps (*writeSatisfiedStatus* and *writeNormalizedMeasure*):

* Per una activitat, si hi ha múltiples objectius que tenen l’element <mapInfo> compartint el mateix *targetObjectiveID*, aleshores només un dels objectius pot tindre el *writeNormalizedMeasure* a *true*. El mateix succeeix amb l’atribut *writeSatisfiedStatus*.

**2.7.2.- <objective>**

És una llista d’objectius educacionals associats amb el node en el que es troba. Cada un dels objectius té un indicador que informa sobre l’estat del mateix i quant falta per a ser completat. Aquest element només pot existir si el *<primaryObjective*> ha estat definit.

**Atributs:**

* **satisfiedByMeasure** (optional, default *true*): Aquest atribut indica que *<minNormalizedMeasure>* serà utilitzat en comptes d’algun altre mètode per determinar si l’objectiu associat amb l’activitat ha estat satisfet.
* **objectiveID** (obligatori): L’identificador associat amb l’activitat.

**Elements:**

* <minNormalizedMeasure>
* <mapInfo>

**2.8.- <randomizeControls>**

Aquest element és el contenidor de les descripcions de com els fills de les activitats s’han d’ordenar duran el procés de seqüenciament. El que fa és jugar amb el *Tracking Model*, o sigui, dintre del *Activity State Information* el que fa és reordenar el paràmetre *Available Children*  que no és res més que un vector amb els fills que se’n poden derivar.

Òbviament aquest element només és aplicable a un cluster (un item no-fulla).

**Atributs:**

* **RandomizationTiming** (optional, default *Never*) : Ens indica quan s’ha de fer el reordenament. Valors {
  + *Never*: Mai
  + *Once* : Només una vegada abans d’accedir-hi per primer cop.
  + *OnEachNewAttempt* : S’aplica sempre abans d’un accés. }
* **ReorderChildren** (optional, default *false*) : Indica si s’han de reordenar els fills.
* **SelectCount** (optional, default *0*): Aquest atribut ens indica el número de fills que reordenarem dintre de tot el conjunt.
* **SelectionTiming** (optional, default *Never*) : Ens indica quan s’ha de fer el reordenament. Valors {
  + *Never*: Mai
  + *Once* : Només una vegada abans d’accedir-hi per primer cop.
  + *OnEachNewAttempt* : S’aplica sempre abans d’un accés. }

**Nota!** Ni a nivell teòric ni a nivell pràctic es pot apreciar la diferència entre l’atribut *RandomizationTiming* i *SelectionTiming* per tant, a efectes pràctics el que es recomana és tractar-los igual, mostrant un avís o un error si un paràmetre és diferent de l’altre.

**2.9.- <deliveryControls>**

Especifica si l’LMS ha d’enregistrar (o no) la informació associada amb l’SCO.

**Atributs:**

* **Tracked** (default *true*) ***:*** Indica si la informació sobre el *progrés dels objectus* i sobre les *vegades que s’ha accedit* a una activitat ha de ser guardada (true, false). D’aquesta manera la informació contribueix en el procés de *rollup* cap al pare de l’activitat.
* **CompletionSetbyContent** (default *false*): Indica si s’ha d’activar el paràmetre *Attempt Completion Status* per una activitat.
* **ObjectiveSetbyContent** (default *false*): El mateix però pel paràmetre *Objective Satisfied Status*.

**2.10.- <adlseq:constrainedChoiceConsiderations>**

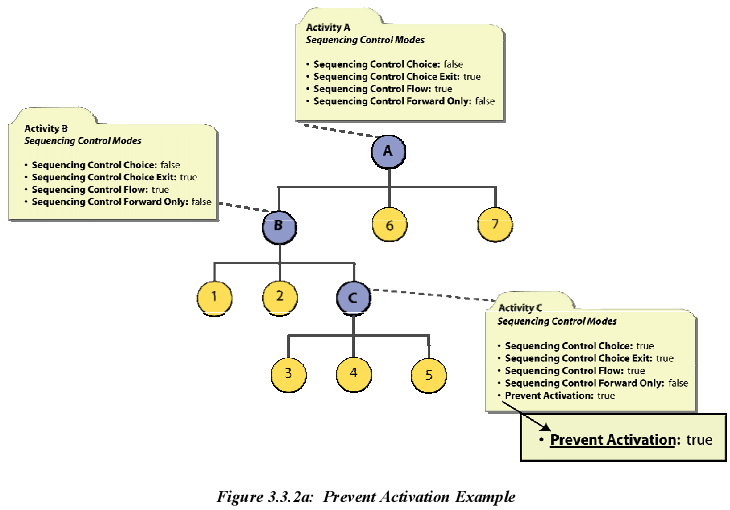
Aquest element és un contenidor per les descripcions de com s’ha de fer la tria de la navegació duran el procés de seqüenciament. Aquest element només s’aplica dintre de l’activitat en la que s’ha definit.

**Explicació més extensa**:

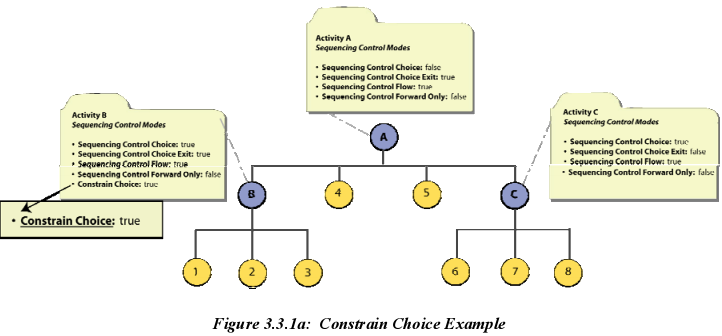
Per defecte, l’especificació IMS SS permet a totes les activitats situades a qualsevol punt de l’arbre d’activitat, els pares dels quals tinguin el *SequencingControlChoice* definit com a *true*, poder ser cridades com a resultat d’una tria de navegació. Mentres que aquesta flexibilitat és molt útil en algunes estratègies de seqüenciament, en algunes altres pot suposar un problema significatiu.

**Atributs:**

* **preventActivation** (optional, default *false*): Aquest atribut indica que no es podrà accedir als fills de les activitats fins que l’activitat actual no sigui el pare. Això evita que l’alumne s’endinsi massa endins del contingut si abans no ha superat una sèrie de prerriquisits.



* **constrainChoice** (optional, default *false*): Aquest atribut indica que només les activitats que són “lògicament” posteriors des de l’activitat pot ser seleccionable per una petició de navegació.

****

**2.11.- <adlseq~~:~~rollupConsiderations>**

És el contenidor per les descripcions de quan una activitat ha de ser inclosa en el procés de *rollup*.

**Atributs:**

* **requiredForSatisfied** (optional, default *always*): Indica la condició que fa que l’activitat sigui inclosa en l’avaluació del seu pare com una regla de *rollup* satisfeta.
* **requiredForNotSatisfied** (optional, default *always*): Igual, que *requiredForSatisfied*.
* **requiredForCompleted** (optional, default *always*): Igual, que *requiredForSatisfied*.
* **requiredForIncomplete** (optional, default *always*): Igual, que *requiredForSatisfied*.
* **measureSatisfactionIfActive** (optional, default *true*): Aquest atribut indica si la mesura ha de ser utilitzada per determinar “satisfacció” durant el *rollup* quan l’activitat és activa.

Cada un dels atributs definits fins ara pot tenir algun dels següents valors: { **always** (default) | **ifAttempted** | **ifNotSkipped** | **ifNotSuspended** }

**2.12.- <sequencingCollection>**

Hi ha situacions en que un conjunt d’informació de seqüenciament (aquella definida en *<sequencing*>) pot ser reutilitzada. Aquest element (<sequencingCollection>) pot actuar com a contenidor per un conjunt d’informació de seqüenciament. El reutilitzament succeeix quan l’atribut *IDRef* de *<sequencing>* fa referència a un atribut *ID* d’un element *<sequencing>* que és un fill d’un *<sequencingCollection>*. Si un element *<sequencing>* utilitza l’*IDRef* i en el seu interior defineix informació de seqüenciament, aquesta informació sobreescriurà la informació d’alt nivell que estigués definida prèviament.

**Característiques:**

* Si es vol utilitzar el <sequencingCollection> nomes se’n pot utilitzar un.
* Si un element <sequencing> és configurat perquè referenciï a un altre <sequencing> definit en una col·lecció de sequencing, l’atribut *IDRef* és obligatori i ha de referenciar al “ID” especificat pel <sequencing> que forma part del *sequencing collection*.
* Si l’element <seqüenciament> és definit com un fill d’un element <item> o <organitzation> aleshores l’atribut *ID* no està permès.
* Per tots els elements <sequencing> definits en una col·lecció de seqüenciament, l’ID és obligatori, en canvi l’IDRef no està permès.

**3.1.- Estratègies de seqüenciament**

Les opcions a l’hora de seqüenciar un contingut són enormes, i més si el contingut és força gran i volem aplicar restriccions ja sigui perquè l’alumne sigui forçat a realitzar alguna/es activitat/s o perquè volem que el nostre contingut educatiu s’adapti als diversos alumnes que tinguem.

En ADL ens proporciona una sèrie d’exemples sobre com seqüenciar un mateix contingut i també d’aquesta manera ens mostra les estratègies de seqüenciament que podem utilitzar de forma més habitual tot i que, com veurem, sempre es pot trobar solucions intermèdies més òptimes.

**3.1.1.- No sequencing**

La solució més fàcil és no aplicar cap tipus de seqüenciament. D’aquesta manera tindrem un curs on l’alumne es podrà moure lliurement per tot arreu. En aquest model el temes del curs seran assets i els sco seran els qüestionaris.

***Nota!*** *No cal que incloguem cap etiqueta <sequencing> si triem aquest model.*

**3.1.2.- Linear**

En aquest model fem que l’alumne hagi de recorre el curs de forma lineal, sense que pugui decidir res més que moure’s endavant o enrere. L’efecte de la “linealitat” l’aconseguim incloent unes poques instruccions de seqüenciament en cada mòdul temàtic:

<imsss:sequencing>

<imsss:controlMode choice="false" flow="true"/>

<imsss:rollupRules rollupObjectiveSatisfied="false"/>

</imsss:sequencing>

Els qüestionaris seran sco i per tant controlarem, com a mínim, que s’hagin realitzat.

<imsss:rollupRule childActivitySet="all">

<imsss:rollupConditions>

<imsss:rollupCondition condition="attempted"/>

</imsss:rollupConditions>

<imsss:rollupAction action="completed"/>

Amb aquest codi farem un rollup de *completed* si tots els fills (tots els qüestionaris) han sigut accedits com a mínim una vegada (*attempted*). D’aquesta manera, si controlem quan és “completed” ens és fàcil fer una condició de sortida quan tot sigui completat.

<imsss:sequencingRules>

<imsss:exitConditionRule>

<imsss:ruleConditions>

<imsss:ruleCondition condition="completed"/>

</imsss:ruleConditions>

<imsss:ruleAction action="exit"/>

</imsss:exitConditionRule>

</imsss:sequencingRules>

**3.1.3.- Linear with Controls**

És una modificació del model lineal explicat en el punt anterior. En aquest model el que varia es que la navegació passa a ocupar-se’n els propis sco, que en aquest cas seran tots els continguts, ja siguin qüestionaris o els mòduls educatius.

El codi extra que haurà de portar cada un dels continguts serà:

<SCRIPT LANGUAGE=JAVASCRIPT SRC="util/Photoshop\_APIWrapper.js"></SCRIPT>

<SCRIPT LANGUAGE=JAVASCRIPT SRC="util/Photoshop\_SCOFunctions.js"></SCRIPT>

<SCRIPT LANGUAGE=JAVASCRIPT>

//sets the nav event to previous and then calls Terminate

function doPrevious()

{

//set the lesson\_location to "" so that the "previous" call works

doSetValue( "cmi.location", "" );

exitPageStatus = true;

doSetValue("adl.nav.request","previous");

doTerminate();

}

function goNext()

{

exitPageStatus = true;

document.location.href = "Lesson2.htm";

}

</SCRIPT>

Com a part rellevant dintre del manifest trobarem l’opció específica per amagar les opcions de navegació a l’usuari.

<adlnav:presentation>

<adlnav:navigationInterface>

<adlnav:hideLMSUI>previous</adlnav:hideLMSUI>

<adlnav:hideLMSUI>continue</adlnav:hideLMSUI>

</adlnav:navigationInterface>

</adlnav:presentation>

Aquestes opcions es troben en el punt *5.6. User Interface (UI) Devices for Navigation* del llibre de *Sequencing and Navigation* de l’SCORM 2004.

**3.1.4.- Linear Choice**

En aquest model oferirem una mica més de llibertat perquè l’alumne pugui triar quin dels mòduls del curs vol realitzar. Un cop seleccionat un mòdul d’aprenentatge aplicarem totes les restriccions possibles perquè l’hagi de realitzar de principi a fi. Per tant aplicarem els mateixos elements de seqüenciament vistos en els models lineals, només ens posarem més restrictius en arribar a l’examen, ja que no li donarem ni l’opció de sortir un cop començat.

<imsss:controlMode choice="false" choiceExit ="false" flow="true" forwardOnly="true"/>

***Nota****!: En el software de Sample RTE de ADL no funciona be aquesta opció de choiceExit!*

Utilitza també un altre element que es el *deliveryControls* amb l’opció *tracked = “false”*. Amb aquesta opció a *false*, teòricament, el propi SCO ja no guarda informació (en aquest cas sobre els estats) però igualment hem d’utilitzar el *rollupRules* amb el paràmetre *rollupObjectivesSatisfied = “false”* per evitar que més tard, quan el pare de l’activitat faci la comprovació sobre si tots els objectius han estat “attempted” o “satisfied” aquests no es compleixin simplement perquè des d’aquesta part del codi no li estem enviant informació.

De fet el codi funcionaria igual si eliminéssim el *deliveryControls*, ja que des d’aquest element simplement evitem que es guardi la informació, però tan si es guarda com si no, després qui decideix si s’ha de comprovar les variables o si simplement s’assumeix com a *true* (*rollupObjectivesSatisfied*) és el *rollupRules*, per tant a la pràctica ens és igual tenir-lo o no. Per tot això podrem dir que és un element prescindible des del punt de vista del LMS.

**3.1.5.- Constrained Choice**

En aquest model donem una llibertat total a l’alumne per saltar a qualsevol activitat de dintre de qualsevol mòdul. L’única restricció que hi posem és que només pot saltar al mòdul lògic següent i al mòdul lògic anterior. D’aquesta manera intentem restringir una mica que no pugui saltar directament a l’última lliçó.

Ho aconseguim amb el següent element, que precisament només té aquesta propietat:

<adlseq:constrainedChoiceConsiderations constrainChoice = "true" />

***Nota****!: Si els mòduls temàtics que tenim són molt grans no ens aportarà gran cosa aquesta propietat, ja que amb pocs “salts” l’alumne podrà anar a qualsevol part del temari, tan del principi com del final. No haurem aconseguit restringir res.*

*L’ utilitat augmentaria si utilitzéssim una granularitat més fina, es a dir, si en comptes d’ utilitzar una divisió temàtica molt genèrica utilitzéssim una molt concreta. D’aquesta manera el fet que l’alumne pogués fer “salts” només cap al tema següent o a l’anterior no ens ha de preocupar tant si tenim una divisió temàtica (per dir alguna cosa) de 50 mòduls.*

***Modificació!*** *Per millorar aquest sistema podríem pensar en algun mètode per evitar que l’alumne no es salti massa temari sense llegir. Una forma d’aconseguir-ho podria ser utilitzant objectives i guardant els “attempted”, d’aquesta manera podríem pensar en un bloqueig si entre els 5 primers mòduls no ha accedit a més d’un 60% del temari.*

**3.1.6.- Remediation**

És un model d’ensenyament més dirigit (punt 1.1 , Seqüències adaptatives) on vigilem més el progrés de l’alumne. La seqüència de navegació seria:

1. L’alumne recorre linealment tota la teoria.
2. Realitza tots els exàmens.
3. Repeteix la part de teoria associada amb els exàmens que ha suspès.
4. Repeteix els exàmens de la part suspesa.
5. Finalitza encara que el resultat no sigui satisfactori.

En aquest model, tot i que sembli el contrari, la complexitat no és gaire gran, ja que l’únic que fem és repetir els temes dintre del manifest afegint-hi pre-condicions per saber si s’ho ha de saltar o no. Per tant, és un “truc” simple i força efectiu.

<imsss:sequencingRules>

<imsss:preConditionRule>

<imsss:ruleConditions>

<imsss:ruleCondition condition="satisfied"/>

</imsss:ruleConditions>

<imsss:ruleAction action="skip"/>

</imsss:preConditionRule>

***Nota!*** *Aquest efecte de “loop” podria ser més efectiu si s’utilitzés una postConditionRule amb l’opció de Retry o RetryAll si la condició és “notSatisfied”. El principal problema és que les postConditions NO funcionen en el SampleRTE, tot i que segons l’especificació hauria de donant-se una solució més potent tal i com acabo d’explicar.*

**3.1.7.- Competency Assessment**

És una variació del *Remediation*. En aquest model fem un pre-test i després passem a la part de teoria en el que l’alumne ha fallat el pre-test, finalitzant amb un examen de la pròpia part suspesa. No cal dir que el “truc” s’aconsegueix exactament igual que amb el *Remediation*, i és repetint codi en l’organització del manifest de manera que, teòricament, el curs hauria de ser més llarg perquè es repeteix més temari però a la pràctica al saltar-nos les parts ja aprovades en els pre-test doncs queda un curs molt més curs i més adaptat a l’alumne.

**4.- Conclusions i proves pròpies de l’autor**

Tot aquest sistema de seqüenciament està pensat per donar una gran llibertat al professor o al creador del curs perquè pugui definir la forma en que els alumnes recorreran el material. Per això, com en gran part de les coses, hi ha gustos per a tot i l’ideal és que l’especificació, i a la pràctica l’LMS, doni suport a suficients elements com perquè es pugui seqüencià el curs exactament com el creador ho tenia en ment.

En els apartats anteriors ja s’han anat comentant aspectes de l’especificació que a la pràctica (en el Sample RTE de l’ADL) no funcionen. Hi ha fins i tot parts que la pròpia especificació et recomanen no utilitzar ja que encara no està del tot estudiat l’element en qüestió. S’ha de tindre en compte que l’especificació del seqüenciament de l’SCORM es basa en el de l’IMS, per tant, qualsevol cosa que no estigui del tot ben definida en l’IMS serà complicat trobar-ho correctament o amb garanties a SCORM.

**4.1.- Proves pròpies**

Per la realització d’aquest estudi s’ha testejat tots els exemples que dona ADL i sobre alguns d’aquests s’ha fet modificacions per intentar testejar més elements de la pròpia especificació. Així aconseguim comprovar dues coses, per una banda si realment funciona tot i per l’altre si és molt fàcil aplicar seqüenciament a una organització ja feta.

**4.1.1.- Modificació del *Linear Controls***

El model del *Linear Controls* es força bo (a gust de l’autor), ja que al situar els controls de navegació a l’interior del contingut del temari d’alguna forma ja estàs obligant a l’alumne a recorre-ho. Sobre aquesta base, una de les coses que no em semblaven correctes era donar per completat els qüestionaris només basant-nos en el criteri de si l’alumne hi ha accedit (*attempted*). Segons aquest criteri un alumne podia fallar totes les preguntes però superar-lo igualment només perquè ha accedit a totes les qüestions.

La solució serà tan simple com exigir que s’hagi d’haver accedit a totes les qüestions com a mínim un cop i que hagi aprovat un 50% d’aquestes.

Per fer-ho anirem per parts: Primer de tot afegim les *rollupCondition* a l’activitat “pare” dels qüestionaris. El *rollupCondition* extra el que farà serà un *rollup* de “*satisfied*” o de *“notSatisfied”*. També tindrem el *rollup* per pujar l’estat “*completed*” o “*incomplete*” segons si s’ha accedit a tots.

<imsss:rollupRules >

<imsss:rollupRule childActivitySet="all">

<imsss:rollupConditions>

<imsss:rollupCondition condition="**satisfied**"/>

<!--

Si tots els fills estan "satisfied" fem pujar aquest valor

cap amunt.

/!-->

</imsss:rollupConditions>

<imsss:rollupAction action="**satisfied**"/>

</imsss:rollupRule>

<imsss:rollupRule childActivitySet="all">

<imsss:rollupConditions>

<imsss:rollupCondition condition="**attempted**"/>

</imsss:rollupConditions>

<imsss:rollupAction action="**completed**"/>

</imsss:rollupRule>

<imsss:rollupRule childActivitySet="any">

<imsss:rollupConditions>

<imsss:rollupCondition operator="not" condition="**attempted**"/>

</imsss:rollupConditions>

<imsss:rollupAction action="**incomplete**"/>

</imsss:rollupRule>

</imsss:rollupRules>

<imsss:objectives>

Com a segon pas el que hem de fer és controlar aquests estats des del nivell de l’arrel (el pare de tots) segons els estats que ens arriben. D’aquesta manera podrem controlar si finalitzem o no (donant el curs per completat i satisfet). Això a la vegada ens implica dues classes de controls diferents, per una banda el control del *rollup* de tots els estats i per l’altre la condició que activarem segons els valors dels estats d’aquest últim *rollup*.

Part del *rollup*:

<imsss:rollupRules >

<imsss:rollupRule childActivitySet="all">

<imsss:rollupConditions>

<imsss:rollupCondition condition="completed"/>

</imsss:rollupConditions>

<imsss:rollupAction action="completed"/>

</imsss:rollupRule>

<imsss:rollupRule childActivitySet="any">

<imsss:rollupConditions>

<imsss:rollupCondition operator="not" condition="completed"/>

</imsss:rollupConditions>

<imsss:rollupAction action="incomplete"/>

</imsss:rollupRule>

<!--

Afegir per controlar els "completed" i els "incompleted"fi afegit.

/!-->

<imsss:rollupRule childActivitySet="all">

<imsss:rollupConditions>

<imsss:rollupCondition condition="satisfied"/>

</imsss:rollupConditions>

<imsss:rollupAction action="satisfied"/>

</imsss:rollupRule>

<imsss:rollupRule childActivitySet="any">

<imsss:rollupConditions>

<imsss:rollupCondition operator ="not" condition="satisfied"/>

</imsss:rollupConditions>

<imsss:rollupAction action="notSatisfied"/>

</imsss:rollupRule>

</imsss:rollupRules>

Part de la condició:

<imsss:sequencingRules>

<imsss:exitConditionRule>

<imsss:ruleConditions conditionCombination="all">

<imsss:ruleCondition condition="**satisfied**"/>

<imsss:ruleCondition condition="**completed**"/>

</imsss:ruleConditions>

<imsss:ruleAction action="**exit**"/>

</imsss:exitConditionRule>

Cal advertir que amb només amb això que hem fet NO funcionarà, ja que aquesta darrera condició no és donarà mai. Això és perquè sempre ho tindrem com a “*notSatisfied”* i com a “*incomplete*”. I tot perquè quan hem fet l’últim *rollup* ens em basat en els estats que ens pugen de TOTS els fills, i com que estem al nivell de *<organization>* , els fills no només seran els qüestionaris (que aquests si que ens retornaran els estats be) sinó també tindrem com a fills els diversos mòduls temàtics que al no tindre cap *rollup* implementat mai retornaran els estats que ens interessen.

Per tant com a últim pas hem de vigilar que els mòduls temàtics no contin per aquesta condició global. I això ho aconseguim amb un dels paràmetres del *rollupCondition*:

<imsss:rollupRules rollupObjectiveSatisfied="false" rollupProgressCompletion="false"/>

Aquesta línia anirà a tots els mòduls temàtics que no vulguem controlar el seu estat ni la seva progressió.

**4.2.- Conclusions: Elements més importants**

Després de totes les proves i d’haver estudiat tots els elements que formen part de l’especificació he arribat a la conclusió de que només alguns poden ser útils. Aquests serien:

<sequencing>

<controlMode>

<sequencingRules>

<preConditionRule>

<postConditionRule>

<exitConditionRule>

<ruleConditions>

<ruleCondition>

<ruleAction>

<linitConditions>

<rollupRules>

<rollupRule>

<rollupConditions>

<rollupCondition>

<rollupAction>

<objectives>

<primaryObjective>

<mapInfo>

<adlseq:constrainedChoiceConsiderations>

<sequencingCollection>

Representar-ho de forma indexada es només per fer notar la relació jeràrquica que tenen els elements entre ells.

**Justificació:**

Com a elements principals necessitem les *ruleConditions* perquè sense aquestes no tenim seqüenciament. A la vegada perquè les *ruleConditions* funcionin necessitem declarar, assignar i controlar els *objectives* , i per això necessitem tant les *rollupRules* com el *primaryObjective*.

Els elements més discutibles serien els <*mapInfo*>, <*limitCondition*>, <*adlseq*:*constrainedChoiceConsideration*> i <*sequencingCollection*> que passaré a justificar tot seguit:

* <**mapInfo**>: La seva funcionalitat bàsica és la d’enllaçar amb un objectiu global per a compartir estats, per això a l’hora de programar es pot imaginar tan simple com una espècie d’apuntador.
* <**limitCondition**>: Tot i que en les *ruleCondition* trobem el paràmetre *attemptLimitExceeded*trobo que es molt més simple i clar utilitzar, tant pel programador com pel creador del curs, utilitzar el *limitCondition* amb un paràmetre numèric.
* <**adlseq**:**constrainedChoiceConsideration**>: Amb els seus paràmetres podem fer una petita limitació simple sobre l’accés a l’arbre d’activitats, tant a l’amplada com en la fondària. Com a programador crec que un cop es té implementat una estructura que emuli l’arbre d’activitats controlar una selecció com aquesta pot ser quasi trivial.
* <**sequencingCollection**>: És una forma còmoda de reutilitzar codi, i també crec ens podem evitar possibles errors humans si facilitem que el creador del curs, davant d’un seqüenciament llarg en comptes de reutilitzar codi hagi de fer *copy&paste* moltes vegades. També disminuïm la grandària del manifest fet que ens anirà be quan haguem de recórrer seqüencialment per construir l’arbre d’activitats.

**4.3.- Conclusions: Estratègies de seqüenciament més importats**

Totes les opcions que he explicat en aquest estudi són perfectament implementables i constitueixen les estratègies més simples i més interessants que ens permet el seqüenciament d’SCORM a dia d’avui.

Simplement faré uns pocs comentaris a mode d’apunt sobre en quina situació i de quina manera ens surt a compte implementar un o altre.

**No sequencing**

L’estratègia de no aplicar cap tipus de seqüenciament és, òbviament, l’estratègia més fàcil. Tot i això jo no ho recomanaria per cap curs que no fos merament consultiu i on és poc probable que l’alumne/l’usuari es miri mai tot el temari ja que per un curs estàndard estaríem desaprofitant els recursos que el seqüenciament ens ofereix. Estic pensant, per exemple, en un “curs” sobre la documentació d’una llibreria o d’un sistema concret on l’alumne només hi accedirà de manera puntual i per revisar la part que l’interessa exclusivament.

**Linear**

El fet d’aplicar linealitat restringeix molt la complexitat de l’activitat didàctica, ja que no es gaire recomanable crear un curs molt extens i obligar a l’alumne a recorre’l linealment sense donar-li cap opció. El fet de linealitzar les coses ens donarà problemes si l’usuari vol revisar punts concrets que ja ha donat.

Pel professor o creador del curs el codi extra que ha d’afegir en el manifest per aconseguir aquest efecte és mínim.

**Linear with Controls**

Com a problemes els mateixos que implica el fet de linealitzar un curs. Pel creador del curs se li demana un esforç extra ja que no només ha d’afegir un petit codi en el manifest sinó que ha de convertir quasi tots els Assets en SCOs per tal que puguin sol·licitar la navegació a nivell d’SCO i no a nivell de LMS.

A nivell educatiu pot ser interessant ja que d’aquesta manera obliguem a que l’alumne hagi de revisar el document encara que només sigui per continuar endavant.

**Linear Choice**

La complexitat no augmenta gaire més respecte els altres models de seqüenciament lineal, ja que aquí l’únic que hem fet és crear una divisió temàtica més gran de l’estructura del curs.

**Constrained Choice**

L’efecte i l’implementació d’aquest seqüenciament és força senzill però a la vegada és força pobre si no se li afegeix cap altre seqüenciament.

Tots aquests seqüenciaments són força simples pel creador del curs, però això variarà molt depenent de com pensem el sistema de testos/exàmens i de les condicions o premisses que vulguem controlar sobre l’alumne. La modificació del manifest per afegir opcions per controlar els qüestionaris no és una tasca ni fàcil ni agradable, i menys per una persona que no hagi estat molt instruïda per fer-ho!

**Remediation i Compentency Assessment**

Aquestes dues estratègies de seqüenciament serien les més complexes d’implementar ja que hem de pensar que redirigim a l’alumne cap al contingut que nosaltres vulguem depenen dels estats dels testos, i això vol dir una càrrega d’instruccions en el manifest i una complexitat força elevada.

Tot i això, tan un com l’altre es poden simplificar i fer que siguin més “lògicament correctes” si els implementem utilitzant postCondicions i no amb preCondicions (que és com ho fan la gent d’ADL). **5.- Bibliografia**

* SCORM® 2004 Content Aggregation Model (CAM) Version 1.3.2 – DRAFT
* SCORM® 2004 Sequencing and Navigation (SN) Version 1.3.2 – DRAFT
* SCORM® 2004 Photoshop Examples Version 1.1
* Estudio de la propuesta IMS de estandarización de enseñanza asistida por computadora, Borja Manera Iglesias.
* KERIS en l’ADL:
  + <http://www.adlnet.gov/downloads/downloadpage.aspx?ID=90>
  + <https://keris.dev.java.net/>