Integration of IMS Learning Design and eGames

Pablo Moreno-Ger
(pablom@fdi.ucm.es)

Daniel Burgos
(daniel.burgos@ou.nl)
Objective: Integration of IMS Learning Design and educational games

- Definition of a communication model
- Creation of sample games
- Creation of a sample Unit of Learning
- Implementation of the communication model
- Evaluation of the proposal

General data

- Timeframe: 6 weeks
- Host researcher: Daniel Burgos
- Funding: <e-UCM> research group at Universidad Complutense de Madrid
Established in the early 90’s under the direction of Baltasar Fernández-Manjón and Alfredo Fernández-Valmayor

Areas of interest

- E-Learning tools
- E-Learning standards and specifications
- Markup languages
- Alternative approaches to learning
  - Skill development
  - Mobile Learning
  - Game-based Learning
Prototype of a LMS supporting a simplified version of the original EML
- Fully-featured LMS constructed around IMS Specifications (CP, SS, QTI, LIP)
  - Design and architecture driven by the specifications
  - Test-bed for further developments

Contents

- Presentation
- About <e-Ucm>
- <e-Adventure>
- Integration with IMS Learning Design
- Preliminary results and future work
- Conclusions and discussion
Web-based editor / player for IMS QTI 2.0
- Service oriented
- Can work stand-alone or integrated in other platforms
- Authoring, playing, question pools, import/export (WCT Quiz, HotPotatoes)
Graphical editor for IMS-Learning Design
- Developed as a Eclipse plug-in
- Uses UML diagrams as graphical notation
The <e-Game> research area

- Research & Development work in the field of games and education

- Addresses four main research questions
  - Educational game design
  - Authoring process
  - Assessment & grading
  - Integration with existing courseware

- <e-Adventure> as development project
Complex design equation
- Educational value
- Fun and engagement
- Eye-candy

Looking for an appropriate genre…

Point and Click adventure games

(images © LucasFilm Ltd.)
Designing Educational Adventure Games

- Study of the educational potential of the genre
  - Guidelines for writing interesting adventure scripts with educational value
  - Study of the different elements of the genre and description of their educational potential
    - What elements are key for the fun-factor?
    - What elements are key for the learning outcomes?
  - Full storyboards for educational adventure games in different fields

- Conclusions: The genre itself is suitable for learning, but we can add additional elements in the design
  - Three levels of description of game elements
  - In-game books that can include sections, titles, images and (eventually) hyperlinks
  - Tagging of game events according to their educational value
Provide a simple authoring process to facilitate the incorporation of teachers or other domain experts in the development

- Domain Specific Language to describe educational point n’ click adventure games
- XML language
- Balance between “simple” and “powerful”

Definition of a development process

1. Provide the script / storyboard
2. Add descriptive markup to the storyboard
3. Create / gather art assets (sounds, music, graphics, animations…)
4. Package the marked up script and the assets and feed them to the <e-Adventure> engine
5. The engine generates and executes the game
Independent definition of assessment rules
- The instructor decides which game events are pedagogically relevant
- Separate XML file with assessment rules included in the game package

Monitorization the in-game experience
- The engine monitors the activity of the learner within the game and matches it to the assessment rules
- The activation of the assessment rules is logged and written to an XML file
- Generation of reports with varying degrees of detail
- The reports are used for grading
Integration with existing courseware

- <e-Adventure> games as Learning Objects
  - Potential integration in standards-compliant LMS

- Different integration approaches
  - Black-box integration
    - The game is launched as any static content
  - Deep integration
    - The engine includes an API to communicate with external systems

- Bi-directional communication
  - The engine can import an initial state and adapt the game according to input from the LMS (user profile)
  - The engine can broadcast the activation of the assessment rules to the LMS
Integration with IMS Learning Design

- Two main objectives for this stay
  - The design of a *generic* bidirectional communication method between an IMS Player and educational games
  - The implementation of this model enabling the communication between <e-Adventure> and a CopperCore + CCSI + SLeD environment

- Two resulting products
  - Modified versions of SLeD and CCSI enabling generic bidirectional communication with games
  - An environment to be deployed along CCSI enabling the specific communication with <e-Adventure> games
Design assumption
- Contents can be launched by interpreting some HTML code in the learner’s browser
- Flash, Applet, Java WebStart …
Games can report what happens while the user interacts with the content.

But, what information should be reported?

- The most common unit of information in a UoL are its properties
- Games are studied in terms of their states
- Proposal: Associate specific game states with property-value pairs in the UoL

When the games enter states associated with properties, the event is notified to the engine

- Design decision: Notify the event when it happens, don’t wait until the game ends
Architecture of the solution

Contents
- Presentation
- About <e-Ucm>
- <e-Adventure>
- Integration with IMS Learning Design
- Preliminary results and future work
- Conclusions and discussion

LD Engine (CopperCore)  QTI Engine  Game service provider

CopperCore Service Integration (CCSI)

Service-based Learning Design Player (SLeD)

Learner's browser

AJAX
Games are a rich and adaptable medium
- Let’s use this to our advantage

Adaptable games
- Rich content can query the player for the state of the properties and act in consequence
- Games can be easier/harder, longer/shorter, internationalized, omit uninteresting levels, change the possible end-states, etc.

Design decision: The game changes its initial state according to the properties of the Unit of Learning
Get Initial State

Contents
- Presentation
- About <e-Ucm>
- <e-Adventure>
- Integration with IMS Learning Design
- Preliminary results and future work
- Conclusions and discussion

LD Engine (CopperCore)  QTI Engine  Game service provider

CopperCore Service Integration (CCSI)

Service-based Learning Design Player (SLeD)

Learner’s browser
Contents

- Presentation
- About <e-Ucm>
- <e-Adventure>
- Integration with IMS Learning Design
- Preliminary results and future work
- Conclusions and discussion

Preliminary Results (1)

- Modified version of SLeD
  - Recognizes a new content type “gamecontent”
  - Tries to locate (through CCSI) a game service to request the HTML code required to launch the game

- Modified version of CCSI
  - New generic adapter interface: GameAdapter
  - New adapter implementation: EAdventureAdapter

- eAdventureWebSupport web application
  - Provides the game service that generates the HTML code for <e-Adventure>
  - Provides listeners for the AJAX-invocations coming from the games
  - Uses CCSI to push and retrieve information from the CopperCore engine
  - Can be deployed in the same container as CCRT and SLeD

- AJAX implementation of the communication API for generic game applets
Sample Unit of Learning: The art and craft of chocolate

Adaptable <e-Adventure> game
Future work

- Inclusion of the modifications in future versions of SLeD and CCSI

- Publish the eAdventureWebSupport application as open source application

- Publish the communication APIs and data models

- Evaluation of the integration model
  - Integration of games other than <e-Adventure> games using the generic products obtained in this work

- Global reflection
  - Can this be extended to include all kinds of “intelligent” content?
Conclusions and discussion

- Can the activities of a Unit of Learning be games? YES
- Does this have any advantages? YES
  - Adaptable content
  - Higher interactivity
  - Live report of results
- Does this work for any kind of game?
  - Most games can be deployed and launched as black boxes
  - Establishing a communication requires the game to be aware of its environment and to participate in the communication
- Value for CopperCore and CCSI
  - The communication CC → EA builds on the experience integrating CC and QTI
  - The communication EA → CC is new and should be evaluated
- Value for SleD
  - An experience in the introduction of a new content type
  - How can this procedure be made more modular?
Integration of IMS Learning Design and eGames

Pablo Moreno-Ger
(pablom@fdi.ucm.es)

Daniel Burgos
(daniel.burgos@ou.nl)