Models and Technology for Open Structured Learning

Daniel K. Schneider

TECFA
Faculté de Psychologie et des Sciences de l’Education
Université de Genève

http://tecfa.unige.ch/tecfa/talks/schneide/tele99/
The Menu

1. A few words on teaching and learning as we see it  
2. Software for the Virtual Campus (quickly, sorry)  
3. TECFA’s Campus project: structure and collaboration  
4. Outlook on technology  
5. Conclusion

Slides and abstract:
http://tecfa.unige.ch/tecfa/talks/schneide/tele99/
1. A few words on teaching and learning as we see it

1.1 Main focus on Learning

- Using Internet for changing education
- Learner & activity centred
1.2 The Learning Environment

These are functions, not necessarily people, organizations, things, ....
1.3 Acknowledgment of variety

“Learning” is not a general thing:

• Learning Types: Attitudes, Factual Information, Concepts, Reasoning, Procedure Learning, Problem solving, Learning Strategies,...

• Degrees: beginners vs. advanced, easy vs. difficult, personal preferences

• All major learning paradigms have a point and a sound theoretical foundation (Constructivism, Instructionalism, Socio-culturalism etc.)

Educational paradigms

• all insist on some form of guidance or scaffolding

• require some form of structuring
Common features:

• **Reading** is NOT learning
  => On must “do” to learn

• ... and **Internet** is NOT interactive (per se)
  => What counts is activity
    (triggered by the system and/or by the task)

• Most people need **guidance** to achieve **instructional goals**
  => External conditioning (teaching, monitoring)

**Virtual Campus software:**

→ must be **flexible** to accommodate
  various learning and teaching paradigms
1.4 On misconceptions

Hypertext ??

- There is no empirical evidence that Hypertext (HTML) really helps learning
- Note: a good book is also a Hypertext

Is individual learning pace important ??

- People who do different things can’t talk to each other
- Cost is very high and benefits are minor

NO: design must start with the learner activity!

- There can be hypertext and accommodations for learning style or pace
- You can give flexible assignments (good for motivation)
2. Software for the Virtual Campus (quickly, sorry)

- We only get “some of it”
- There isn’t any globally satisfactory virtual campus software
2.1 **Standard Internet Tools**

1. **WWW** (hypertext), e.g. for:
   - planning, curricula, agendas, assignments
   - texts, manuals, resources and pointers
   - assignments (student productions)
   - collaboration within group projects

2. **Email**, e.g. for:
   - agenda planning (teacher)
   - search for information (student)
   - information about updates (student, teacher)
   - short comments (teacher)

3. **Discussion Forums**, e.g. for:
   - debates (about articles or themes)
   - technical Q/A
   - student-student help (!)

4. **The MOO** (advanced chat)
   - urgent things
   - co-presence (common virtual space, radio channels)
   - virtual meetings
Problems with the “basic four”

• “Please send Email” does not work very well
  • Typically, students hesitate to ask questions (or too late)
  • Teachers are overloaded: overlook details, do not prompt

• “Use the forums”
  • Need constant coaching/animation else they die
  • Students think their problem is private
  • Slowness of WWW-based tools, features missing within standard News

• “Be on the MOO”, chat, etc.
  • Connection costs
  • difficulty to find a date for large group meetings
  • Not sexy enough
  • No teacher animation (lack of time)

• Web pages
  • updating
  • structuring and navigation, meta-information, etc.
2.2 Mainstream virtual campus software

• See: [http://www.ctt.bc.ca/landonline/](http://www.ctt.bc.ca/landonline/) (best feature comparison)

Some market leaders:

• WebCT: [http://www.webct.com/](http://www.webct.com/)
• Topclass: [http://www.wbtsystems.com/](http://www.wbtsystems.com/)
• Cose: [http://www.staffs.ac.uk/COSE/](http://www.staffs.ac.uk/COSE/) (new product)
• others on available .....
Main features:

• Asynchronous Communication: email, forums
• Synchronous Communication: chat, whiteboard, teleconferencing, group browsing, application sharing,...
• Student tools: home page, self tests, bookmarks, progress tracking, ....
• Student Mgmt Tools: progress tracking, on-line grading, ....
• Lessons tools: authoring (structured HTML text), testing (e.g. JavaScript generators)
Main differences:

- Authoring (ability to import, hand coding, HTML forms-based editing tools)
- Student Management (at course- and general level)
- Application sharing and co-editing
- TeleConferencing
- Flexibility and possibility to adapt the software

Missing features:

- Virtual Space
- Interactive applications (besides HTML forms)
- Interactive shared applications
- “openness” (most are quite locked)
Our evaluation of WebCT and TopClass

• lack of flexibility (closedness)
• Incorporated tools don’t do much more than separate tools
• Student management is ok (no task management tools)

Major benefits:
• integrated tools (forums, index, quiz engine)
• automatic coherent look & structured text
• student management above task level

..... nice & flexible instructionalism, but not spectacular
2.3 Virtual Environments

- featuring: spacial organization, avatars (identities), building spaces, forums, sophisticated chat and more
- Varieties: MUDs, VRML-based, custom 2D, custom 3D

Why multi-user environments?

- People collaborate better with a feeling of co-presence (same space = same context)
- People working in pairs can infer what the other is doing from his location
- 3D space metaphors might make hyperspace navigation easier
- Community building ("feeling there")
2/3D Environments

... under development

... potential is not clear yet (besides community)
2.4 Summary for all tools

Support for:

- Traditionalist teaching
  - Mostly reading and answering questions

- Resource and project-based learning
  - Students do things outside the environment
  - Communication & collaboration

Common problem: students need a lot of attention

- They have a tendency to get lost in text space (they rather read than produce in any environment)
- We don’t always cope with student management

Build more structured activity into the virtual campus
3. TECFA’s Campus project: structure and collaboration

Overall Design

• Spacial metaphor
  • Zones = courses
  • buildings = main activities
  • rooms = sub activities

• Featuring a growing amount of structured learning/teaching activities
3.1 Our growing activity collection

Features (each has some)

- Sharing
- Project monitoring
- Construction
- Discussion & argument
- Information classification
- Navigation tool construction

The principle:
3.2 **Zone Example: Educational software course**

**Main buildings (activities):**

- Argue & Graph
- Mister QCM
- Design Studio
- Concept Factory

- A **building for each activity** (teaching/learning)

- Each building:
  - sets up various tasks with special tools
  - contains the necessary information
  - features collaboration
3.3 Activity examples

A. The Courseware Studio
   - Goal: Learn to design edu software

Phases (learner activities):
1. Topic specification
2. Objectives specification
3. Content analysis
   (analyse the objectives)
4. Operational pedagogical objectives spec
   (of other pairs)
5. Architecture and scenario spec
6. Learning activities spec
   (of 3 learning activities)
7. Preparation of Materials
8. Development of a module (with Authorware)
9. Assembly of all the modules
B. Argue & Graph

- Goal: Support classroom discussion about pedagogical style

Scenario:

1. Students fill in a questionnaire about design choices
   - Campus produces summary information (incl. graphs)
2. Collaborative fill in
   - Teacher selects opposite pairs
   - Pairs fill in questionnaire
   - Campus produces summary + details
3. Classroom discussion
4. Synthesis (HomeWork)
   - Each student writes a text

(1) Fill in
(2) Discussion
(3) Fill in together
(4) Discussion
(5) Synthesis

On the Web
C. The Iconometer

- Goals:
  - Test icons used in web pages
  - Learn about “monosémie”

Scenario

1. Look at an icon
2. Formulate hypothesis
   - one or several
   - each with a confidence factor
   - total must not exceed 100%
3. Look at results
   - Look at each hypothesis
   - Look at summary information
4. Discuss ....
**D. Mister QCM**

- **Goal:** Students have to experience QCM design effects

- **Scenario:**
  - Students fill in 2 badly designed MCQs
    They also indicate their level of competence.
  - Students look at results
    (question by question and overall plot)
  - They are very angry at the results
    (bad performance on what they know and good one on what they don’t know)
  - Now they are motivated to understand QCM design and participate in classroom discussion

- The tool has design notes on each question (including “don’ts”)

---

[Graph showing the relationship between prediction and real score, with arrows indicating good and bad performance in domains they know and don’t know.]
3.4 Information Space

Motivation:

• Better support for less structured scenarios ...
• Resource-based teaching implies support with documentation

Details:

• An RDF-type of database
  (http://tecfa.unige.ch/guides/rdf/pointers.html)
• Various interfaces (queries, graphical networks, etc.)

..... No details (sorry)
3.5 XML Grammars for project support

- Grammars are central to understanding and doing

\[
\text{XML} = \text{grammar} + \text{content}
\]

- XML will add structure to web contents (and more ...)
  ([http://tecfa.unige.ch/guides/xml/pointers.html](http://tecfa.unige.ch/guides/xml/pointers.html))

Simple use of XML

- Write a grammar (schema, DTD)
- Students use it to write something
- Give feedback, organize discussions for different phases
- Server-side programs can extract and combine information

Tools for simple XML authoring (1999)

- exist, but are either expensive or not totally usable
Example: Project Management

1. Students do a feasibility study
2. Feedback:
   Teacher fills in slots with comments
3. Students write the specification
   Teacher adds comments
4. ......
5. ......
6. ......

• Technical note for display:
  best current solution is XSLT
  (translate to HTML)
Summary:

A lot of structured activities:

1. Students do something
   • using tools and resources
2. Results are collected
3. Debriefing, synthesis, feedback

mostly work flow tools

• Tasks are often pseudo tasks,
• phases are sometimes repetitive,
• most are collaborative or “collective”,
• collaboration is structured,
• teacher can monitor, add, help, edit ...
A word of caution

• Some student projects/exercises will remain “open”
  • We have to avoid the “video game effect”
  • We must plan the mastering of the content, not of the tool

Traditional project based teaching

Don’t stop here!

With:
  scaffolding
  guidance
  learning tools
We have more ...

- some ideas are spreading (medical school, social sciences)

It’s research for us!

- We do data collection on some tools
- A lot of people should mine data!!
- Campus systems: only Virtual-U allows data-mining
  http://virtual-u.cs.sfu.ca/vuweb/VUenglish/

• Note:
  • We never will build any authoring environment
  • You can steal ideas from us:
    http://tecfa.unige.ch/campus/tecfarama/campustour.php
4. Outlook on technology

4.1 Integrated campus software

- ok for traditional, resource or project-based teaching
- not much support for learner activities (tasks)
- limits on “on-line” creativity
- student management tools above task level are ok
- progress in the next few years, but not a lot I believe (unless we get easily to understand open source)

.... of course, you can combine (and a lot of people do)

💡 Pick a campus tool that allows adding your own tools
4.2 The open environment

- Standard communication tools (possibly better integrated)
- SQL server and maybe an object store
- Some Java applets and servlets
- Custom plugins (e.g. Authorware)
- LDAP (for central authentication and person look up)
- XML and less HTML (at Tecfa: near future)
- Server-side html-embedded scripting languages (Php, Asp, ColdFusion, Java-HTML)
- Advanced Groupware (like CVW, Workplace)
- Multi-user environment servers (DeepMatrix, Moo, etc.)

...... but you have to do it yourself :(

...... pay a programming course to one of your people
How much does our development cost?

Technology is mostly free

- Simple HTML, soon: XML
- MySQL (a free relational SQL database)
- PHP
  - server-side html-embedded scripting language
  - supports SQL, XML, graphics, LDAP, etc.
- A bit: Javascript, Java (applets), Authorware and plug-in, Python, Perl, VRML, Java Web Server (servlets), MOO

Resources:

- Teaching assistants (know basic programming)
- Cost for one activity: 2 weeks - 1 month
- Little for a real production team, a lot for a teaching team!
5. Conclusion

The old CBL debate:

- is still important
- there are (fine) CBL applications on the Web
- E.g.
  - simulations
  - tutoring systems
  - instructionalist text
  - ........

- Various good and proven solutions exist
The new Web debate

E.g. open learning adds:
  • big information space
  • collaboration

needs:
  • some monitoring
  • some scaffolding
  • some structure

• Main stream virtual campus software is not flexible enough .... but what can we use instead?
I am a bit pessimistic
... we all can/will do better
... but it will cost!

Accept some instructionalism
but don’t kill Piaget and Vygotsky

Try to give learners better support
... You can! (even with simple tools)