

A Framework for the Development of Educational Presentation Systems and its Application

Georg Turban
Darmstadt University of Technology
Hochschulstraße 10
D-64289 Darmstadt, Germany
turban@informatik.tu-darmstadt.de

Max Mühlhäuser
Darmstadt University of Technology
Hochschulstraße 10
D-64289 Darmstadt, Germany
max@informatik.tu-darmstadt.de

ABSTRACT

In this paper, a framework for the development of educational presentation systems is presented that eases and decouples the development of optional components that contribute to a very basic presentation system. Optional components are introduced in order to support audience response systems, lecture recordings or content sharing using a learning management system, for example. After the presentation of our design goals, we present selected aspects of our framework and the application to our presentation system.

Categories and Subject Descriptors

K.3.1 [Computer and Education]: Computer Uses in Education – *Computer-assisted instruction.*

General Terms

Documentation, Design, Human Factors, Standardization.

Keywords

Face-to-face learning, framework, educational presentation system, multimedia based presentations.

1. INTRODUCTION

In the context of electronically assisted face-to-face learning, various research has shown the benefits of individual and different systems. For example, the benefits of interactive devices [1], [2], [3] or systems that assist interactive face-to-face sessions have been presented in [4], [5] and [6]. Also, the contribution of lecture recording has been focused within the last few years and evaluation results have been presented in [7], [8] and [9], for instance.

The trend that can be observed from current developments is that systems increasingly combine the functionality of different areas: presentation systems have been enhanced to support interactive sessions and typical lecture recording systems such as [10] started to add support for the creation of interactive environments.

Our observation and the depicted trend motivated our work to provide a framework for the development of educational

presentation systems that may provide very basic up to multifarious support for face-to-face learning. It should be emphasized that our intention is not to develop systems that comprise a maximum of functionality. The benefits of an extensible core implementation are others: The administration of a single system and development of new components is more economic while using an extensible solution rather than isolated systems. In respect to handling of content, processes can be simplified. In addition, we expect higher loads for lecturers who use different, independent systems, especially while using them in parallel.

2. DESIGN GOALS

Our main goal is to provide an environment for face-to-face learning that supports different teaching styles by composition of various subsystems for presentation, interactive sessions, recording and so on. In correspondence to our main goal, we address the following subgoals, briefly sketched here.

Extensibility. The framework should follow a component-based approach and ease the replacement of existing and introduction of new functionality. As a basic design decision, the framework should not rely on platform-specific issues.

Usability. A system that supports face-to-face sessions should obviously be easy to use. While using separate applications may easily result in juggling with them, a framework that provides the functionality of different applications by composing subsystems should cover concepts and mechanisms that ease their handling.

Maintenance. While different applications contribute to the presentation of content, the corresponding content should also be stored for later usage. A framework should deliver mechanisms so that the presentation for the audience is not disturbed and the recordings entirely reflect the presented content.

Ink-awareness. Minimum chalkboard functionality should be provided by the framework, making it ink-aware. While a framework supports ink, other subsystems may also benefit from its annotation facilities.

Environment-awareness. In our terms, a framework should assemble different components based on the requested, individual target scenarios and therefore be able to distinguish between different users and configurations. Especially while setting up the target scenario and maintaining the content, information about the environment is essential and eases the interaction with content management systems, for example.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

EMME'07, September 28, 2007, Augsburg, Bavaria, Germany.
Copyright 2007 ACM 978-1-59593-783-4/07/0009...\$5.00.

3. CONCEPT

The framework currently distinguishes between *core components*, *replacements* and *extensions*. Core components provide basic functionality which is required either from a technical or functional point of view. For instance, technical components realize the loading/unloading of other components, while functional components provide basic whiteboard and presentation functionality. Replacements are distinguished because some components (either core or other components) may be instantiated only once. For example, the core session handling may be replaced by another component.

We rely on core components that are grouped to a presentation subsystem in order to provide very basic presentation and annotation functionality (please refer to figure 1 which represents the selection of core components that provide the basic presentation functionality). The core also consists of other components (some will be described in following sections) for session- and content-handling, for instance. While moving from the basic presentation scenario to more enhanced scenarios like those supporting interactive sessions, we distinguish between the basic presentation and other methods by introducing *contexts*. Contexts reflect different teaching styles respectively phases and may occur during a single lecture. For instance, a lecture often starts with the presentation of slides and is then interposed by questions from the audience or prepared quizzes from the lecturer.

We presume that the lecturer works in contexts while they occur in non-overlapping phases and therefore distinguish on the framework level between *active* and *inactive contexts*. The impact of contexts on the user-interface is expected to be high. While working in the presentation context, visual elements of other contexts should not clutter the user-interface and provide their elements for the user-interface on separate panes. In order to provide the lecturer with a possibility to monitor inactive contexts, a small, dynamic visual representation may be contributed from contexts.

While working in a context that differs from the basic presentation context, components may also produce slides on the fly. By transforming a student's question into an image, it may be forwarded to the presentation subsystem and annotated as well. The

framework therefore relies on a component that is able to annotate images painted on top of colored surfaces. These three basic types of layers have been presented in [11] and can be represented by so-called slide models. Images may be updated frequently and may also cover regional updates in order to annotate video streams that may be delivered from a screen grabber. For instance, [12] describes the usage of the virtual network computing protocol for transmission of remote screen updates in educational scenarios. Therefore, the basic presentation part of the framework covers the following core components.

Surface. The *Surface*-component delivers a simple color so that the presentation system is capable of representing different surfaces, like those of white- and blackboards.

ImageSelector. The *ImageSelector*-component is able to select images provided from different ImageProviders and operates according to events. Events can be generated by the user or other components.

InkCanvas. *InkCanvas* is basically a controller component that tracks the mouse and can be attached to any area of the screen. It determines mouse events and collects points, which are grouped into strokes, distinguishes between different strokes that form a so called stroke-collection that is assigned to a single slide and finally delivers ink-objects for further processing.

SlideRenderer. The *SlideRenderer*-component continuously renders the ink provided by *InkCanvas* over the image (which may contain transparent regions) from the *ImageSelector*-component and the surface obtained from the *Surface*-component.

As depicted in figure 1, images for the *ImageSelector*-component are typically delivered from files, slide-based presentations or screen grabbers, but may also be obtained using any other component (quizzes, call-ins, etc.) that implements the ImageProvider-interface. The *ImageSelector*-component can even be omitted, resulting in a simple whiteboard, provided by the remaining two components *Surface* and *InkCanvas*.

The selection of one image source from all connected choices is based on events that can be fired manually by the user, automatically detected during context switches or forced by contexts that are able to deliver slides on the fly. A *quiz*-component for example may

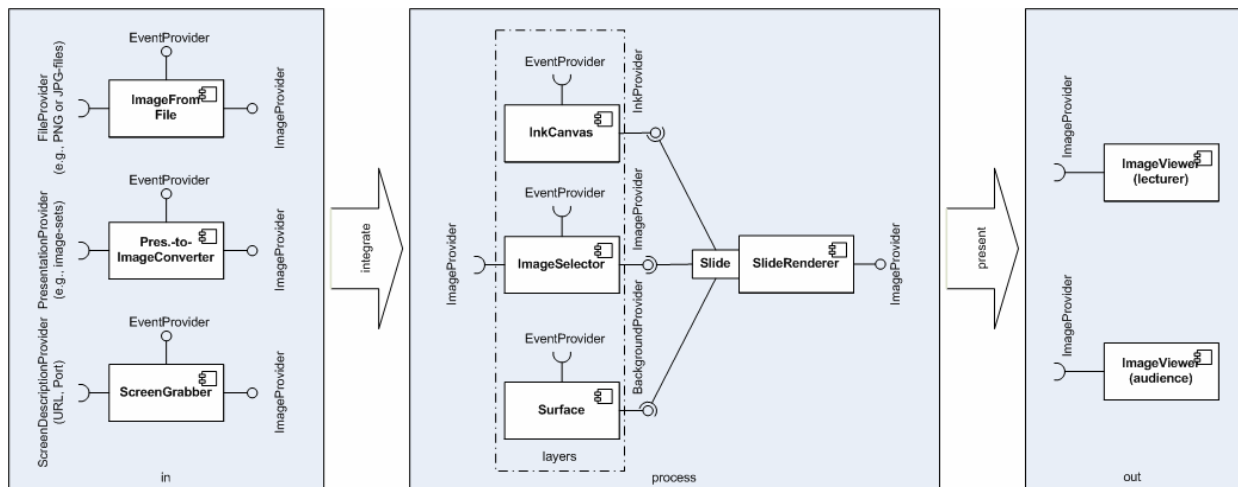
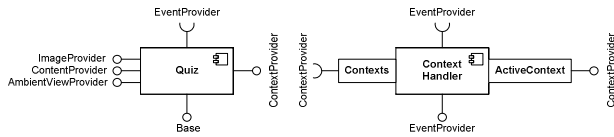


Figure 1: Components for core presentation support.

Introducing the *ImageSelector* allows switching between different image sources and covers static and dynamic content.



```

<?xml version="1.0" ?>
<quiz>
  <single-choice>
    <question>The best way to recognize whether [...]</question>
    <answer>plotting the variable against time and [...]</answer>
    <answer>calculating the least squares regressi [...]</answer>
    <correct><answer>plotting the logari [...]</answer></correct>
    [...]
  </single-choice>
</quiz>

```

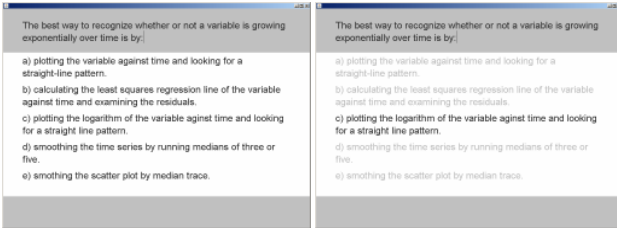


Figure 2: An example for a quiz-component.

automatically generate a slide that contains a diagram of the analysis of submissions from the audience and immediately request presentation focus. Examples using contexts and ambient displays are discussed from a different viewpoint (in correspondence to their benefits for developer, auditorium, lecturer and supporters) in [13].

Contexts can also contribute their underlying content in order to ease the creation of explorative lecture recordings. While slides are obtained from images, they are obviously useful for lecture playback, but lack more enhanced post usage scenarios. For instance, the quiz definition file that has been sketched in figure 2 may not only be used for presentation purpose. While its image representation is processed by the *SlideRenderer*-component and the corresponding views for the lecturer and audience are updated, it contributes to a component capable of storing any content apart from presented and annotated slide-images.

ContentHandler. This component is able to save online as well as offline content including temporal information. Online content may directly result from images or texts generated and still present in memory, while offline content like PowerPoint-files may be copied from arbitrary locations into the content pool.

Configurator. Finally, a component provides the possibility to add or remove components such as the *quiz*-component and to configure them using a graphical user-interface. Disabled components do not waste resources and their artifacts can also be deleted.

4. APPLICATION

Presenter, the presentation component of the digital lecture hall project [14], has been implemented according to our framework and its underlying design goals. It is currently used at several universities and supports multifarious scenarios by different configurations. In some lectures, it is used just for presenting simple slides while no other components have been requested by the lecturers. In a single lecture, it is even used just as a chalkboard. Other lectures use the recording, interaction or other components in addition to the basic functionality.

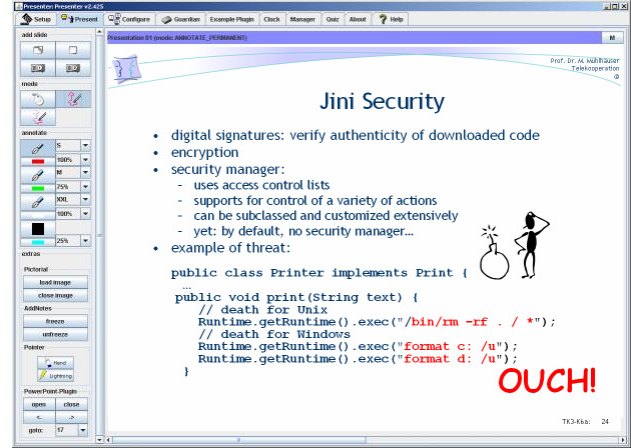
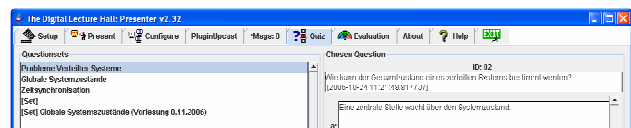


Figure 3: Lecturer's user-interface during a slide-based presentation while showing the presentation context.

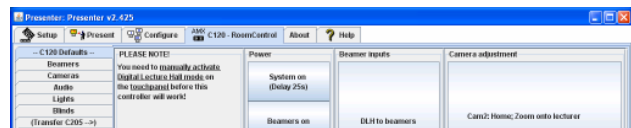
The user-interface for the lecturer is presented in figure 3, showing an activated presentation context. It has been maximized and in contrast to a full-screen PowerPoint-presentation, the remaining visible area for a presentation slide is not significantly reduced. The user-interface has been created on a screen with a resolution of 1280x1024 pixels, with the remaining slide area still covering around 80% of the screen. On the left side of the presentation context, typical controls for ink aware applications are provided in order to change pen attributes, for example. Elements in the latter region vary since components can also contribute to the presentation context. The tabs on the upper part of the user-interface represent the different contexts and provide the functionality of small ambient displays. The title as well as the icons can be dynamically changed by the corresponding contexts.

Referring to figure 4, different sets of components have been assembled in order to support different scenarios by Presenter. Set-up a) enables interactive sessions, e.g., the speedometer in the

Set-up a) contexts (call-in, quiz, evaluation) for interactive sessions:



Set-up b) controlling the hardware of the lecture hall from a context:



Set-up c) sample SDK-contexts for development and demonstration:

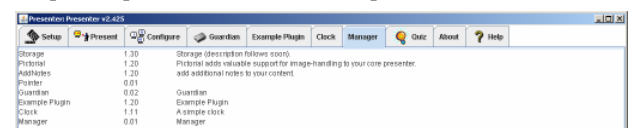


Figure 4: Lecturer's user-interfaces for different set-ups.

seventh tab represents the assessments of students regarding the lecture speed. Submissions from the audience are accepted during the whole lecture and continuously evaluated.

For our three lecture rooms, we developed a component that has been integrated into Presenter and automatically controls the infrastructure of the corresponding lecture hall. According to the current configuration within the session handling in set-up b), projectors are turned on and cameras are aligned to the presentation desk, for instance. The framework triggers this control so that shutting down the presentation system also results in shutting down the corresponding hardware devices. The component is also a context and assists the lecturer in other ways; the video streams of the cameras (we use up to four cameras) can be manually assigned to the projectors for monitoring purpose by a single click. Introducing latter component eased the usage for lecturers especially in environments of multiple projectors and cameras.

Set-up c) consists of different components and supports development and demonstration. An *example*-component implements and uses most of the available interfaces and can be used as a starting point for the development of new components. Other components, like the one activated in the snapshot, display meta-information and visualize the composition of components that contribute to the current scenario.

The sample *quiz*-component (please refer again to figure 2) heavily uses the framework and has been developed rapidly. It consists of less than 350 LOC and parses a simple definition file for quizzes (in XML-format), creates a quiz-object and renders two images. The image to the left is presented to the audience at the beginning, while the second image highlights only the correct answer and is presented after the session. By implementing the ImageProvider-interface, latter images can be delivered to other components and annotated within the presentation subsystem. In addition, the definition file is transferred to the content pool by implementing the ContentProvider-interface. The definition file can be used in post usage scenarios, so that the quiz can be replayed in combination with the *quiz*-component.

5. SUMMARY

We introduced our framework for the development of educational presentations systems and its application to Presenter, the presentation component of the digital lecture hall project. The presentation system has been developed using the framework and follows the presented design goals. We highlighted our idea of components that contribute contexts to the core presentations system and we used the sample component for quizzes in order to illustrate the design of contributing components and their handling using the framework. The composition of components has been discussed and some examples for different Presenter set-ups that range from very basic presentation support to more complex ones have been presented.

6. OUTLOOK

Current implementations use our own plugin-framework to manage components and their underlying resources. We are evaluating different implementations of the OSGi-framework [15] at the moment and may substitute the corresponding parts. In addition, Presenter is currently connected to a component that replaces the core session handling and experimentally synchronizes content with a learning management system.

7. REFERENCES

- [1] Mock, K.: Teaching with Tablet PC's. In: Journal of Computing Sciences in Colleges. Consortium for Computing Sciences in Colleges, USA, 2004.
- [2] Hulls, Carol C. W.: Using a Tablet PC for Classroom Instruction. In: Proceedings of the 35th ASEE/IEEE Frontiers in Education Conference, Indianapolis, IN, USA, 2005.
- [3] Hoppe, H. U.; et al.: Interactive Presentation Support for an Electronic Lecture Hall - a practice report -. In: Advanced Research in Computers and Communications in Education, IOS Press, 1999.
- [4] Scheele, N.; et al.: Experiences with Interactive Lectures: Considerations from the Perspective of Educational Psychology and Computer Science. In: Proceedings of the International Conference on Computer Supported Collaborative Learning, Taipei, Taiwan, 2005.
- [5] Latessa, R.; Mouw, D.: Use of an audience response system to augment interactive learning. In: Journal of Family Medicine - Innovations in Family Medicine Education, 2005.
- [6] Draper, S. W.; Brown, M. I.: Increasing interactivity in lectures using an electronic voting system. In: Journal of Computer Assisted Learning 20, Blackwell Publishing, 2004.
- [7] Lauer, T. Müller, R. Trahasch, S.: Learning with lecture recordings: key issues for end-users. In: Proceedings of the IEEE International Conference on Advanced Learning Technologies, Joensuu, Finland, 2004.
- [8] Lane, C.: UW Podcasting: Evaluation of Year One. University of Washington, October 2006 Technical Report (http://catalyst.washington.edu/research_development/papers/2006/podcasting_year1.pdf), 2006.
- [9] Randolph, C.; Thompson, E.: Increasing the value of recorded lectures: An offshore experience. In: Proceedings of the 16th Open and Distance Learning Association of Australia, Canberra, Australia, 2003.
- [10] IMC AG: Website: Lecturnity 2. Website: <http://www.lecturnity.de/>, last visited July 1st, 2007.
- [11] Turban, G.; Mühlhäuser, M.: A Category Based Concept for Rapid Development of Ink-Aware Systems for Computer-Assisted Education. In: Proceedings of the 7th International Symposium on Multimedia, Irvine, CA, USA, 2005.
- [12] Ziewer, P.; Seidl, H.: Transparent TeleTeaching. In: Proceedings of the Australasian Society for Computers in Learning in Tertiary Education conference, Auckland, New Zealand, 2002.
- [13] Turban, G.; Mühlhäuser, M.: An Open Architecture for Face-to-Face Learning and Its Benefits. In: Proceedings of the 8th IEEE International Symposium on Multimedia, San Diego, CA, USA, 2006.
- [14] Mühlhäuser, M.: Digitale Hörsäle: Wo Präsenz- und Cyber-Universität sich treffen. In: Studieren im Cyberspace?, Reihe: Bildung und Technik. LIT Verlag Münster, Germany, 2005.
- [15] OSGi Alliance: About the OSGi Service Platform. Technical Whitepaper (<http://www.osgi.org/documents/collateral/OSGiTechnicalWhitePaper.pdf>), last visited July 1st, 2007.