# An adaptive and adaptable learning platform with real-time eye-tracking support: lessons learned

Holger Schmidt[1], Benjamin Wassermann[1], Gottfried Zimmermann

Responsive Media Experience Research Group
Media University Stuttgart, Nobelstraße 10, 70569 Stuttgart-Vaihingen
schmidtho@hdm-stuttgart.de,
benjamin.wassermann@uni-tuebingen.de,
gzimmermann@acm.org

**Abstract:** For a learning platform to be adaptive, it needs to monitor the user in real-time. At the ScienceCampus Tübingen, we conduct empirical research on learning strategies, in particular with regard to system adaptivity. Our experimental Web-based adaptive multimedia learning environment is able to communicate with an eye-tracking system in real-time, and links gaze data to browser-based learning content. By processing simple evaluation scripts in the browser, we can control the learning process by adapting the learning content in response to predefined conditions and user inputs. The experimental system serves as a basis for a series of psychological studies in the learning context.

We have already conducted some studies with our experimental learning platform, but its development is still ongoing. In particular, we are adding new features required for future studies. In this paper, we describe the actual state of our experimental learning environment, lessons learned from its application in the studies, and our plans for future development.

## 1 Introduction

The process of learning is a complex interaction between a learner and learning content. Traditional and emergent learning methods are facilitated by new technologies, under the influence of digital media [Az11]. There are individual needs for every single type of learner. A fundamental goal of the empirical research for education is to understand how learners process learning content and how regulation can aid the learning process and increase the learning success [Am13].

The project cluster Adaptable and Adaptive Multimedia Systems (AAMS) is one of many projects under the administration of the ScienceCampus Tübingen[2], initiated and supported by the Leibniz-Foundation[3]. The main goal of the ScienceCampus is to build a network of empirical research for education. AAMS is one of eight project clusters with a total of 27 projects. It engages in the interdisciplinary research of self-regulated

[1] Authors contributed equally to this publication
[2] http://www.wissenschaftscampus-tuebingen.de
[3] www.leibniz-gemeinschaft.de

learning from multimedia, and involves the University of Education (PH)[4] in Freiburg, the Institute of Psychology[5] of the University of Freiburg, the Knowledge Media Research Center (KMRC)[6] in Tübingen, and the Media University Stuttgart[7]. The studies conducted by our cluster require proper test environments for generating and simulating controlled learning situations. Therefore, the Media University Stuttgart has developed the Adaptive Learning Module (ALM) as an extension of the open-source learning platform ILIAS[8]. ALM makes eye-tracking data available for Web applications running in a browser in real-time. ALM also offers an authoring environment for the preparation of learning content presentation and a viewer mode for full-screen presentation of the adaptive learning content.

The remainder of this paper is structured as follows. In section 2, we give a brief overview of related work. In section 3, we describe our experimental learning environment from two perspectives: that of an author and that of a student. Section 4 takes a rather technical view on the experimental system and its components. Section 5 describes the application scenarios and their use in psychological studies. In section 6, we summarize the problems we have experienced so far, and how we plan to solve them in the near future. Section 7 provides a short outlook on the further development of the system.

## 2 Related work

In this brief overview of related work, we focus on a few selected examples of specialized systems and tools that have high relevance for our adaptive learning environment.

The Smart Sparrow software [BHB14] is a Web-based adaptive e-Learning platform developed by the adaptive e-Learning research group at the University of New South Wales. It analyses how students learn by evaluating their responses to the learning content. Because this software does not use eye-tracking, an important source of real- time information is missing.

Text 2.0 [Bi10] uses a text-based approach of adaptive real-time interaction with eye-tracking. It was developed by the Institute for Research of Artificial Intelligence in Kaiserslautern. With its eye-tracking interface, Text 2.0 is able to follow the user's gaze on the text. When reaching a specific text position, the system shows illustrations or plays audio automatically. The framework is based on Java and JavaScript. The low-level events of the framework (e.g., gaze-in and gaze-out) can be applied to all HTML content elements, but higher-level events are designed for text purposes only.

[4] www.ph-freiburg.de
[5] www.psychologie.uni-freiburg.de
[6] www.iwm-kmrc.de
[7] www.hdm-stuttgart.de
[8] www.ilias.de

MetaTutor [Az09] is a learning environment that provides tools for self-regulation. Depending on the chosen learning method and path, MetaTutor adapts the learning process and content to fit the learner's preferences.

AdeLE [AG10] has developed a real-time eye-tracking application for support of research for education. It uses eye-tracking as an information gathering tool in learning situations to facilitate new techniques of adaptive interaction between learning content and learner. The resulting application is known as the AdeLE eye-tracking system [Gü04]. This application bears high resemblance to our adaptive e-Learning environment.

Although MetaTutor and AdeLE support all major media types and provide many useful features, neither system provides the flexibility required for our project. For example, we plan to conduct experiments on mobile platforms, using a responsive design approach on the Web.

## 3 Adaptive e-Learning environment

The actual version of our adaptive e-Learning environment is based on the open source e-Learning platform ILIAS and was developed as extension thereof. This extension is the foundation for a series of already conducted and upcoming empirical studies within the AAMS cluster. The extension can be divided into two main parts: (1) the authoring environment for the creation and editing of the content for the learning units, tests and tasks, and (2) the lecture viewer to visualize the learning content.

### 3.1 Authoring environment

The authoring environment (see figure 1, part A) is built seamlessly into the ILIAS user interface (see figure 1, part B). It uses the same concepts for creating and manipulating content as ILIAS for its own content objects. This reduces the barriers for learning content authors that are familiar with the ILIAS platform. ILIAS provides as set of objects of different types that can be created within a repository. The repository is organized hierarchically, i.e., some objects can contain child objects. Examples for these container objects are categories, courses, groups and folders. Examples of other (non- container) objects are files, fora and wikis. Access rights are inherited through the hierarchy. In addition, access rights can be configured individually for every object in the hierarchy.

The ILIAS course object is used as the base container of learning content. It provides a rich set of user access management tools and serves as the basis for an "Adaptive Learning Module" (ALM) lecture (see figure 1, part C). A lecture is composed of multiple content pages called learning units. A learning unit is represented by the ALM object extension within ILIAS. ALM objects can be created inside course objects in any number (see figure 1, part D). Each ALM object can hold learning units of various media types, including text, image, audio, video, and interactive animation (flash). Depending

on the media type object, the author can select required resources, add metadata, and configure the adaptive behavior of the media type. For example, an image requires a title and an image resource, and may accept a subtitle or metadata for keywords, author, etc. Additionally, a custom configuration for adaptive behavior can be provided.

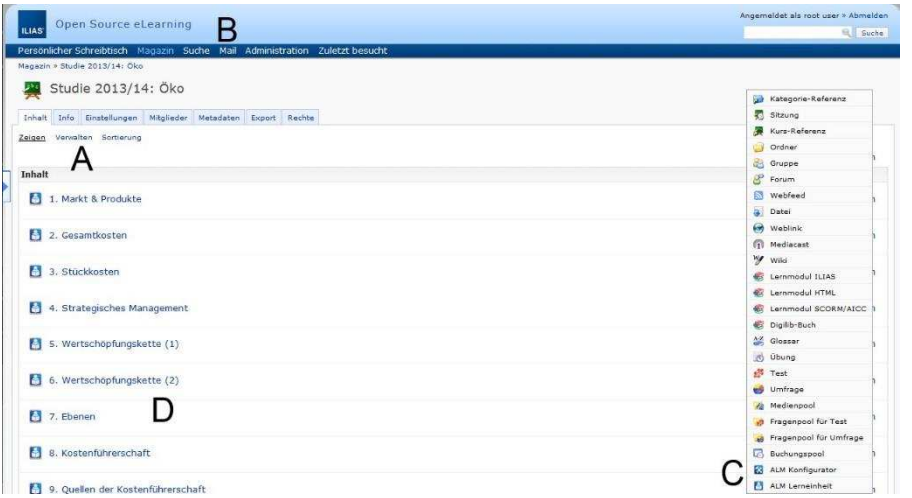


Figure 1 - Authoring ALM units

A learning unit contains any number of content objects and multiple sets of tasks to realize pre- and post-tests. It provides a set of properties for customization (e.g., choosing templates or selecting default content) and a JavaScript file to manage custom behavior of the content. For administration of learning content, we use the ILIAS user interface and its course structure. The lecture presentation can be either embedded in the ILIAS interface or presented in a separate full screen mode, thus hiding the ILIAS context. The ALM lecture viewer shows the learning content as a combination of the presentation of static learning content, pre-generated tasks and adaptive components which provide the dynamic functionality of the system.

### 3.2 Lecture viewer

The ALM lecture viewer is based on HTML5 [Be14] and presents learning content in a rather classical Web-page layout (see figure 2). On the left side, a navigation area lists the available learning units. The main area on the right side shows the learning content but can also have a top bar for further functions and a bottom bar for linear navigation. The main layout can be adapted by templates. Thus the author can add, replace, and remove control or content areas and change the style. The template can be configured for every unit, to adapt the controls within a lecture. The navigation area (see figure 2, part A) indicates the current position within the lecture and allows the user to navigate to any unit. The navigation area can be disabled by configuration. In this case, the user can only navigate via the bottom bar or via a custom navigation bar. The bottom bar (see figure 2, part B) is for linear navigation only, allowing the user to move back and forth between

units. The author can disable the "previous" button or control the visibility of the "next" button by adaptive methods. The area between the buttons can be used to display messages.

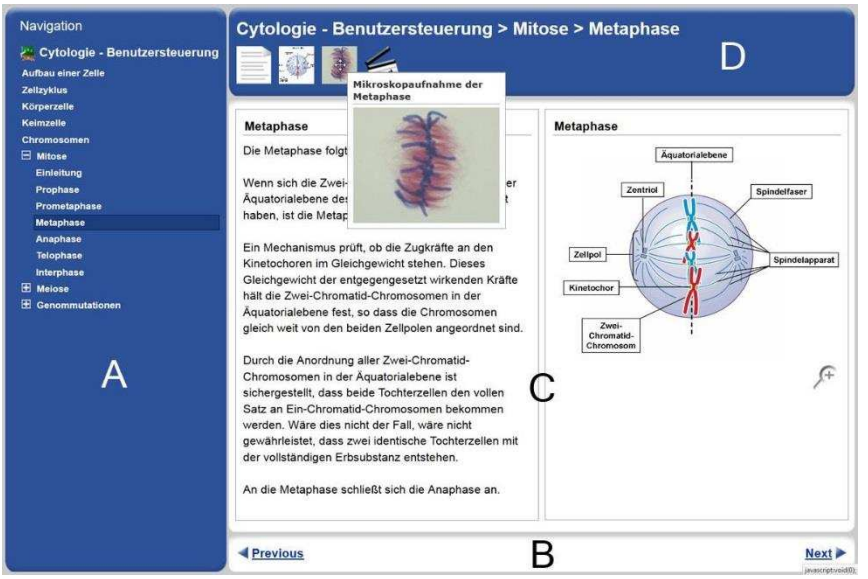


Figure 2 - Example for a 2-slot learning environment with Mediashelf

The main content area can use various layouts. By default, a side-by-side 2-slot content layout is used (see figure 2, part C), but other layouts like a single-content slot or a 4-slot layout are available. Special layouts can be added via custom templates. Like the main layout of the lecture viewer, the content layout can be set individually for every unit. A content area or slot is also a pre-defined indicator for gaze events and can be used by the adaptive methods of the system. The author defines the default content for each slot, i.e. the content that is initially shown to the user. If no default content is defined, the slot area will be empty and can be filled by the user. By default, the top bar displays the actual unit name and gives a selection of unit items (see figure 2, part D). We call this bar Mediashelf, because it provides a choice of content elements of different media types for the actual unit. For each content element in the Mediashelf, a preview is dynamically displayed when the mouse hovers over it. The user can place any Mediashelf element in any slot of the main content area via drag and drop (user-initiated adaption).

## 4 Technical details

This section describes the architecture of the adaptive e-Learning environment which is based on the open source framework ILIAS 4 (see figure 3). ILIAS 4 provides a plugin instrument for developers to extend ILIAS with their own components. Multiple plugin interfaces are available to create components for different application areas. For example, developers can modify standard components through the User Interface Plugin

API, and they can create their own content objects within the ILIAS repository through the Repository Object Plugin API. Through the APIs, ILIAS provides access to core components and services of the framework. Since the ILIAS framework is based on the PHP programming language, the plugin interfaces are provided in PHP.

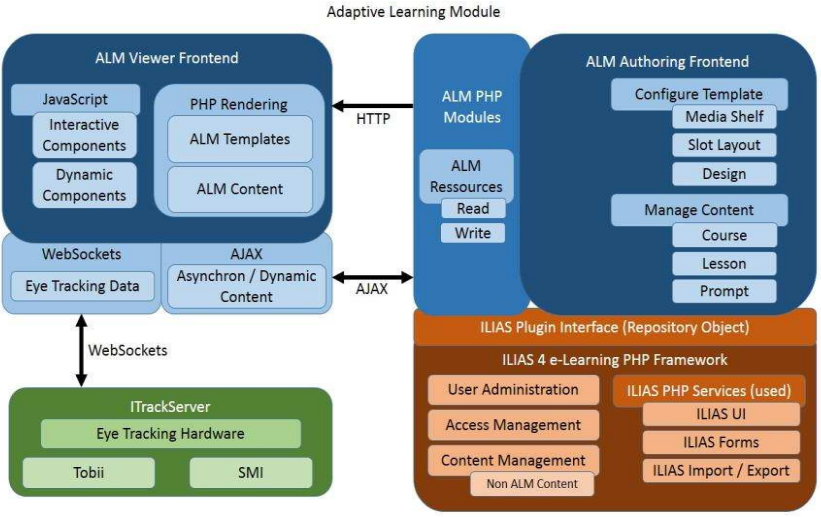


Figure 3: Adaptive Learning Module (ALM) architecture

Our Adaptive Learning Module (ALM) is implemented as an ILIAS Repository Object Plugin component. Since the Repository Object Plugin represents an ILIAS object within the repository tree, the ALM objects inherit all functions of repository objects. The most important functions are the instantiation tools for creating, modifying and deleting objects within the repository, and the user and access right management tools. For the rendering of the extension graphical user interfaces (GUIs), ILIAS provides a set of services. The ALM authoring front-end uses these services to render the GUI.

The content of the ALM objects is stored in a separate file structure. Each object is defined by an XML [Br14] descriptor and its related resources. The ALM extension provides tools for handling low-level operations for the resource management independent from ILIAS. However, the user interfaces for content management are provided by ILIAS.

Within the ILIAS GUI, the ALM authoring front-end is displayed by default. The authoring front-end provides a preview section with the ALM viewer embedded. Thus, the user can start the viewer without the ILIAS context. The preview is available for all users that have access to the associated course and have viewing rights.

Server-side PHP templates are responsible for rendering the main part of the ALM viewer front-end. Initially, the client loads a page with all content objects. For learning units that have the Mediashelf function enabled, user-initiated content changes for the slots are handled by Ajax requests. Upon such a request, the server generates the new content and

sends the data to the client that dynamically exchanges the displayed content. Additional remote commands are available for logging, to manage user data or to evaluate tasks. In addition, the ALM viewer includes client components for interactive and dynamic rendering of custom data, for adaptive methods, and for tracking and evaluating browser events.

If an eye-tracking system is connected, the regular browser events are supplemented by user-triggered gaze events. For our setup we use the RED eye-tracking system of SMI [Te14]. Since the SMI communication interface is not able to communicate directly with the Web-browser, we developed a stand-alone tool called iTrackServer as a gateway between RED and the Web-browser. The iTrackServer links to the RED via a UDP network connection. The Web-browser communicates with the iTrackServer via the HTML5 Websocket API [Hi14a]. For this link, we use a direct network connection due to the high performance requirements of real-time gaze data transmission. We developed iTrack, a JavaScript library for Web-authors for the reception and interpretation of gaze data from the iTrackServer. More details on this topic are available in a previous publication [WHZ12].

## 5 Application scenarios

This section provides an overview of the features of our e-Learning environment. We hereby refer to the studies of our cluster for the purpose of illustrating the capabilities of our e-Learning system. For the results of these studies please refer to the referenced literature.

One of the first studies in our cluster was conducted by the team at University of Education in Freiburg. They investigated self-regulated learning by using the interactive Mediashelf component of the learning environment [RP12]. The Mediashelf presents all available content elements of the actual learning unit. The learner is free to set up the content area of the learning unit by dragging content elements from the Mediashelf to any of the slots in the content area. The learner can reset or exchange slot content at any time.

After evaluating the Mediashelf, an interactive helpdesk was developed and evaluated in a subsequent experiment [Ru14]. The author specifies the location of the helpdesk, and defines the information to go with it. Alternatively, the author can define free text areas in which the learner can type their own content. The visibility of the helpdesk can be configured in multiple ways. First, the area can be displayed permanently on a specific side of the learning unit. Second, it can be indicated as closed bar which can be flipped by the learner. Third, it can be flipped from closed to displayed in response to predefined events, e.g., controlled by timer, fixation/transition of particular elements, or answers to test items. The event data provided by the platform is a combination of user input, such as mouse position or mouse clicks on the learning content, and real-time gaze events like gaze-in, gaze-out and fixations provided by the iTrack component. The delivered gaze data contain information about the time, duration, and position of the gaze. For more information about the iTrack component see [WHZ12].

ALM's capabilities for adaptation of content, navigation and presentation are going to be used in a study at the KMRC[9] Tübingen. The author can define custom scripts[10] for learning-units to specify a particular behavior for a particular page. When the user presses the "Continue" button, the system evaluates the gaze data of the current page with regard to minimal requirements on fixations and gaze transitions between predefined areas of interest or content slots. If the requirements for continuing are not met, the system blocks the continue process and shows an adapted presentation of the last content to the learner. For example, if a specific figure has not received sufficient attention by the learner, its presentation is complemented by pop-up prompts. This method directs the focus to the image by placing it in the center of the screen while the rest of the screen is shaded in grey. The popup is locked for a certain time before the subject is able to continue with the lecture. As another example, the system can hide or deactivate the "Continue" button until the learner has reached a minimum number of gaze transitions between an image and its describing text.

Our system's capabilities with regard to adaptations in response to rapid assessment tests are illustrated by an ongoing study at the University of Freiburg. This study makes use of a combination of adaptive methods and predefined tasks. The learning environment supports the following types of tasks: Free-text tasks in which the learner can add text to answer a question, and single-/multi-choice tasks. Upon pressing the "Continue" button, the system validates the learning success of the learner based on the answers given. If the subject fails to answer the tasks correctly, the system will react with predefined adaptive modifications of the content depending on the evaluation of the gaze data.

## 6 Lessons learned

The ongoing process of implementation and improvement of the system is guided by our experiences and those of our cluster partners in the conduct of their studies with ALM, and by requirements for upcoming studies. This sections gives an overview of the major problems we have encountered so far, along with our proposed solutions we are about to implement in the near future.

### 6.1 Dependency on ILIAS

Using ILIAS as basis for the ALM learning environment saved us a lot of development effort on user and access management, and also provided extensive tools for creating user interfaces within the framework for free. In most of the studies, using ALM with ILIAS was beneficial in terms of using the rich functionality of the platform. But in some cases, the preparation and conduct of the experiments would have been easier to conduct and more flexible by using a more light-weight framework than ILIAS. Therefore, we decided to decouple ALM from ILIAS. This will also allow us to run future studies, using ALM within other frameworks than ILIAS. To complete the framework, we will

[9] www.iwm-kmrc.de
[10] Script language: JavaScript

develop a simple course management system, including user and access management tools.

The new version of our adaptive e-Learning environment will be composed of two main parts: The authoring environment (independent of ILIAS) and the lecture viewer (an extended and decoupled version of the current one). Both will be implemented as HTML5 Web applications. The lecture viewer will run as standalone Web-application, using a single course package as content resource. Since we have good experience using the ILIAS framework as basis, we also plan to revise the existing extension to use the new authoring and viewer application within ILIAS. In addition, we consider integrating our tools within other e-Learning platforms such as Moodle[11].

### 6.2 Modularity

We designed the adaptive learning module to represent a learning unit that can contain an arbitrary number of content elements. Later we had to extend the system by tests containing tasks. Since the system wasn't designed completely modularly, the subsequent extensions were implemented as "dirty" add-ons within the learning unit component rather than as additional modules with a clean separation from the extended system. Because of this, we decided to revise the system to be more modular, based on items described by modules. A module defines the item type, the dependencies between items, the item properties and the content fields. The revised system will also provide the configuration tools for the authoring and the rendering methods for the viewer.

Using this approach, we can easily add additional modules, e.g. for new content types. Folder modules can support the author in structuring a course and in applying configurations to a set of modules. For example, a time limit for a sequence of learning units may be applied, or the order of the units may be randomized. Or a module may influence the flow of content by defining a conditional jump or a termination. Any module can own its own scripts (in JavaScript) to realize custom behavior. For example, a content module can use a script to realize adaptation methods, or a jump module can use a script to determine the destination.

### 6.3 Authoring support

The current authoring tools are still incomplete, making it cumbersome and erroneous for the author to create learning content with adaptive behavior. Therefore, we plan to provide authoring support in the following ways:

- Currently, the tasks for a test have to be defined manually within the XML description of a unit. The future system will provide authoring tools to create and manage tests and tasks.
- Currently, the custom scripts have to be edited within an external editor. The new system will provide a source editor with syntax highlighting.

[11] www.moodle.de

- Currently, the tools for the definition of areas of interest for eye-tracking, in particular for images, are quite limited. The revised system will provide more sophisticated AOI-tools for text and a shape editor for easy drawing of AOIs within images.

### 6.4 Page reload

The current learning content viewer uses the classical approach for loading the data of a learning unit by reloading the whole page. This leads to the following problems:

- For every change of learning unit, a complete page reload is performed, resetting all the state variables in the scripts.
- A page reload causes a reconnect to the eye-tracker, producing gaps in the gaze data stream.

While we could easily solve the first problem, using the HMTL5 Web Storage [Hi14b] to store the state variables, the second problem is still unsolved. Therefore we will refactor the viewer component to solve the problem with page reloads. The viewer will be realized as HTML5 Web application that will load all content asynchronously, and will dynamically change its content. This will prevent page reloads when changing the learning unit. Another benefit of this approach is that we can run the application as standalone on a local system without an Internet connection.

### 6.5 Eye-tracker calibration

The calibration quality of the eye-tracking system fades when using it over a long period of time. The main reason for this is the subject who naturally changes their head position in relation to the eye-tracker over time. This impacts the precision of eye-tracking and reduces the quality of the gaze data. To mitigate this problem in our studies, the calibration process needs to be repeated for every experiment. Currently, the Experiment Center of SMI is the only tool we have for calibration. We now plan to implement our own calibration tool running in the Web-browser, in order to calibrate the eye-tracker at the start of every experiment without switching between applications. It also allows easy recalibration during the experiment, keeping the gaze data quality at a more consistent level. This also improves the experiment situation, making it more like a common e-learning situation for the subject. Other solutions like eye-tracking systems with a head fixation create an unnatural position for the subject.

## 7 Outlook

Besides the problems and their solutions, as they are described in the previous section, we plan to further improve our framework. We will extend the adaptability of the system by implementing further functions for customization of the user interface and interaction with the learning content. For general use, the user should be able to adapt the main layout (e.g., showing

or hiding toolbars, changing the slot alignment). Additionally, we plan to use technologies like HTML5 Canvas [Ca14] and SVG [DD14] to be able to include complex and animated shapes in the learning content. Thus, the user will be able to annotate the learning content, as follows:

- Place/draw geometric shapes over text or image areas
- Place/draw arrows and other signs
- Mark text sections with colors
- Draw freehand text and shapes

Future studies, as planned for the project at the University of Education in Freiburg, will evaluate whether these annotation technologies will improve the learning process.

Eye-tracking is not the only possible custom source of real-time gathered information. We want the system to be able to use additional sensory hardware as input device, such as EEG[12] measuring. The cluster on "Brain Computer Interfaces and workload-adaptive informational Environments"[13] of the ScienceCampus Tübingen is already working with EEG. In addition to multiple input devices, we plan to optimize the learning environment for mobile devices, since we want to include mobile devices like tablets or smartphones in future studies.

In the near future, the partner projects will conduct further eye-tracking experiments in the learning environment. Thus, we can access large amounts of gaze data related to learning content. We plan to use statistical methods to analyze these gaze data for higher level gaze events (e.g., reading, skimming), which can be used to trigger adaptive actions to support the learning process.

Finally, we plan to make our experimental adaptive learning environment publicly available to the ILIAS community. Thus, developers and researchers can make use of it for their learning platforms and empirical studies.

## 8 Acknowledgements

The research leading to these results has received funding from the Ministry of Science, Research, and the Arts Baden-Württemberg. The research project is part of the Leibniz ScienceCampus Tübingen "Informational Environments“, an interdisciplinary research collaboration of the Knowledge Media Research Center[14] and the University of Tübingen[15]. The opinions in this publication are those of the authors and not necessarily those of the funding agencies.

[12] Electroencephalography (EEG)
[13] http://www.wissenschaftscampus-tuebingen.de/www/en/index.html?ref=folder244
[14] http://www.wissenschaftscampus-tuebingen.de/
[15] http://uni-tuebingen.de/

## References


[AG10] Suliman Al-Khalifa, H., George, R. P. (2010). Eye-tracking and e-Learning: Seeing Through Your Students' Eyes. http://elearnmag.acm.org/archive.cfm?aid=1833511

[Am13] Amadieu, F., Tricot, A., Mariné, C. (2013). Individual differences in learning from hypermedia: learners' characteristics to consider to design effective hypermedia. http://andre.tricot.pagesperso-orange.fr/Individual_differences_hypermedia.pdf

[Az09] Azevedo, R., Witherspoon, A., Chauncey, A., Burkett, C., Fike, A. (2009). MetaTutor: A MetaCognitive Tool for Enhancing Self-Regulated Learning. https://www.aaai.org/ocs/index.php/FSS/FSS09/paper/viewFile/995/1253

[Az11] Azevedo, R., Cromley, J., Moos, D., Greene, J., Winters, F. (2011). Adaptive Content and Process Scaffolding: A key to facilitating students' self-regulated learning with hypermedia. http://p16277.typo3server.info/fileadmin/download/ptam/1-2011_20110328/06_Azevedo.pdf

[Be14] Berjon, R.; Faulkner, S; Leithead, T.; Navara, E.D.; O'Connor, E.; Pfeiffer, S.; Hickson, I. (2014). A vocabulary and associated APIs for HTML and XHTML. W3C Candidate Recommendation 04 February 2014. http://www.w3.org/TR/html5/

[BHB14] Ben-Naim, D., Ho, S., Belinson, Z. (2014). Smart Sparrow. https://www.smartsparrow.com/

[Bi10] Biedert, R., Buscher, G., Schwarz, S., Möller, M., Dengel, A., Lottermann, T. (2010). Text 2.0. http://data.text20.net/documentation/paper.text20framework.pdf

[Br14] Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., Yergeau, F. (2014). Extensible Markup Language. http://www.w3.org/TR/REC-xml/

[Ca14] Cabanier, R., Graff, E., Munro, J., Wiltzius, T., Hickson, I. (2014). 2D Canvas. http://www.w3.org/TR/2013/CR-2dcontext-20130806/

[DD14] Dahlström, E., Dengler P. (2014). Scalable Vector Graphics. http://www.w3.org/TR/SVG/

[Gü04] Gütl, C., Pivec, M., Trummer, C., García-Barrios, V. M., Mödritscher, F., Pripfl, J., Umgeher, M. (2004). AdeLE: Theoretical Background, System Architecture and Application Scenarios. http://www.eurodl.org/materials/contrib/2005/Christian_Gutl.htm

[Hi14a] Hickson, I. (2014). The websocket api. http://dev.w3.org/html5/websockets/

[Hi14b] Hickson, I. (2014). Web Storage. http://www.w3.org/TR/webstorage/

[RP12] Ruf, T., Plötzner, R. (2012). Interaction Design for Self-regulated Learning with Multimedia. http://www.editlib.org/p/40930

[Ru14] Ruf, T. (2014). Gestaltung kognitiver Unterstützungsangebote in multimedialen Lernumgebungen. Entwicklung einer gebrauchstauglichen Benutzerschnittstelle und empirische Evaluation der Nutzung. Berlin: Logos Verlag.

[Te14] Teiwes, W., SMI (2014). RED eye tracker. http://www.smivision.com/en/gaze-and-eye-tracking-systems/products/red-red250-red-500.html

[WHZ12] Wassermann, B., Hardt, A., Zimmermann, G. (2012). Generic Gaze Interaction Events for Web Browsers: Using the Eye Tracker as Input Device. WWW2012 Workshop: Emerging Web Technologies, Facing the Future of Education.