Elise Eiden, Eric Wallace, May Kristine Carlon
Project Proposal - Development Track

Problem Description

For this project, we will be addressing the problem of integrating metacognitive skill evaluation and development with existing educational tools.

The process of this evaluation and development have been well-documented[1 2 3 4], and boil down to the following strategies:

- Pre-assessments to identify what one does and does not know
- Planning, to ensure understanding of the task and how one might accomplish it
- Identifying the thinking process and how the strategy was selected
- Post-assessments to identify progress
- Reflection journals to assist with remembrance and understanding of the process

However, these strategies are left to be implemented by course creators and/or instructors, who often have a lot of content to develop and present in a short amount of time. Also, it is often difficult to determine which strategies are specific, useful, and mindful of the student and instructors' time.

Existing Solutions

There are plenty of existing tools and that evaluate and/or develop metacognitive skills for students:

Independent Evaluation Tools

- Metacognitive Activities Inventory (MCAI)[5] and/or the Interactive MultiMedia Exercises (IMMEX)[6] allow for a complex evaluation of metacognitive skill. MCAI measures

[1] Blakely, E. (1990). Developing Metacognition. Retrieved May 28, 2017, from http://www.itari.in/categories/metacognition/StrategiesforDevelopingMetacognition.pdf
[2] Peirce, W. (2003). METACOGNITION: Study Strategies, Monitoring, and Motivation. Retrieved May 28, 2017, from http://academic.pg.cc.md.us/~wpeirce/MCCCTR/metacognition.htm
[3] TEAL Center Staff. (n.d.). TEAL Center Fact Sheet No. 4: Metacognitive Processes. Retrieved from https://lincs.ed.gov/programs/teal/guide/metacognitive
[4] Chick, N. (n.d.). Metacognition. Retrieved from https://cft.vanderbilt.edu/guides-sub-pages/metacognition/
[5] Committee on the Status, Contributions, and Future Directions of Discipline-Based Education Research, Education, B. O., Division of Behavioral and Social Sciences and Education, Council, N. R., Singer, S. R., Nielsen, N. R., & Schweingruber, H. A. (2012). Discipline-Based Education Research: Understanding and Improving Learning in Undergraduate Science and Engineering. National Academies Press. Retrieved from books.google.com.
[6] Sandi-Urena, S., Cooper, M. M., & Stevens, R. H. (2011). Enhancement of Metacognition Use and Awareness by Means of a Collaborative Intervention. International Journal of Science Education, 33(3), 323-340.

students' self-monitoring of thinking during problem solving; IMMEX uses Artificial Neural Networks (ANN) and Hidden Markov Models (HMM) to partially reconstruct metacognitive strategies.

Independent Development Tools

- The Self-Assessment Tutor (SAT)[7] utilizes reflection in the areas of prediction, attempt, reflection, and projection to develop metacognitive skills.
- NORMIT-SE[8] is a web-based tutor on data normalization which prompts learners for self-explanation for any new concepts or problems which the students answered incorrectly.

Combined Metacognition Evaluation and Development Tools

- The Reflective Assistant (RA)[9] defines specific metacognitive skills to be targeted, appropriate timings for the training, promotion of skill development, and ways to evaluate those skills.[10] This is a model that clearly provides all of the necessary elements of metacognition tools that have been highlighted extensively in research[11] -- students need to be aware of which skills they are using, and when to utilize those skills.
- The Adaptive Coach for Exploration (ACE)[12] is an open learning environment for mathematical functions which has been extended to track learner's behavior and customize scaffolding accordingly to help improve the self-explanation process where deemed deficient.
- SE-Coach[13] solicits self-explanation from learners and uses a Bayesian network to assess student understanding and determine whether further explanation is needed.
- Geometry Explanation Tutor[14] is a learn-by-doing cognitive tutoring system which uses natural language processing of students' self-explanation responses to guide feedback.

---

[7] Roll, I., Ryu, E., Sewall, J., Leber, B., Mclaren, B. M., Aleven, V., & Koedinger, K. R. (2006). Towards Teaching Metacognition: Supporting Spontaneous Self-Assessment. Intelligent Tutoring Systems Lecture Notes in Computer Science, 738-740.

[8] Mitrovic, A. (2003). Supporting Self-Explanation in a Data Normalization Tutor. Supplementary Proceedings of the 11th International Conference on Artificial Intelligence in Education, Sydney, Australia.

[9] Gama, C. (2004). Metacognition in Interactive Learning Environments: The Reflection Assistant Model. *Intelligent Tutoring Systems Lecture Notes in Computer Science,* 668-677.

[10] Gama, C. A. (2005). The Reflection Assistant Model. In Integrating metacognition instruction in interactive learning environments.

[11] Geddes, L., [., B., & A. (2016, March 02). A Metacognitive Peer Tutoring Model: Linking Thinking, Learning and Performance in a Peer Tutoring Program - Part I. Retrieved May 28, 2017, from https://www.thelearnwellprojects.com/thewell/a-metacognitive-peer-tutoring-model-linking-thinking-learning-and-performance-in-a-peer-tutoring-program/

[12] Bunt, A., et al. (2004). Scaffolding Self-Explanation to Improve Learning in Exploratory Learning Environments. Intelligent Tutoring Systems, Springer.

[13] Conati, C. and K. Vanlehn (2000). "Toward computer-based support of meta-cognitive skills: A computational framework to coach self-explanation." International Journal of Artificial Intelligence in Education (IJAIED) 11: 389-415.

[14] Aleven, V., et al. (2004). Evaluating the effectiveness of a tutorial dialogue system for self-explanation. Proceedings of Seventh International Conference on Intelligent Tutoring Systems, ITS 2004. J. C. Lester, R. M. Vicari and F. Paraguaca. Berlin, Germany, Springer: 443-454.

All of these tools provide standard, peer-reviewed methods to evaluate and develop students' metacognitive skill and can be highly useful for reference in developing our tool. None of them, however, integrate seamlessly with existing educational tools.

Tool Design

We determined that the best way to make a meaningful contribution to the edutech community is by contributing a new feature to an existing open source project. This not only provides a greater community for support during the development process, but also allows the focus of development to be on the metacognition-based feature itself, rather than the architecture/usability details that would be a part of a standalone project. Due to the class's support of Open edX as a sponsored project, we will be developing an XBlock for Open edX.

Implement Reflective Assistant

The first aspect of the tool we plan to tackle is by implementing Gama's integrated Reflective Assistant (RA)[15]. The RA primarily fosters the metacognitive skill of 'knowledge monitoring', which in turn supports the development of other skills, such as 'evaluation of the learning process' and 'selection of metacognitive strategies'.

Our XBlock will be integrated with both the educator and student interfaces of OpenEdX, to work directly with any 'problem solving' aspects of each course. There are two major aspects to the RA: (1) preparation, (2) evaluation. For preparation, the student completes an assessment evaluating problem comprehension and difficulty. The student also reflects on three problem solving strategies relevant to the problem at hand. For evaluation, the student completes an assessment evaluating their use of resources, time management, etc. At the beginning of our development stage, we plan to keep the model as general as possible, so it can be applied to any domain. However, as we move toward completion, we anticipate creating a scaffolding that allows educators to integrate more specifics into the assessments as they see fit.

The assessments will be shown to students in a problem-solving context as indicated by their instructors/course creators.

A significant amount of time will be spent on familiarizing ourselves with the EdX coding style and necessities, to ensure we comply with the established standards and can eventually contribute our XBlock to the community.

Implement Learning Profile

[15] Gama, C. (2004). Metacognition in Interactive Learning Environments: The Reflection Assistant Model. Intelligent Tutoring Systems Lecture Notes in Computer Science, 668-677. doi:10.1007/978-3-540-30139-4_63

Once we have made significant progress on the XBlock itself, we plan to implement a Learning Profile for each student, utilizing the statistics developed by Tobias and Everson.[16] We will compute Knowledge Monitoring Ability (KMA), determined by comparing the student's pre-assessment with their results in solving problems and store the students' results over time, to evaluate progress. We will also compute Knowledge Monitoring Bias (KMB), to account for any deviation that occurs within one's KMA.

These statistics will be stored in a global Learning Profile and will be used to inform the student of their progress, and, as a stretch goal, adapt the content of the assessments to improve the student's KMA and reduce their KMB.

Another stretch goal is to monitor other metacognitive skills based on our assessments, displaying them for the student and adapting the reflections as possible.

Technical Description of Tools, Languages, Other Resources

The primary languages, libraries, and tool set we will use are dictated by the edX initiative who coordinate and maintain the Open edX source code. The Open edX platform (including the course authoring tool, edX Studio) is predominantly built on Django, a web engine written in Python. Thus, the back-end interactions and data management are mostly built in Python, while the front-end client is a web browser with HTML/CSS for content presentation and JavaScript/AJAX for interactivity. The platform maintains its own data storage (XBlocks in edX Studio use MongoDB on the back-end), but our module only interacts with the XBlock API for its data storage needs. The edX initiative has also provided an SDK to assist in templating and testing XBlocks, and pre-built virtual machine images for integrated testing with the entire Open edX software stack.

Languages:

- **Python 2.7** - core language for edX and Django
- **HTML 5, CSS 3** - content and UI markup in web browser clients
- **JavaScript/ECMAScript 5** - interactivity in web browser clients, AJAX posting to Django/Python core
- **GitHub-flavored Markdown** - content markup for README & related docs in GitHub repository

[16] Everson, H. T., & Tobias, S. (2001). The Ability to Estimate Knowledge and Performance in College: A Metacognitive Analysis. Metacognition in Learning and Instruction Neuropsychology and Cognition, 69-83. doi:10.1007/978-94-017-2243-8_4

Tools:

Development:

- **edX XBlock SDK** - software development kit for creating and testing XBlocks (https://github.com/edx/xblock-sdk)
- **Git** - source control (https://git-scm.com/)
- **virtualenv** - sandbox for project-specific Python prerequisites (https://virtualenv.pypa.io/)
- **GNU gettext** - for internationalization/localization support (https://www.gnu.org/software/gettext/)
- **edX i18n-tools** - for internationalization/localization support (https://github.com/edx/i18n-tools)

Hosting

- **GitHub** - public host for Git source code repositories (https://github.com/)
- **Georgia Tech's Open edX** - web-based LMS and course authoring (http://openedx.gatech.edu/)

Testing

- **Open edX Devstack** - Open edX instance for software development and testing, release Ficus.3 from 2017-04-21 (https://openedx.atlassian.net/wiki/display/OpenOPS/Running+Devstack)
- **VirtualBox** - virtual machine host, required for Open edX Devstack (https://www.virtualbox.org/)
- **Vagrant** - virtual machine management, required for Open edX Devstack (http://www.vagrantup.com/)

Integrations / External Resources to Be Obtained

The only external resource we require is the class's Open edX instance. There could be many unforeseen issues when combining our work to create one complete XBlock. Additionally, if our XBlock is unable to get scores/results from the quiz questions (in other XBlocks), this breaks our ability to measure KMA/KMB or other statistics for our Learning Profile implementation.

Our springback plan to address these issues is working to create a simple, more generalized XBlock implementation of the Reflective Assistant. We have many stretch goals (assessment scaffolding, a complete Learning Profile, and assessment adjustments based on that Learning Profile), but if our development cycle does not go as planned, we anticipate doing a very basic implementation that provides all the necessary functions of a metacognitive skill evaluation/development tool.

Project Output

Project Deliverables

1) The final **projec**t deliverable itself should include:
   a) Source code available on public Github repository under Open Source license
   b) All code adheres to edX style guidelines, including support for internationalization/localization so it can be translated into other languages
   c) Detailed “how-to” documentation to guide course creators on how to install and use the XBlock
2) The final **presentation** will be a video overview of the project and demonstration of the use of this software module.
3) The final **paper** which will explain in detail our motivation for creating this module, the research supporting the efficacy of the methods used, and specifics about the implementation.

Intermediate Milestones

1) **UI Prototype** - *due July 2, 2017*
   - model the look-and-feel of the student prompts, i.e. how it appears when implemented within a course
   - model the look-and-feel for the course creators’ interface, i.e. how it appears within edX Studio
   - ideally this prototype should be delivered as a visible-if-not-yet-functional XBlock

2) **Beta Testing** - *due July 16, 2017*
   - install the XBlock in Georgia Tech’s Open edX instance
   - open it up to classmates to test as they build Open edX courseware
   - opportunity for us to fine-tune the learner profile data collection and results

Work Efforts

Responsibilities

Here’s a rough picture of how we’ve divided the work responsibilities among the three team members. Fortunately, we are all capable of assisting in any of the tasks, so we expect this to be an easy collaboration.

Elise:

- JavaScript interactions
- Learner Profile development

- documentation

Eric:

- user interface HTML/CSS from mock-up to final implementation
- how-to documents for course creators on installing and using the XBlock

May:

- Python interactions
- edX integration
- final paper

Action Plan

Phase Overview

Task	Assignee	Wk1	Wk2	Wk3 (IM1)	Wk4	Wk5 (IM2)	Wk6
Draft scaffolding and prompts	Elise	x	x	x			
Prototype UI and interaction spec	Eric	x	x	x			
Research API and Python components	May	x	x				
Develop XBlock’s shared data model	May		x	x	x		
Coalesce the XBlock code	Eric			x	x	x	x
Develop Learner Profile	Elise			x	x	x	x
Extend UI for Learner Profile	Eric				x	x	x
Increase complexity, as possible	entire team				x	x	x
Complete five weekly status checks	entire team	x	x	x	x	x	
Finalize Git repository and docs	Eric					x	x
Author final paper	May				x	x	x
Record final presentation	Elise				x	x	x

Detailed Action Plan

Week 1

Elise: initial draft of evaluation scaffolding/prompt questions

Eric: initiate shared Git repository; create initial user interface HTML/CSS

May: complete XBlock coding tutorial and API research

Team: prepare development toolset; get access to the class Open edX instance and an edX Studio course; weekly status check

→ *Deliverables:* Week 1 Status Report

Week 2

Elise: refine the evaluation scaffolding/prompt questions; begin JavaScript integration

Eric: build UI into initial XBlock format; create spec for data sending/receiving with AJAX and Python back-end

May: develop data model which XBlock uses to store and share the stats

Team: finalize scaffolding/prompts; weekly status check

→ *Deliverables:* Week 2 Status Report

Week 3

Elise: begin developing Learner Profile features/formulas

Eric: integrate the UI prototype as a functional XBlock; implement edX's accessibility and i18n guidelines

May: continue data model/Python-side development

Team: test the UI prototype; weekly status check

→ *Deliverables:* Intermediate Milestone #1 (UI Prototype), Week 3 Status Report

Week 4

Elise: solidify and begin developing Learner Profile functionality

Eric: create UI for Learner Profile's stat display; ongoing integration of UI/JavaScript/Python components into XBlock

May: solidify data model for gathering statistics; Python-side integrations

Team: readiness “gut check” and discuss opportunities for refinement; weekly status check

→ *Deliverables:* Week 4 Status Report

Week 5

Elise: finalize Learner Profile functionality; begin draft of final presentation

Eric: thorough testing of XBlock functionality in multiple scenarios; review edX security guidelines for avoiding cross-site scripting attacks

May: begin draft of final paper

Team: “beta testing” the XBlock together; discuss opportunities to fine-tune and increase intelligent response; weekly status check

→ *Deliverables:* Intermediate Milestone #2 (Beta Testing); Week 5 Status Report

Week 6

Elise: complete writing and recording of final presentation

Eric: complete packaging/presentation of project repository, including integrating user how-to documentation, reviewing Open edX style guidelines, and submitting to edX for review

May: complete writing final paper

Team: collaboration and finishing touches on paper, presentation, and project submission

→ *Deliverables:* Final Project; Final Presentation; Final Paper