A Tool for Collaborative Annotation
Harvard University’s Collaborative Annotation Tool, commonly shortened to CATool, gives faculty and students the ability to collaboratively annotate text, images, audio, and video. Students and faculty highlight points of interest and discussions with text or media annotations. Other users may create additional annotations, or reply to previous annotations, thus creating threaded discussions around common points of interests.

Integrated with Learning Management
The CATool was originally designed to integrate tightly with Course iSites, Harvard University’s proprietary Learning Management System. Course iSites is a powerful platform, providing faculty and staff with easy-to-use tools, all integrated with Harvard data. Course iSites also provides extremely flexible permissions management, allowing course administrators to quickly establish which users, both within and without the Harvard community, can access their course site and tools. By taking advantage of the Course iSites framework, developers could rapidly implement new features of the CATool without coding their own schemas for authentication, authorization, and course and student data.

The Move to Open Source
As interest in using the CATool grew at Harvard and beyond, so did interest in making it available outside the Harvard environment. At the same time, Harvard University Information Technology (HUIT) has begun to foster a culture of collaborative software development both within the institution and with partners outside the University. The CATool seemed like an excellent opportunity to share our work with the community.

Unfortunately, early design decisions resulted in a software architecture that is difficult to untangle from the Course iSites framework. The CATool also relies on third-party technologies, such as JW Player, with software licenses that make it challenging to promote outside development.

We embarked on a project to rework the CATool as an open source venture. By open sourcing the CATool, we could ensure that our code could be integrated with a wide variety of platforms, be easily available for distribution, and ultimately, we hope, take advantage of community-driven software development.

Trials and Tribulations
Moving to open source required fundamental shifts in our current methodologies because we were now coding for a global audience of developers rather than our

Figure 1. Harvard University’s CATool, shown here in its original Course iSites framework, allows faculty and students to annotate and discuss common points of interest within a variety of media, including text, images, audio, and video. Here a student comments on a recording of Wolfgang Amadeus Mozart’s singspiel, “The Magic Flute.”
Harvard colleagues. We needed to rethink (1) our development processes, (2) how we architected the software, and (3) how we would ultimately distribute the software. Of course, none of these methodologies are new or necessarily specific to open source practices; they can be applied to any software project.

**A Change in Development Methodologies**
As we started planning our project to open source the CATool, we quickly realized that coding for an internal audience of Harvard coworkers would be much different than coding for an external audience that might span the globe. Assumptions of internal knowledge could no longer be made. Shortcuts could no longer be taken. Ad hoc releases were no longer acceptable. Good documentation was a must. We reinvented our software development methodology to take advantage of Agile principles, and standardized on a variety of best practices, including style guides, test-driven development, peer code reviews, and quality assurance.

**A Change in Architectural Methodologies**
We also realized that some of the technologies used in the original CATool and Course iSites platform were not as well suited for open source development. We switched from using Oracle as our primary database to MySQL. We could not distribute JW Player, a commercial video plug-in, along with our code, so we migrated to JPlayer, an open source alternative. We replaced Ext JS for client-side scripting with jQuery and Backbone.js because we found that they had larger communities of support and were more conducive to the open source model. Finally, we abstracted much of the CATool’s software architecture so that the tool is easily integrated with a wide variety of technologies, such as OpenID authentication.

**A Change in Distribution Methodologies**
As we continue moving forward with open sourcing the CATool this summer, we still face many questions, particularly concerning the distribution of the software. Our questions include:

- Under what licenses do we distribute the software?
- Where should we host our source code?
- What processes can we model for distributing other Harvard projects?
- How do we engage the open source community to participate in our project?

**A Model for Other Tools**
We hope that the lessons we are learning can be applied to other Harvard-developed applications, and that other institutions can take advantage of what we have learned. A second project is underway to open source Harvard University’s Quiz Tool, another application integrated with Course iSites, using the methodologies and technologies described above. We anticipate that the source code for both the CATool and the Quiz Tool will be available by the end of summer, 2012.

Please visit us at:
http://blogs.law.harvard.edu/acts/catool/

---

**Figure 2.** Early design and technology decisions made it difficult to extract the original CATool from its Harvard environment (left panel). The CakePHP core was tied to Harvard’s Oracle and authentication platform. Extensive use of Ext JS in the Course iSites platform and the use of the commercially licensed JW Player restricted distribution of code to outside the University. We addressed these issues when open sourcing the architecture (right panel). We replaced Oracle with MySQL; we abstracted our authentication and other schemas to enable additional sources of information; and we replaced our client technologies with open source alternatives, which permit the software to run on a variety of platforms, including mobile devices.