A Design Case of an Enterprise-Wide Learning Management System

Michael C. Johnson, Larry L. Seawright, and Jason K. McDonald

Introduction

In this case we describe the in-house design and development of an enterprise learning management system (LMS) at Brigham Young University (BYU). The purpose of the project was to replace a commercially available LMS that was becoming too costly as well as unresponsive to the interests of faculty and other stakeholders. In the case we discuss why administrators made the decision to develop a complex piece of software using university resources instead of relying on other commercially available products. We also describe their goals for the project and how we attempted to meet those goals by designing the new system on a foundation of existing components and by focusing on the most frequently used functions from the previous LMS. A central feature of our discussion is how we and other participants made decisions in a high-stakes environment of multiple stakeholders and a compressed timeline, which had an impact on the emerging design of the product. We also examine some of the challenges that arose among members of the design and development teams during the course of the project as pressure on the team became more intense.

Throughout the case we focus primarily on the actions of the design team located in the university's Center for Teaching and Learning. This team included three instructional designers (one of whom is the lead author on this paper), two members of the Center's leadership team (one of whom was responsible for production and project management and is the second author on this paper), and a user experience

M. C. Johnson (✉) · L. L. Seawright
Center for Teaching & Learning, Brigham Young University, Provo, UT, USA
e-mail: mc_johnson@byu.edu; larrys@byu.edu

J. K. McDonald
Instructional Psychology & Technology, Brigham Young University, Provo, UT, USA
e-mail: jason@byu.edu

M. J. Bishop et al. (eds.), *Handbook of Research in Educational Communications and Technology*, https://doi.org/10.1007/978-3-030-36119-8_31

designer. We also describe interactions between us, as the designers, and the development team, which included three, full-time, permanent developers as well as two temporary developers.

Context

Brigham Young University is a private, religiously affiliated university in the United States. The university employs over 1500 full-time faculty, and between 1500 and 2000 part-time faculty, to teach about 33,000 full-time students. BYU has 11 colleges and schools that serve these students. It is accredited through Northwest Commission on Colleges and Universities, a self-governing body of 163 US and Canadian higher education institutions located in the northwest.

BYU has been an early adopter of academic technology. For example, among other academic systems it purchased or created, BYU developed and implemented one of the first telephonic class registration systems in the late 1970s. This resulted in the ultimate development of a custom student information system. The university was also an early adopter of learning management systems, deploying its first in 1999, licensed from a company that evolved into one of the major LMS vendors. This system was used by the university for 10 years.

When BYU began using this product, it received immediate vendor response to not only problem reports but also to new feature requests, because the university was one of the first, large organizations to use the product. However, over the next 10 years, as the LMS grew in popularity, the vendor developed more formal processes for both problem resolution and vetting feature enhancement requests. Additionally, it added a pricing structure for adding custom features that the company determined to be outside the scope of what would be added via the normal feature enhancement process. This meant that changes BYU requested were causing significant financial charges from the vendor. And updates or fixes were being delayed due to the company prioritizing them to fit into a larger product road map.

In response to growing dissatisfaction over both pricing and vendor responsiveness, a university committee of faculty, administrators, and information systems professionals was formed to study the LMS needs of the university and to consider whether to continue to use the existing LMS or switch to another vendor. During that review process, one of the vice presidents on the committee noted the existence of several, internally developed tools that performed some of the functions typical of most LMSs: a Syllabus Builder, a discussion forum (called Digital Dialog), a gradebook, and test scoring system used in the university's Testing Center. Additionally, the university had recently created a system to store and report the learning outcomes for each of its programs and courses. The vice president making this observation noted that more than half of the most used features of the current LMS were duplicated by those tools. He wondered if they could be combined together, supplemented with development to create the remaining most-used features and result in a proprietary learning management system.

Those tools had been developed by the BYU Center for Teaching and Learning in response to various needs that had been observed across campus. Because the Center reported to the vice president who originally noted the overlap with LMS features, he requested that personnel from the Center respond to a request for a proposal for converting the existing tools into an LMS with the following features: a content platform, a communication tool, an exam engine, a gradebook, a schedule, and a syllabus. Additionally, it should link to the BYU Learning Outcomes website as well as continue to display syllabi on the university's website, without authenticating into a specific course (this requirement was to comply with a university commitment made during an accreditation visit to have publicly viewable learning outcomes and syllabi).

In response to his request, Center personnel conducted a brief feasibility study and determined that it could combine the existing tools and add additional functionality, with an estimated initial project duration of 24 months to create a beta product that could be tested. Based on this estimate, the university committee charged with reviewing university LMS needs recommended that the university proceed with development of a proprietary system to be known as BYU Learning Suite. Two primary reasons were cited for proceeding with the development of its own LMS instead of licensing an existing LMS. First, the university could integrate custom-developed learning tools with existing custom university data sources, and second, it could adapt and update learning tool features more quickly than could a commercial vendor. The university's academic vice president and chief information officer convened a meeting in July 2010 to review the viability of the proposal. It was approved. However, due to contract issues with the current vendor, during late 2010, the duration of the project was reduced from 24 months for delivery of a beta product to about 18 months for delivery of the initial public version. This decision would have significant impact on the project and all the people involved, from designers and developers to users and sponsors.

Initial Design Considerations and Guiding Principles

As mentioned, the major components of Learning Suite originated as either standalone products or as components of other systems. University stakeholders originally hoped that because these systems already existed, they could easily be repurposed for the new LMS. But, generally, this proved to not be possible because of differences in the technical architectures of the different products. For example, the Center for Teaching and Learning had previously developed the Syllabus Builder in response to BYU's Accreditation Board, who wanted measures of course learning outcomes across the university and what instructors were doing to align their course activities with those outcomes. Because each syllabus was stored centrally, information about any course using the LMS could be reported to accreditors. Although university administrators hoped the existing Syllabus Builder could be repurposed

for the new system, because of differences between the system architectures, developers had to create a new Syllabus Builder from scratch.

Even in cases where the underlying technology between existing components was compatible, we as a design team had difficulty repurposing the original products. Our desire to create a unified user experience meant that the surface layers of each product would have to be significantly redesigned. This was more than the styling of various elements, such as their colors, button styles, etc. While the existing components had designs based on providing an easy user experience in a standalone environment, those designs were not conceived of as being integrated into a larger whole. So one of our primary activities as a design team was to consider how each of these components fit together into a coherent navigational scheme and how they were presented in the context of one course as well as in multiple courses. Where possible, we reused individual design elements, but generally either the new context of use was different enough, or enough additional experience had been gained since the original designs were created that we redesigned more components from the existing tools than administrators had hoped.

As it became clear that significant redesigns would happen, we established some guiding principles for how the unified system should behave. Our primary goals were to keep the user experience easy, intuitive, and, as much as possible, based on existing common academic workflows to which faculty were accustomed. We also attempted to build a product that was considered fast, intuitive, and engendered confidence. Practically speaking, this meant we adopt guidelines (Table 1).

We provide some examples to illustrate how we applied these principles. When someone creates an assignment, for example, we attempt to minimize the number of actions users have to take. So when the original assignment record is created, corresponding records are automatically added in the schedule (on the due date selected when the assignment was created) and in the gradebook. Assignments can also be edited from either of these locations as well as in their original record. Another example is the process of adding discussion prompts. When users create a prompt in the discussion forum, they are able to choose that prompt for all selected users or instantly create multiple threads for each individual or group in the course using the same prompt. Broader than the design of an individual component, these principles also led us to integrate the LMS with other proprietary systems provided by the university's IT staff, including the student information system, the Learning Outcomes website, the Student Ratings website, and the BYU catalog and class schedule. We also integrate with commercial applications used by the bookstore (for a student's booklist) and the BYU library (for copyright reserve requests and information about subject matter librarians).

These principles proved to be a good foundation for creating an LMS that seemed to meet important stakeholder goals. BYU's central administrators achieved their goal of closely associating the LMS with two important sources of accreditation information—learning outcomes and syllabi. Faculty stakeholders benefited because the LMS also helped them maintain those two accreditation sources as part of their everyday workflow within the system. In addition, the system also gave faculty the basic components that 80% had used most of the time in the previous

Table 1 Guidelines we adopted to support our design process

Guideline	Specifics
Simplicity	Design each page from the ground up Remove or hide all non-essential functions Define good defaults
Facilitate communication	Between instructor and student Student to student Instructor to instructor
Every millisecond counts	Treat users' time as sacred Even small performance gains are worthwhile Streamline data entry and other tasks Do not make users do things twice
"Functional" is not the same as "finished"	An unfinished feature might be worse than no feature at all The last 10% of a feature makes a huge difference Polish breeds trust
Start up with style, fail with grace	There are three states to every screen: blank (zero state), normal, and fail The initial screen (with no data) is the first impression Do not abandon users in their moment of need
Have a pleasing "personality" and allow for pleasant surprises	Labels, instructions, notifications, warnings, etc. should always be courteous, cheerful, and concise Layouts should be clean, clear, and calm Interactions should seem loyal, helpful, friendly, and fast

product. Student stakeholder needs were met because most faculty were using the same LMS, in mostly the same way, so it was easier for students to find their courses as well as to develop a consistent mental model of how a BYU course would work. Tight integration into other university systems also benefited students as it meant that changes made by faculty were replicated across other systems and LMS components, eliminating redundant data entry for faculty and enabling viewing across those systems by students.

Description of the Design

To provide a feel for what the user experience is like while using Learning Suite, we provide descriptions of major components. Using Learning Suite begins as one authenticates into the system and sees the list of courses available for the current semester (Fig. 1). Instructors see all courses to which they are assigned through the university's academic management system. Students see a list of all their courses but can only access those which instructors have set up and published. Learning Suite also provides an option to view courses from previous or future semesters, under the same conditions of availability as for the current semester. Additional management tools are also available to instructors. These include creating test

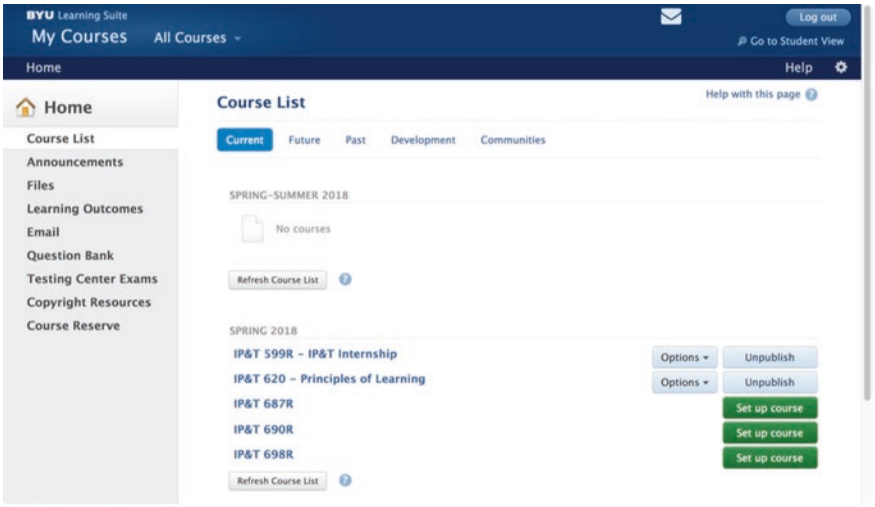

Fig. 1 The Learning Suite course list

questions or uploading files, sending messages, or adding announcements. All of these options are also available within individual courses as well.

When instructors select a course for the first time, they are walked through a set of screens to set up the various aspects of the course, if they elect to use that capability at all. Even choosing Learning Suite as the management tool for that course is an option. Instructors can either elect to make a Learning Suite course available to students, or they can insert the URL of a course hosted elsewhere, such as in another course management system or on an open website (Fig. 2). If they choose Learning Suite for their course, they then set up discussion forums (known within the system as Digital Dialog), exams, content pages, and a syllabus. In each case the first choice is a) whether to open the specific capability using Learning Suite functionality, b) point to an outside URL, or c) to not include it at all. If instructors select to use Learning Suite functions, they can begin to add information, copy a structure from another course (either one of their own or from another instructor who has made a course available to be copied), or return another time to complete either of the previous tasks.

Upon opening a course, the most prominent display is the dashboard, which provides a summary of the upcoming schedule and announcements (Fig. 3). The major course sections can be selected by a set of tabs across the top of the interface, with navigation within a tool available on the left. For example, in the case of the Home tab, the ancillary navigation includes the dashboard, access to the course email system, and, for instructors, course tools like global settings and creating student groups.

Instructors are always able to access every tool through the tabbed navigation. If they have not set up that tool, by selecting it they are presented with the settings information as described earlier. If the tool is set up, they are presented with options

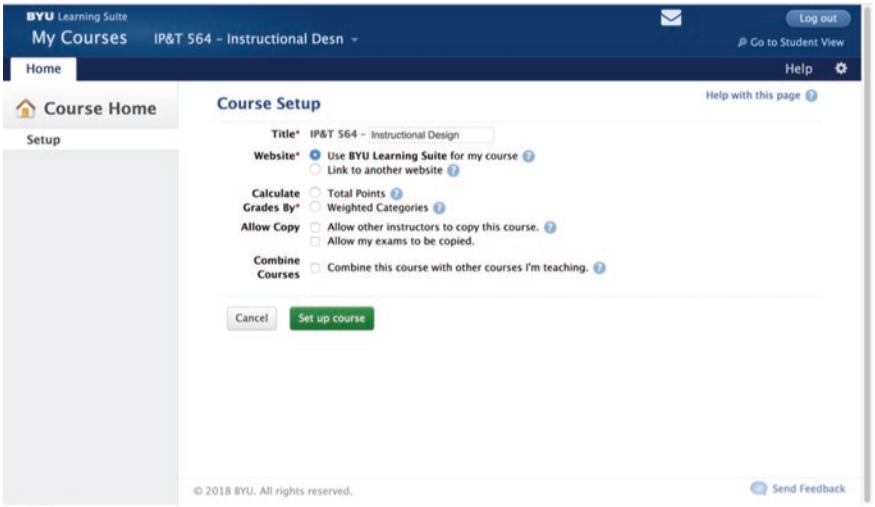

Fig. 2 Initial course setup

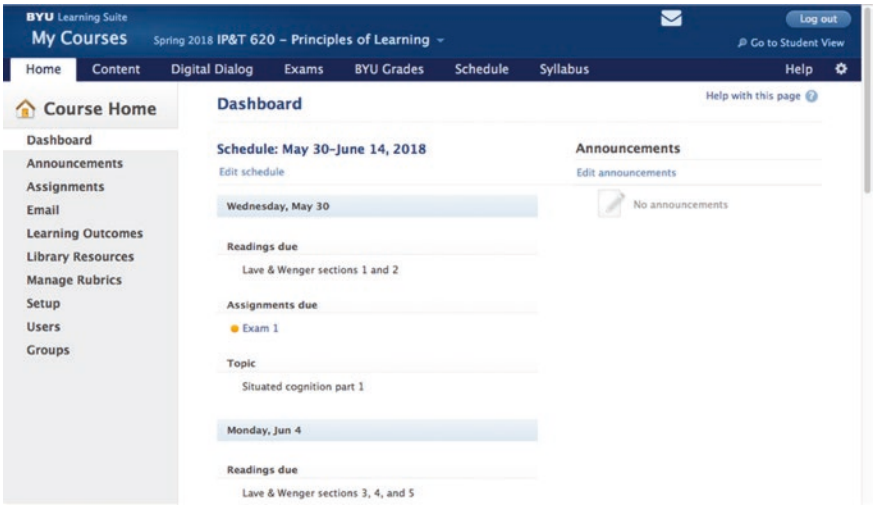

Fig. 3 The course dashboard

to modify settings for that tool or add information or structure to display to students. Students only see tabs for tools that instructors have set up. To give a general sense of the user expereince in BYU Learning Suite, we describe two of these tools in more depth, the gradebook and the schedule.

The initial gradebook display is a spreadsheet that lists all students in rows, with assignments in columns (Fig. 4). Instructors can select any cell within the spreadsheet and add a score directly. In the case of computer-scored assessments, scores

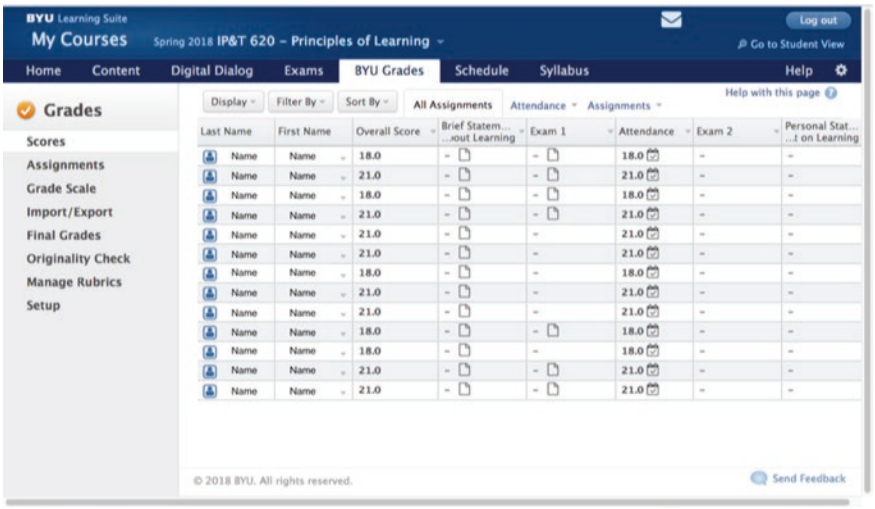

Fig. 4 Initial gradebook view

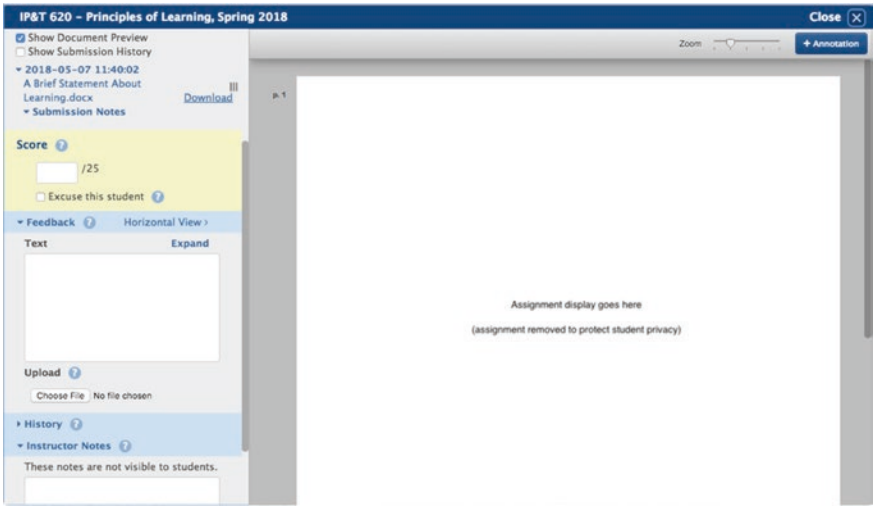

Fig. 5 Assignment grading

are automatically entered by the system. The gradebook automatically calculates a student's final grade based on scores for individual assignments as well as other rules determined by the instructor, some of which include assignment/category weights and grading scale distributions. For an individual assignment, instructors can select the "submitted assignment" icon to see documents that students may have submitted and any comments the student has included with the assignment, offer feedback in return to the student, or grade the assignment using a rubric (Fig. 5).

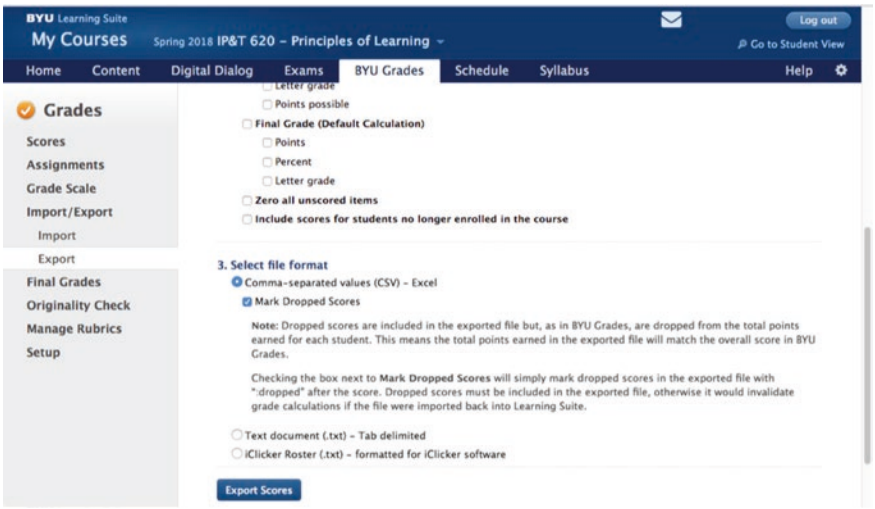

Fig. 6 Exporting grades

Along the left, instructors have additional navigation to create/modify individual assignments, customize the grade scale, export grades, send final course grades to the university information system, perform plagiarism checks, create/modify assignment rubrics, or modify gradebook settings. Selecting any of the ancillary gradebook tools replaces the spreadsheet interface with the requested information, for example, choosing to export grades allows instructors to select through checkboxes what information to export along with choosing among various export formats (Fig. 6).

The schedule is a large, multi-paneled calendar that instructors can customize. A row, representing every scheduled class period for the semester, is pre-populated from the university's master calendaring system. Rows are also automatically added for holidays, other university-wide events, and final exams. Instructors can add additional days if they want. By default at least one column is always available that displays due dates of any assignments (due dates are initially set when instructors create the assignment). But instructors can also add other columns to display whatever information they want students to see about each class period for the semester (Fig. 7). By double-clicking on an open cell, instructors can add any open text they want, including links out to other sections of the course or elsewhere on the Internet. Once information is available in a cell, it can be reopened and modified, dragged to another cell, or deleted. Dragging an assignment to another cell automatically changes its due date in the grading system.

Beyond these components, Learning Suite also includes the following functions:

- Content repository: allows instructors to create simple HTML pages (using a WYSIWYG editor or editing HTML directly). Pages can also embed standard file types.

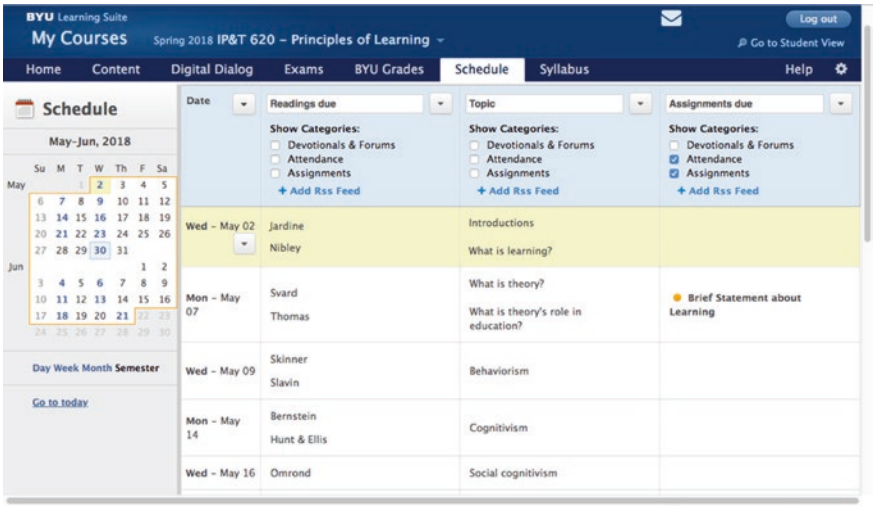

Fig. 7 The course schedule

- Messaging systems: an internal messaging system allows for in-course communication between faculty and students (more similar to an internal email system than a reproduction of the discussion forum). An email component was also made added for messaging outside of the LMS.
- General course announcements: permits posting of announcements that can be displayed when students login to Learning Suite or sent to them through email.
- File system: allows for storage of files that instructors include in their course (e.g., PDFs, PowerPoint slides); files can be linked to in multiple courses using the same location in the file system.
- Groups: allows instructors to segment classes into smaller groups, primarily for discussion purposes.
- Course copy: allows instructors to copy a course or course components from semester to semester. Instructors can also open their course for other instructors to copy.

Effects of the Shortened Timeline on the Product

As noted earlier, within a few months of the decision to create a custom LMS, the central administration made another decision to not renew a license for the existing LMS at the conclusion of its contract. That meant that our replacement product needed to be completed, tested, and rolled out to users in less than 18 months. The effects of this change led to some significant stress for us as well as the development team, but also led to some process innovations that allowed us to still meet the deadline to which we were accountable.

The tight time frame meant that development of LMS components needed to start almost immediately. This was a significant problem for us as a design team; there was little time for any product design beyond basic functionality, and there was not time to adequately iterate any initial design ideas. To compensate, we chose to use existing research on the characteristics of usable learning management systems instead of engaging in our ideal plan to investigate users' workflows and processes directly. We were able to conduct focus groups with faculty and students to get their opinions about the designs of the existing products being integrated into the new system. Using paper prototypes created for those focus groups, we developed high fidelity mock-ups and delivered them to developers so they could begin the actual programming for system components.

However, once designs were handed off to the development team, there was little time to verify that our design intent was actually implemented in what was produced. We initially assumed this would be acceptable because some of the development staff had previous experience creating learning management systems, and we believed their experience could help them fill in the gaps in the designs they received. But what often happened was developers replaced even these too-simple designs with still simpler interactions, to reduce the time needed to create a stable product that could be released. We, as designers, were sometimes not included in this reduction process, however. When a test product was developed enough to be shown to us, we were sometimes surprised that our designs had not been followed. Even more, we were told to keep providing developers with designs so they could stay productive, even though we had not adequately reviewed much of their work that was based on our previous designs.

In fact, this caused a significant degree of conflict during the initial development cycle. We believed we were being asked to approve a product even though it had not been subject to a review that would ensure it was meeting our design specifications. But developers believed they were not receiving designs fast enough to implement them all by the deadline. The resulting stress and inter-team conflict resulted in even less efficiency and effectiveness than both sides were already worried about. Our ability to meet the deadline at all was at risk.

To resolve the conflict, the two teams had to work together on a new process that addressed these concerns at least enough to allow for successful completion of the project. As a design team, we agreed to skip a low-fidelity wireframe stage of our process and only produce high-fidelity prototypes that were annotated with significant information developers needed to know. We created these prototypes using a set of templates created by a graphic designer and that the instructional designers on the team could use to rapidly add interaction elements. We also created a living style guide that developers could use for information about interface elements such as button colors and size and dialog box interactions. We call this style guide living because, although both we and the developers preferred to have all these decisions made before actual coding began, the work patterns created by the compressed timeline led to a just-in-time approach to making new interface decisions only when it became necessary for developers to complete their work (Figs. 8 and 9).

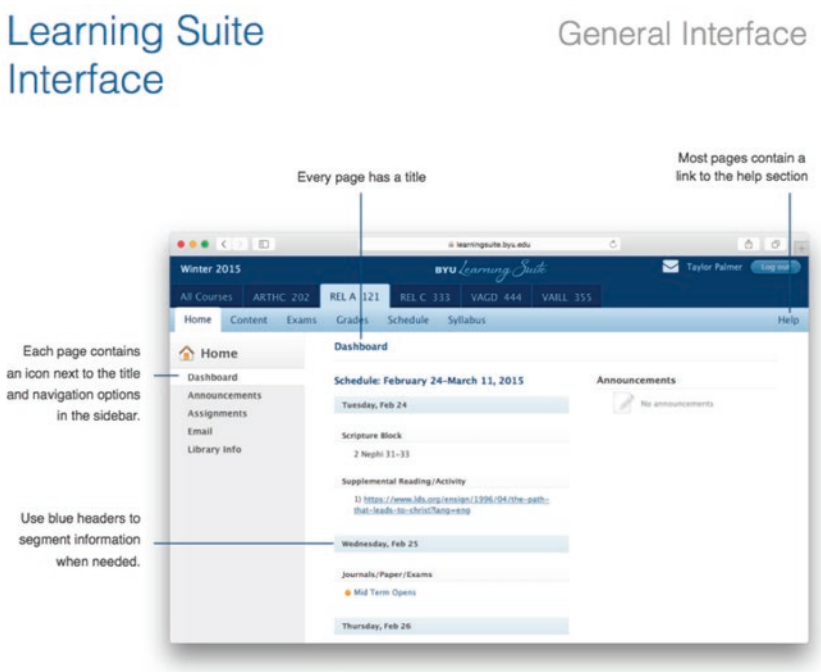

Fig. 8 Example page from BYU Learning Suite style guide

Initial Implementation of Learning Suite

When the Learning Suite LMS was released, it was not as stable or as easy to use as hoped. Initial months after release saw a number of bug fixes and other optimizations. We were still not significantly involved in this phase as designers, because the pressures to make the system stable meant developers had to implement whatever the simplest solution they could create on their own in order to solve the immediate problem.

After the first push for stability, there were fewer pressing deadlines, and we could take more time to produce our design documents for new features, as well as to work more interactively with developers to explain our decisions. Additionally, we became more familiar with the processes used by the developers, allowing us to more clearly articulate the intent of our designs. Over time, the designers and developers have begun to trust each other and work more collaboratively with one another.

This did not completely eliminate all the challenges in developing the Learning Suite user experience, however. For example, when we designed the email tool, we initially sent all messages as a blind carbon copy, meaning that while someone could send a message to multiple recipients, any responses back from a recipient

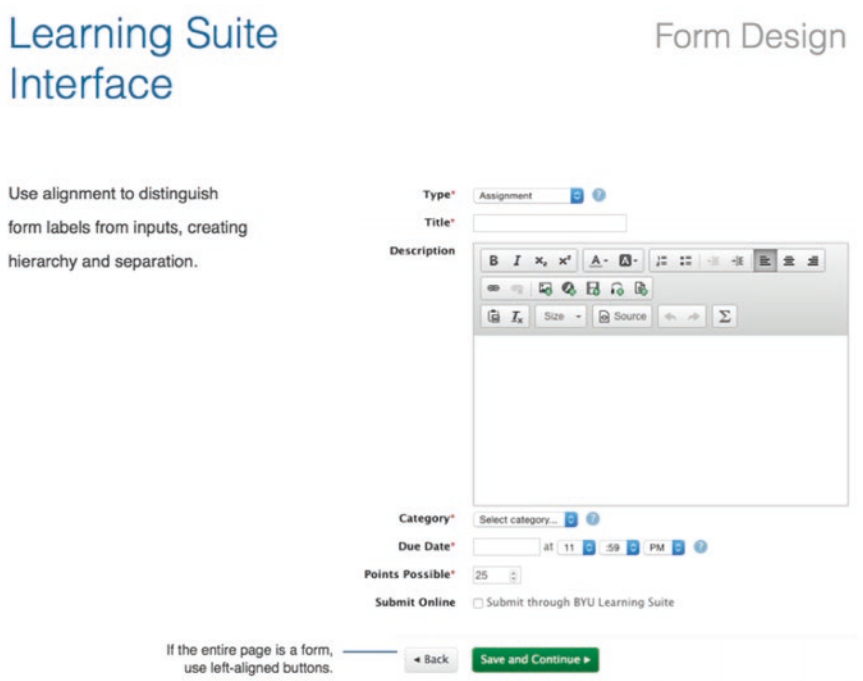

Fig. 9 A second example from the BYU Learning Suite style guide

were only sent to the person originally sending the email and not to the full list of recipients. This did not facilitate communication as well as instructors and students desired, so we built a messaging system internal to the LMS that would more easily allow for many-to-many communication. Yet this still did not meet the expectations of those using the product; they wanted a simple way to respond to messages through their email client and did not want to be required to go into the LMS to reply to messages. Yet, due to lack of infrastructure and competing priorities, we have not been able to add this feature.

Even with challenges such as these, with each successful semester of use, faculty and students have begun to trust the Learning Suite system, especially during high usage periods such as at the beginning and end of each semester. The initial usage of the LMS among faculty was at the same percentage as the previous system—80%. Over the 5 years of usage, faculty usage has grown to above 85%. During that period, faculty and student calls to the service desk for usage support ("how do I do this"-type questions) have grown by 30%, while support calls ("something doesn't work") have gone down by 20%.

Conclusion

In this case we have described the design of BYU Learning Suite, a proprietary learning management system. We described the context and constraints of the design and development of the tool to give a deeper understanding into how these factors influenced the design, design processes, development process, and the relationships between the design and development teams. Finally, we described the challenges faced in implementation.

Though the initial time constraints were tight, the further reduction in the timeline created a seemingly insurmountable obstacle. This in turn further magnified the usual differences that exist between designers and developers. Yet this case shows that, with some innovation and compromise in the product and process, the team was able to complete the project, though initially not the product any had hoped for. Still through iterations and increased collaboration the product has become much more stable, usable, used, and generally accepted. And the design and development team has learned to work together in a much more cohesive manner.