Massachusetts Institute of Technology

BLACKBOARD 9.1 EXPERIMENT: ANALYSIS AND RECOMMENDATION

Derek Jaeger
Education Systems, IS&T
August 12th, 2011

INTRODUCTION

At the direction of the Steering Committee for Learning Management Systems, IS&T field-tested a customized version of the Blackboard 9.1 Learning Management System in Spring 2011. Fourteen courses spanning six disciplines were included in this evaluation, representing the participation of thirty-three course administrators and over six hundred MIT Students.

In order to mitigate the dissonance often associated with product migrations, the version of Blackboard 9.1 implemented at MIT was heavily customized to follow a workflow logic closely paralleling that of Stellar, MIT's current Sakai-2 based Learning Management System. In addition to following or complementing Stellar workflow as closely as possible, the application UI was also customized to present a look and feel that was consistent with the production Stellar environment.

As part of this evaluation, a rigorous technical assessment of the Blackboard platform was conducted, and feedback was collected from course administrators and student users regarding their experience with the application. This document presents the highlights of our Blackboard technical assessment, as well as an overview of the feedback gathered from users via focus group sessions and online surveys.

The results of our technical assessment are consistent with community feedback, and suggest strongly that at this time Blackboard is not a viable LMS option for MIT. This conclusion is informed by numerous technical drawbacks documented by the development team responsible for supporting and maintaining Blackboard at MIT, coupled with a number of shortcomings reported by users in comparison with Stellar.

The recommendation at this time is to halt further experimentation with the Blackboard platform and shift resources to the continuing development of the Modular Service Framework, which is intended to gradually replace existing Sakai-2 based Stellar functionality with a set of discrete, flexible web services driven by a common data framework and based on a standardized set of APIs.

This document is organized as follows:

Part I presents a technical analysis of the system

Part II presents feedback gathered from the community

Part III presents the final recommendation, coupled with an overview of the Modular Service Framework

Part I: TECHNICAL ANALYSIS

The technical analysis of Blackboard 9.1 revealed several significant issues with respect to the product, highlighted by shortcomings documented within the following areas:

- Supportability
- Maintainability
- Core functionality
- Extensibility and customizability
- Value-added functionality

Each of these areas is discussed in order in the following narrative.

A. SUPPORTABILITY
The ability to support the application in production while addressing performance bugs

General:

- The Blackboard application operated more or less consistently during the evaluation, with isolated exceptions. However, it is worth noting that the course load during the evaluation was comparatively light, as it corresponded to limited departmental use-case representation.

- Individual technical support interactions with our primary Blackboard technical contact, George Kroner, were useful, although he and his team mentioned many features of Blackboard that were only documented on an internal corporate wiki and therefore not available to us.

- Blackboard has a broad user community, and collaborating with other users via the Blackboard mailing lists is generally straightforward.

Difficulties:

- Backboard's response to and resolution of bugs identified within the core software is slow. As customers, we do not have access to the core code, and are dependent on Blackboard's development cycle, over which we can exercise limited control. This hinders our ability to address key bugs and significantly erodes response/remediation time in comparison to the process followed within Stellar.

- Product documentation has been difficult to find, poorly organized, inconsistent, sometimes obsolete, and occasionally inaccurate. Examples:

    1. The online documentation is split across two websites (edugarage.com and behind.Blackboard.com) and several sections (knowledge base, forum answers, and PDF document library) only some of which are actually searchable. Text search within PDF documents, for instance, is completely unsupported. Blackboard has acknowledged that

lack of adequate documentation is a common complaint among users, and that they are "actively developing" improved documentation architecture that will "unify, centralize, and index all of Blackboard's existing resources". No time line has been specified for completion of such an overhaul.

2. The Blackboard Knowledge Base contains inaccurate information regarding critical aspects of customization support: we found that the documented method of exempting certain custom files from being deleted during upgrades was applicable only to older versions of Blackboard.

3. According to our primary Blackboard technical contact, George Kroner, the Custom Authentication documentation for release 9.0 contained incorrect information about the required Java interface. As of the last date of our use of the documentation, this information is still referenced for release 9.1SP3 and beyond. In subsequent discussions, Blackboard has stated that inadequate support for custom and modular authentication systems is a broad-based pain point, adding that "changes to Blackboard's authentication architecture [are] planned for later this year".

B. MAINTAINABILITY

The ability to maintain the product while managing upgrades and adapting to changes

Difficulties:

- The upgrade path is difficult at best. Service pack and license upgrades require rolling back key settings and merging them back in later—a time-consuming and tedious process. As an example, our custom authentication module, which is required for Touchstone authentication, is particularly impacted by this limitation. Blackboard documentation directs users to remove any custom authentication modules before applying application-level upgrades, and reinstalling them after maintenance is completed. In practice, this is a protracted and significantly error-prone process.

- Custom files and directories are deleted during upgrades by default, and Blackboard template files (which are intended to be edited) are overwritten during upgrades. Both of these factors work in conjunction to significantly complicate the upgrade process, essentially requiring that modifications be maintained as a set of diffs that are reapplied after the upgrade. In our experience, upgrades also tend to cause port conflicts until changes are reapplied. Blackboard acknowledges that "the altering of template files is a known issue with the upgrade process" and states that this behavior "is to be analyzed as part of the installer/upgrader work for 2012". No specifics have been offered regarding either the analysis process itself or the expected timeline for actual remediation of issues identified during the analysis process.

- Blackboard upgrades frequently break third party Blackboard Building Blocks. As an example, we experienced significant issues with both the student photo roster and Echo360 Building Blocks. These issues are discussed further in section E: Value-added functionality. Blackboard acknowledges that "we have dropped the ball on ensuring forward compatibility of third party extensions relying on Building Block APIs" and suggests a planned remedial approach that includes “implementing improved instruments in Product Development as part of our application build processes”. Specific improvements have not been enumerated, and no timeline has been specified with respect to implementation.

C. CORE FUNCTIONALITY
The quality of built-in application features

Difficulties:

- Access control is inconsistent. We found several significant bugs with respect to guest access, as well as general access to the Blackboard Content Collection.

- The built-in Blackboard calendar is inadequate for our needs and represents a step down from current Stellar calendaring functionality. As an example, dated materials and assignments do not appear on the calendar—a serious user-facing shortcoming. This is a broad-based issue that would require significant development time to address.

- The built-in Blackboard gradebook is also inadequate for our needs and does not meet user expectations. Prior to any future experimentation with Blackboard 9.1, a more functional alternative to this component would need to be deployed.

- Permalinking (the ability to access a particular view via bookmarked URL) is not adequately supported—a significant issue given the requirement to redirect Stellar traffic for evaluation participants by linking their Stellar course listings to the appropriate Blackboard pages. The frames utilized by the application preclude bookmarking, and the URL of any given frame does not return the user to the same view. This is a significant usability issue. In addition, the URLs are based on internal unpredictable database ID numbers as opposed to external course IDs. As a result, there is no way to ascertain in advance what the URL for a particular course would be. Blackboard suggests using "deep linking" to address this issue, but this cannot be characterized as a straightforward or preferred approach. For context, the Stellar URL for the course ESD.40 for SP 2011 is:

  http://stellar.mit.edu/S/course/ESD/sp11/ESD.40

  while the Blackboard equivalent is:

  https://stellarng.mit.edu/webapps/portal/frameset.jsp?tab_tab_group_id=_2_1&url=/webapps/Blackboard/execute/courseMain?course_id=_3661_1

  Blackboard acknowledges that this issue is a confirmed common paint point among “a number of clients”. They add that “the need is under consideration for future releases, though no specific roadmap commitment exists yet”.

D. EXTENSIBILITY AND CUSTOMIZABILITY

The ability to customize and extend built-in functionality

General:

- Extensive user-level customization is possible via utilization of the system admin screens in the UI. Custom theming, permission filters, and the default course template itself can be largely managed without root access to the application server (although isolated bugs in the product do require root-level control in certain instances).

- Certain types of automation are straightforward. We developed a script to populate and synchronize the Blackboard courses and their memberships with data from our registration feed using Blackboard's built-in snapshot tool. This was a relatively simple solution that functions quite adequately.

Difficulties:

- Blackboard Custom Authentication support does not integrate seamlessly with the rest of the product. While it was least somewhat straightforward to implement the module for Touchstone authentication, the dependency on this module manifests itself in surprising ways. As an example, many of the product's configuration files must be updated to refer to the new authentication jar file, despite the file having nothing to do with authentication (for instance, the Blackboard snapshot tool runs on the application server and does not send any request over HTTP.) These dependencies cause certain seemingly unrelated parts of the product to fail, especially subsequent to upgrades. Blackboard has stated that this particular issue was addressed in Service Pack 8. However, this update was released in the closing weeks of the Spring 2011 semester, and required, according to Blackboard, a complete “re-code of [MIT’s] custom authentication module” in order to allow it to be tested against the new framework. Installation of SP8 would also have required a complete uninstall and reinstall of our other customizations, potentially introducing new points of failure within application behavior, as had been our experience with past Blackboard upgrades. Additionally, installing SP8 would not have resulted in any tangible benefit to users, as we already had in place a functioning Custom Authentication module that reconciled the idiosyncrasies nominally accounted for in SP8. With respect to our particular scenario, the primary potential benefit of SP8 would have been possible simplification of the update process for post-SP8 upgrades. Due to the timing of the release, the work involved in recoding our authentication module, the service disruption required to uninstall and reinstall all of our customizations, the risk of introducing new points of failure during the closing weeks of the semester, and the lack of tangible benefit to our users during the remainder of the experiment, SP8 was not installed, and the extent of the utility it could potentially have provided cannot empirically be assessed at this stage.

- The web UI relies heavily on XHTML frames that do not pass validation tests, and omit much semantic information. We developed a custom theme for Blackboard 9.0 to address this shortcoming, but it could not be used in Blackboard 9.1 because of a change in the Blackboard UI structure between versions. The theme could not identify menus by semantics (e.g., class name), and instead had to identify them based on heuristics (e.g., "the first <li> element which is a grandchild of a <div> element whose parent has a particular class name). Blackboard maintains that their approach is a “perfectly acceptable practice”, and states that they do not “find it realistic to give every single element in the DOM its own class”. While this position negates an overly zealous approach that we had not in fact advocated (associating a class with each individual DOM element), it does little to address the issues we identified, which could be remediated by a adopting a more rigorous semantic approach. In any event, we encountered several issues suggestive of the fact that theming behavior is not adequately modular within the application.

- We were expecting basic extensibility requirements to be addressable via limited development, but this was not the case. General-purpose functionality such as inclusion of a copyright warning screen or the ability to add class materials due dates to the course calendar cannot be readily accommodated without complicating the already difficult product upgrade path, and there is no way to add such extensibility in Blackboard 9.1 without compromising maintainability. According to George Kroner, our main Blackboard technical contact, there do exist solutions that involve using undocumented hooks into built-in functionality and duplicating certain Blackboard pages. This is unlikely to be a feasible or desirable approach for us. In any case, very little of our requirements are addressable via small-scale development within the BB 9.1 framework.

E. VALUE-ADDED FUNCTIONALITY
The quality of plug-ins and third-party Building Blocks or components

General:

The Blackboard Building Block model, which is intended to augment core functionality via an open-code approach relying on the development of application extensions, can be characterized as a largely unregulated space that may potentially hold future promise, but is currently sub-optimal. There is a broad heterogeneity in Building Block quality, compatibility, maintainability, upgradeability, and pricing. Accurate information on the breadth of available Building Blocks or disposition of certain projects is not readily available. For context, no meaningful comparison can be made between the Blackboard Building Block community and the Confluence plugin community, upon which we heavily rely for extending the usefulness and operational scope of our Wikis.

Difficulties:

- Many third-party Building Blocks are poorly categorized and difficult to find. As an example, we needed a student photo roster Building Block and could not find any in the Blackboard Building Block repository. We messaged other users in the Blackboard 9.1 mailing list, and managed to obtain relevant source code via this route. We invested significant effort in evaluating the code in order to determine its quality and applicability, followed by a comparable amount of time to properly configure and deploy it. This particular feature should be a very common component, yet it is neither in the core software nor readily available as an extension.

- When Building Blocks involve third-party software, it is in practice very difficult to determine where to obtain support for the component. We experienced several issues with the Echo360 Building Block, and had to navigate between support streams with both Echo360 and Blackboard technical support in order to identify workable solutions.

- Upgrades released by Blackboard often break Building Blocks. In our experience, even minor version releases of the product are not backward-compatible with much of the functionality used in a number of Building Blocks. We do not have access to information that assist help us in determining whether such incompatibility stems from a particular Building Block using functionality no longer supported in a particular Blackboard version release, or whether Blackboard is explicitly breaking its own APIs. In either case, incompatibility between the core product and product extensions renders the latter largely unreliable as a means of strategically and systematically extending core functionality in a well-planned and maintainable manner. Blackboard has stated that the planned launch of "new and exciting initiatives" will remedy this scenario in the near future. One such initiative is described as a "prototype 'compatibility checker' for Building Blocks", which according to Blackboard, will be designed to check for the compatibility of a given Building Block against a given release prior to the application of upgrades. However, there is a high likelihood that the utility of this prototypical service will be significantly hampered by the fact that Blackboard has been heretofore unable offer a complete listing of available Building Blocks and their functionality and/or disposition. The lack of such information would severely hamper the effectiveness of any compatibility-checking service.

CONCLUSION

Systemic issues associated with supporting and maintaining Blackboard 9.1, coupled with limitations on core functionality and extensibility, render the product less than suitable to MIT's current needs from a technical perspective. These shortcomings are exacerbated by the relative utility of the Blackboard Building Block model, which is not sufficiently mature to provide positive value to users seeking to easily extend the functionality of the core product in a programmatic and sustainable manner. Subsequent discussions with Blackboard suggest that while company is aware of the shortcomings of the Building Block business model—an initiative that it has heavily marketed as step toward open standards—it cannot articulate a sufficiently clear path toward remediating these issues. The primary points of failure in the Building Block model hinge on the apparent incompatibility of the application's current architectural model with a true open-standard or open-code approach, coupled with what appears to be a lack of familiarity with best practices for the management and growth of a vibrant open-code community. These aspects of the product, which amount to both functional and strategic liabilities, suggest strongly that Blackboard 9.1 is currently suboptimal as either an end-to-end closed-system solution or an extensible open-code framework for use by the MIT community.

Part II: COMMUNITY FEEDBACK

In May 2011, online surveys were sent to 33 MIT Blackboard course administrators and 618 MIT Blackboard student users. Two in-person focus groups were also held in order to gather additional feedback. The results of this exercise are discussed in what follows.

RESPONSE RATE

Admin surveys:	36%
Student surveys:	14%
Focus groups :	12%

1. RESPONDENT DEMOGRAPHICS

A. Administrators

Distribution:
31% Management (15.053, 15.0810, 15.840 and 15.220AB)
22% Engineering (ESD.40 and ESD.802)
13% Aeronautics and Astronautics (16.423J)
13% Physical Education (PE.0505.1, PE.0600.1 and PE0909.1)
11% Physics (8.251)
10% Literature (21L.007, 21L.006 and 21L.310)

Role:
38% Instructor
38% Teaching Assistant
08% Course Builder
15% Other (including departmental admins)

B. Students

Distribution:
53% Management (15.053, 15.0810, 15.840 and 15.220AB)
22% Engineering (ESD.40 and ESD.802)
11% Physics (8.251)
08% Literature (21L.007, 21L.006 and 21L.310)
03% Aeronautics and Astronautics (16.423J)
03% Physical Education (PE.0505.1, PE.0600.1 and PE0909.1)

Class standing:
55% Graduate
45% Undergraduate

The following sections present relevant information captured from course administrators and students via the survey and focus groups.

2. OVERALL IMPRESSIONS

The majority of administrators and students found **Stellar easier to use overall than Blackboard**.

A. Overall Preferences - Administrators

- 90% preferred Stellar
- 10% preferred Blackboard
- 0% had no preference

Sample comments:

"I liked tracking when students looked at the site, photo roster, ability to selectively release data to students"

"The main problem is not lack of features. It is usability."

"It would be better if there was less functionality, but the critical features were stronger."

"The numerous subcategories for functions often make it harder rather than easier to find what I need."

B. Overall Preferences - Students

- 68% preferred Stellar
- 13% preferred Blackboard
- 18% had no preference

Sample comments:

"I really prefer Stellar for its simplified layout and user interface."

"I know [Blackboard] can do a lot more, but I just need it to be good at being a document repository and homework submission space. If it were easier to do these things I would be delighted."

"On your grades page, you can see comments that your instructor posted about that grade. But if you click 'view more' it gets all screwed up."

" It was harder to locate information based on class dates compared to classic Stellar."

3. FEEDBACK ON PRIMARY BLACKBOARD FEATURE CATEGORIES

A. Content and Materials Management

- 97% of administrators found the Blackboard Content Collection difficult to use
- 92% of administrators found content management to be easier in Stellar
- 87% of administrators bypassed the Blackboard Content Collection, and managed content manually
- 57% of students preferred the content access and submission process in Stellar to that of Blackboard

Sample administrator comments:

"Content Collection is slow, and it is time consuming to import it to materials."

"I might say that that 'materials management' was worse [in Blackboard than in Stellar] because I never figured out the Content Collection and was never sure I was doing the best thing with my materials."

"Uploading materials took longer….I couldn't connect solutions to [problem sets] as in current Stellar."

Sample student comments:

"I really like the idea of having upcoming and past due assignments clearly visible on the front page, WITH associated due dates. They almost were in [Blackboard], but hidden behind the little pull down menus."

"Easier to find assignments and due dates in classic stellar. Generally easier to find documents"

B. Grading

Students displayed no strong preference between accessing their grades in the Stellar Gradebook versus the Blackboard Gradebook.   However, the majority of administrators were dissatisfied with grade management in Blackboard.

- 91% of administrators preferred the Stellar Gradebook to the Blackboard Gradebook tool
- 62% of administrators described the Blackboard gradebook tool as difficult to configure and use
- 64% of administrators considered a more robust and easier to configure gradebook tool to be important to their needs

Sample administrator comments:

"Using this gradebook was hard so we used an excel sheet instead."

"There's not a clear way to track make-ups (it's in the grade history but this is very cumbersome in the way it's formatted)."

"This is probably my biggest problem [with Blackboard]."

Sample student comments:

"Not much difference [between Blackboard and Stellar Gradebooks], really…"

"Classic Stellar was a lot more clear. "

"The grade display was better than classic Stellar, since team assignment grades were shared among all members."

C. Membership Management

Membership management is a feature that is used exclusively by course administrators.

- 67% of administrators described the management of participant/staff lists as difficult
- 46% of administrators chose to use the Blackboard Groups tool to organize students
- 77% of Groups tool administrators reported that the tool did not satisfy their needs

Sample comments:

"In one of our courses, I tried to set up [membership] groups but wasn't able to set it so students could switch to another group. Because of this that course had to be switched back to current Stellar."

"Tried to create 5 (sections) under groups so that students could switch between (sections). Couldn't figure out."

"Group lists/student names are alpha by first name only…[Can't] sort users by 'role'"

D. Course modules

Blackboard course support modules received mixed reviews, but approval rates were unimpressive overall:

- Announcements: considered useful by 62% of admins and 55% of students
- Calendar: considered useful by 23% of admins and 37% of students
- Journals and blogs: considered useful by 38% of admins and 31% of students
- What's New alerts: considered useful by 24% of admins and 37% of students
- Needs Attention alerts: considered useful by 18% of admins and 34% of students

Sample administrator comments*:*

"I liked the visual layout of the home page with access to different functions (I kept track of forum entries, for example). I appreciated that there were many new functions for me to try."

"There are many calendar functionalities from Stellar that haven't been integrated [in Blackboard]."

"I didn't use these features except for announcements."

Sample student comments:

"The blogs were great. I liked reading other student's responses to books."

"Only thing I really used was the announcements."

"Too much redundancy of information."

CONCLUSION

The majority of users found Blackboard more difficult to use and administer than Stellar. Users reached this conclusion in spite of the fact that the version of Blackboard 9.1 implemented at MIT was heavily customized to follow a workflow logic closely paralleling that of Stellar. In addition to following and complementing Stellar workflow as closely as possible, the application UI was also customized to present a look and feel that was consistent with the production Stellar environment. User dissatisfaction appeared to be less a function of the dissonance commonly associated with product migration experiences and more the result of key functional and usability shortcomings associated with the core product. The user feedback collected adds an additional and significant dimension to the argument that Blackboard would not be an optimal LMS choice for MIT at the current time.

Part III: RECOMMENDATION

Based on the information presented in the preceding sections, the recommendation at this time is to halt further experimentation with the Blackboard platform and shift resources to the continuing development of the Modular Service Framework, which is intended to gradually replace existing Sakai-2 based Stellar functionality with a set of discrete, flexible web services driven by a common data framework and based on a standardized set of APIs.

As existing core Stellar services are functionally accounted for with the Modular Service Framework, focus can increasingly shift to the integration of value-added functionality satisfying specific unmet or emerging user needs. Such functionality may be developed in-house or, where adequately represented by existing compatible third-party functionality, incorporated within the framework. This focus on flexibility and integration positions the model well for MIT's future needs, including the technological evolution mandated by Digital MIT, as well as emerging trends in curriculum development and online education.

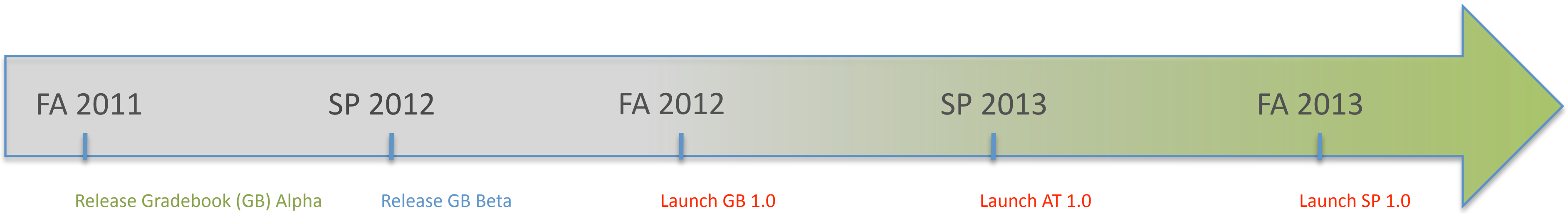
In this model, key functional components are represented by individual web services that can be utilized as either standalone modules or as part of an integrated set of user tools. These web services are driven by common core data sets, and share common standardized APIs. This aspect of the model eases integration and interoperability with community-developed and third-party tools, as long as such tools are compatible with the published API model. Such an approach encourages community innovation while balancing individual customizability and extensibility with service standardization and the reduction of support overhead.

This approach also facilitates a gradual and user-friendly approach to upgrading legacy Stellar functionality by separating core functional components from the architectural core while maintaining interoperability between components. In an optimal scenario, the Stellar core would over time be simplified to a document and material management framework extended by an interoperating set of web services sharing common APIs. These services would be recombined as dictated by the instant pedagogical and/or administrative scenario, and presented to the user within a simple and customizable module-based framework.

Current planning projects a 48-month delivery trajectory to the initial core components of the Modular Service Framework. This core set of components includes grading, attendance, calendaring, content and material management, forum integration, and blog/wiki integration. These represent existing functionality currently delivered within the Sakai-2 based Stellar framework. Additional components would include value-added functionality identified and prioritized via a community requirement gathering process.

The infographic on the following page represents the Modular Service Framework delivery trajectory through FA 2013.

Modular Service Framework Progression Through FA 2013



FA 2011	SP 2012	FA 2012	SP 2013	FA 2013
Release Gradebook (GB) Alpha	Release GB Beta	Launch GB 1.0	Launch AT 1.0	Launch SP 1.0
Gather GB Alpha feedback	Process GB Alpha feedback	Release AT Beta	Release Student Photos (SP) 0.5	Release CT Alpha
Begin dev on Attendance Tool (AT)	Gather GB Beta feedback	Process AT Alpha feedback	Gather SP 0.5 feedback	Gather CT Alpha feedback
	Release AT Alpha	Gather AT Beta feedback	Begin dev on Calendering tool (CT)	
	Gather AT Alpha feedback			

- Initial planned functionality includes grading, attendance, student photos, calendering, course materials, and images
- Targeting 1 pre-release and 1 full release per semester starting FA 2012
- Pre-release feedback gathered at end of term, reflected in successive releases
- Redundant Stellar functionality decommissioned step-by-step
- Specific service progression subject to change