



Open Source 2010

Reflections on 2007

By Brad Wheeler

Much has happened since the 2004 prognostications of my *EDUCAUSE Review* article “Open Source 2007: How Did *This* Happen?”¹ The article peered into the future through the lens of two possible outcomes for open-source *application* software by 2007. In the first scenario, higher education and commercial firms overcame many challenges to evolve a new “community source” model for developing and sustaining enterprise-scale, quality software. In the second scenario, the obstacles for collaboration and coordination of investments were simply too great: institutions could not find ways to agree. The article thesis asserted that the actual outcome for 2007 would reflect the collective actions of colleges and universities. For those of us in higher education, it was *our* outcome to choose...

Brad Wheeler is Chief Information Officer for Indiana University and Dean of IT for IU-Bloomington. He is also an associate professor of information systems in IU's Kelley School of Business.

Open Source Past and Present

So, what did we choose, and where are we now? First, a major shift occurred in the conversation about open source. The journey began with reasoned skepticism that higher education could or even should collaborate in developing software, in light of commercial offerings.² Among open-source skeptics were those intent on defeating the fallacy that “open source is free,” which was never a substantive topic regarding the real issues involved in open source. Nevertheless, software-development efforts were started, many of them seeded with matching startup funds from the Andrew W. Mellon and William and Flora Hewlett Foundations. The software work-products were made freely available to all in the hope that a community would form to sustain and evolve the software over time—and that is now happening. The Sakai Project has transitioned from a two-year, grant-funded project into a 120-member independent foundation and worldwide community. Colleges, universities, and commercial firms are pouring over \$1 million annually into the Sakai Foundation and are contributing much greater resources in staff time through collaborative community work. uPortal has hundreds of implementations,

there has been no mass exodus from commercial application software by 2007, there is a growing base of adoption of open-source applications among a broad array of institutions. Thus, the conversation today has shifted. The question now concerns what adopters must do to ensure sustainability for this community-owned software.

Second, the commercial world is adapting, as good capitalists always do. Oracle has become a Sakai Commercial Affiliate, alongside thirteen other small and large firms that are providing consulting and support for open source. rSmart is offering a “Kuali Appliance” of a server with a pre-installed open-source stack and Kuali Financials ready for institutional configuration. The commercial embrace of open source mirrors trends for other industries in the business world and also makes clear the complementary nature of commercial capabilities alongside open source. The rSmart offering is just one example of a fee-for-service in which a commercial firm reduces the complexity of an open-source implementation.

Third, open-source projects are consolidating to take advantage of natural efficiencies. The Open Source Portfolio Initiative has merged its administration

In my assessment, the *pace* predicted in the 2004 article was overly optimistic. Many of the *trajectories* proffered for collaborative software development, however, were right. Higher education has learned to develop, maintain, and improve quality applications that scale for large deployments. Colleges and universities have a clear template for how to pool investments to execute a community-source project, and they have learned how to develop, fund, and scale a global software community with broad commercial support.

An Answer to Four Challenges


If we assume that higher education institutions will continue to refine their abilities to develop and sustain software together, the fair question remains: Why? I see four challenging trends that are motivating the collective interest in open source. The first two speak to the demand for and funding of IT services, and the second two address problems with the software market for higher education.

Greater Demand and Modest Resources

Every administrator faces the challenges of balancing demand and supply, but nowhere is that task more acute than in the provision of IT to a rapidly changing college and university community. The evolution and revolution of “Scholarship 2.0” continues the digitization of academic processes for research, teaching and learning, service, and the critical social elements of scholarly communities.³

Unbridled Demand for IT Services

Students, faculty, staff, and other academic stakeholders are all contributing to an unbridled demand for IT services. For example, global disciplinary research communities, which collaborate via networks and scholarly repositories, are becoming voracious users of storage, networks, and computational resources. Various reports outline the need for local and national investments in cyber-infrastructure for modern scholarship across the sciences, humanities, and arts.⁴ Research grants often proffer the long-



Although there has been no mass exodus from commercial application software by 2007, there is a growing base of adoption of open-source applications among a broad array of institutions.

with more in process and some migrations from commercial portals (the University of Wisconsin). Only halfway through its thirty-month startup grant, the Kuali Foundation is adding members and attracting hundreds to its conferences. Kuali Financial System 1.0 was released as an enterprise-scale administrative system, and Kuali Research Administration is in development. Moodle shows growing adoptions worldwide, and VUE, Fedora, DSpace, and the Open Source Portfolio (OSP) all continue to develop their software and growing communities. Although

into the community of the Sakai Foundation. Yale’s Common Authentication Service (CAS) is now part of JASIG and is also gaining popularity in the commercial sector. MIT’s Coeus system for awards management is looking to a software future in collaboration with the Kuali Foundation. These moves are early examples of projects being freed to concentrate on their core interests of developing software while efficiently leveraging back-office support for such shared services as community collaboration, tools, release packaging, and conferences.

term preservation and curation of digital artifacts and the provisioning of network access to those resources.

In the area of teaching and learning, the demands for course management systems, digital libraries, e-portfolios, distributed education, and new tools for effective learning continue to push the frontiers for IT services. Storage and network demands for courses and repositories continue to grow. Most administrative systems are extending their self-service capabilities to 24x7 support, and users expect integration among systems with personalized views of their data.

Our stakeholders are also changing. The pages of *EDUCAUSE Review* and other publications have chronicled the rise of “The Millennials”—today’s digital-ready students—and their expectations for college and university IT services. With public services offering free, practically unlimited storage for e-mail, videos, and other services, there is a perception that campus IT should enable the

same or that institutional offerings are irrelevant. Thus, the benchmark for colleges and universities’ IT services is perceptually assessed against what Google did yesterday. Of course, such free or advertiser-subsidized services offer little in quality-of-service agreements, policy enforcement with college/university rules, and assured longevity. For example, Google and other commercial providers can advertise four gigabytes (GBs) of free storage and set users’ expectations for IT capacities. Of course, they grossly oversubscribe this offering, since very few users take advantage of the full four GBs, and thus the commercial providers invest in only a fraction of what would be required for every Google account. In college and university environments, however, provisioning four GBs per user will, in time, produce the very real and full cost of provisioning, supporting, and backing almost four GBs per account.

Although some of these service demands offer few options for control, application software is one domain

where colleges and universities can exert choices. Unlike back-office networks or data centers, higher education application software is among the most visible user experience with campus IT. Users’ perceived satisfaction with course management, content management, research grants administration, and student services shades their perceptions of IT service quality. The ability to rapidly provision application software and shape these systems to specific local needs in an era of rising expectations is an assumed competency for campus IT. No dean wants to hear that he or she cannot implement a new distance-education offering because the administrative software is not able to enroll and bill students for a particular degree program innovation.

Modest IT Resource Growth

A second trend reveals that the resources enabling the digital college or university will be unable to keep pace with the rapidly growing demand for IT services. There is rising pressure to contain the

cost of higher education and to leave more money in the treasury for academic pursuits rather than for “overhead” expenditures.

For example, the University of Missouri System has announced a plan to repurpose \$20 million in administrative expenses back to academic programs: “Under the plan, 92 full-time-equivalent positions would be eliminated across the system. . . . The reductions also include moves to reduce credit-card fees, slash funds for some administrative departments, and pass the full cost of information technology and other services on to other entities at the university.”⁵ The shell game of “hide or pass the IT cost” is unlikely to be efficacious over time, but the trend is clear. The academic community has rightly lost its appetite for multimillion-dollar administrative systems and their insatiable upgrade costs and disruption.

IT leaders must be good stewards to seek out IT efficiencies and to support, in every way, the academic endeavor of the college or university. Demand-management strategies and rationing, where appropriate, may offer one option for cost containment, but such tactics will

and began with the presumption that a competitive commercial marketplace should mostly obviate the need for home-grown or pooled software development. Counter to this presumption, however, they found considerable dissatisfaction with the cost, performance (fit to needs), and control of software. They also found numerous academic efforts outside of the commercial marketplace. In assessing why, Courant and Griffiths proffered two major observations, both addressed by what they called “directed open source” development.

A Marketplace Failure

The first observation—our third trend—is that the size and structure of higher education as an industry is not conducive to sustaining a robustly competitive market:

Higher education, however, is different in a variety of ways. First, it is small relative to other large sectors of the economy, which may lead enterprise resources planning (ERP) vendors to produce to a broad marketplace that is quite different from higher education. The relatively small size of higher education may


while being deeply puzzling to the corporate world.⁷

This insight is particularly telling when one considers the last fifteen years across many types of software in higher education. For ERP applications, library systems, and course management, for example, the pattern is clear. After a period of intense early competition, there is considerable commercial consolidation—if not near monopolization—of each software category. This is economically efficient for the firms that serve the higher education industry, yet it leaves colleges and universities with few options when the values of software owners and users diverge. Price increases, forced upgrades, and/or dropped support for current products are sometimes imposed by rational commercial interests pursuing their values.

The Shortest Distance

Finally, effective software must meet the needs of its users, and some interviewees expressed concerns that large vended systems give institutions too little flexibility to adapt systems for specific needs. This problem is best understood as a growing distance between the users and the software developers. For home-grown systems, this distance can be very short, but it may invoke a very large system lifecycle cost over the years. The commercial model of software development often adds many layers of staff and organizations between users and developers, thus creating greater distance.

Courant and Griffith observed that open-source development provides the shortest distance between a software user and a software developer. Not all institutions choose to write software, and the vast majority of colleges and universities are software-consuming institutions rather than software-producing institutions—though producers are rabid consumers too. Nevertheless, even the few hundred institutions that employ permanent development staff have considerable and direct sway in modifying and improving the behavior of any open-source software for the benefit of all. Consuming



The vast majority of colleges and universities are software-consuming institutions rather than software-producing institutions—though producers are rabid consumers too.

face increased scrutiny in light of users’ personal experiences with online services in other parts of their lives.

Software for Higher Education

A recent study by Dr. Paul Courant, professor of economics and former provost at the University of Michigan, and Rebecca Griffiths reveals the third and fourth challenges.⁶ The work was commissioned and funded by seven academic institutions, with added support from two foundations. Courant and Griffiths conducted in-depth interviews with sixty-six stakeholders across higher education

also make it especially vulnerable to monopolization. Whereas one vendor may find HE [higher education] to be profitable, there may not be enough of a market to stimulate the entry that is so essential to effective competition. (This is especially troubling in light of the high initial investment and switching costs imposed on customers and the relatively low number of competing vendors in the software industry.) Second, higher education really is idiosyncratic and has many business practices that are unique and essential to the sector

institutions can also easily rent development skills from the commercial sector for specific needs.

There are also many ways to contribute to great software other than by programming. Developers are of little use without clarity of requirements for what users actually need. For example, in open-source projects, a faculty member or instructional consultant at Texas State University can directly participate in the worldwide, public requirements discussion for the Resources Tool user interface in Sakai. There is real-time visibility of mock-ups, plans, and testing environments, which are available to anyone. Thus, when an e-mail requesting a feature or system change is sent to a developer list, it has a very real probability of influencing a near-term system change. In some cases, this even leads to a rapid-fire public dialogue that quickly moves through several design alternatives and reaches a community decision as the software is still being written. Experience demonstrates that end users actually *do* engage in these community activities. Open community discussions among developers, users, support staff, and others are simply not the common norm in the commercial marketplace.

Building Together

Collectively, these four challenges—rising demand, modest resource growth, marketplace dominance, and a growing distance between users and software developers—beg for new approaches for IT services, including application software. To the extent that the open-source model for developing and maintaining software mitigates some of these concerns, it merits the full scrutiny of academic leaders.

Open source's greatest appeal is the leveraging of resources of the partners and the community for shared value

creation. For example, in the past, when Cornell University spent \$500,000 for some system, the investment provided no advantage to San Joaquin Delta College. Or when Rutgers University developed a clever piece of cross-language middleware code, Indiana University did not benefit. When Cambridge University developed a teaching tool, U.S. and other institutions gained little. The open-source model changes all of this: it provides a real tool to solve the “do more with less” challenge facing higher education. To put it bluntly, all of us in open source are mutually using other people's money to get and sustain the systems we need. Open-source communities are foremost a coordination mechanism for institutional, corporate, and personal investments of resources, ideas, and talent.

A strong belief in such leverage and collaboration is one of the motivations for the Andrew W. Mellon Foundation's millions of dollars of investments in open-source software for higher education. Ira H. Fuchs, vice president for research in information technology at the Mellon Foundation, observed: “There are good and fitting reasons that we support OSS [open-source software] efforts . . . in general it is because these projects have the potential to solve problems facing our constituencies and we believe that a collaborative approach to solving these problems with an open source license has the highest probability of benefiting the most people and organizations.”⁸

Still, what do we really know about the open-source model for application software and its sustainability over the years?

Insights Since 2004

Indiana University chose the open-source, build-together approach as part of its software application strategy in 2003

(see sidebar on p. 56). Our broad engagement in leading, participating in, and consuming open-source software across many domains provides some insight regarding the progress and perils of open source in recent years. This section offers three reflections on that experience, as well as three essential lessons learned.

Reflections

Reflection #1: The Community Source Model for developing and sustaining software is a remarkable fit to the culture and values of higher education. The academy draws its millennia-refined values from the research, teaching, and service roles of the faculty. Administrators can add efficiencies, resource stewardship, and an ethos of service in the provision of IT services, but the core values of higher education are steeped in discovery, knowledge sharing, and scholarly communities. Thus, the behaviors of staff in open-source software communities align with the subtle but pervasive values of the academy. In open-source software, communities draw on the leverage of a shared core system without constraining the option for local add-ons to meet specific needs.

Although the Community Source Model—directed open source for higher education—remains in its infancy (see sidebar on p. 58), an unexpected benefit has been the remarkable sharing and staff development across the communities. Institutions have freely shared training materials, support documents, tutorials, installation configurations, and much more. The communities have created a near frictionless means of sharing and repurposing these valuable materials across institutions. Likewise, the communities and project conferences have been an unexpected boon in staff development. Working in open-source communities is a positive for staff retention.

Reflection #2: The unbundling of software and support is efficient. Software—especially mission-critical software—requires support to update it for security matters, integrate it with other software and data sources, fit it to local system configurations, and improve it as new features



Open-source communities are foremost a coordination mechanism for institutional, corporate, and personal investments of resources, ideas, and talent.

become available. The commercial model of vended software bundles a license to use the software with few options for support. Most often, only the vendor has access to the source code to modify and distribute changes; thus, it has a monopoly on pricing support services, though some support activities may be provided by third parties.

The open-source model unbundles ownership of the intellectual property (the software) and support for it.⁹ Thus, any college, university, commercial entity,

or person can download, without fee, the Fedora or DSpace software with full rights to use, modify, and redistribute it. Installation, integration with other systems (e.g., the registrar or library system), and local customization (e.g., skins, logos, URLs) require some support expertise. Institutions may employ permanent staff for these matters, rent staff via consulting engagements, or even choose to buy a commercial Sakai package that includes support options (i.e., rSmart Sakai CLE or Unicon's Sakai Pilot Program).

No matter which path is chosen, a college or university retains rights to the software once it is installed; these rights are retained via the implied perpetual use in open source. No company can say to stop using the software if new fees are not paid. If the support from a company or consultant is poor, then an institution can change to another company for support without *affecting its installation* (Sakai has fourteen commercial affiliates, with competitive offerings). Predictably, this model brings new efficiencies to the market-based pricing of competitive support options.

Another efficiency of the unbundling is the remarkable ability of the community to support itself. The effect of community support is best understood through the book *The Wisdom of Crowds*.¹⁰ Wise crowds include a diversity of opinion, independence, decentralization, and a means for aggregation. Community-source projects directly connect thousands of staff—from community colleges, from research universities, and on multiple continents—who are supporting the software in a variety of technical environments under differing local imperatives. E-mail lists, wikis, and friends provide remarkably efficient and timely access to deep technical knowledge, software roadmaps, and insights that are almost impossible to obtain via contractual services.

Reflection #3: Enlightened self-interest is the glue. My colleague John Norman, at the University of Cambridge, first observed that Community Source Model projects are ultimately held together by *enlightened self-interest*. I think this market-based behavior works extremely well for higher education: it is a very efficient means of allocating resources to real needs. For example, the University of South Africa was reevaluating its entire technology-support strategy for a merger of two larger universities and chose to focus its resources on adding locally developed tools to the Sakai software rather than integrating its own homegrown systems. Some institutions that are fine with their current commercial course management system implementation are actively involved as paying members of the Sakai Foundation for the value of the community beyond the open-source software.

Open-Source Investments at Indiana University: A Firsthand View

Indiana University views the open-source, build-together model (the Community Source Model) as one tool for sourcing our software needs. We continue to buy much commercial software, and when necessary, we still write some software by ourselves. Our early and growing experiences with the Community Source Model have exposed us to the full range—great, good, bad, and ugly—of learning how to work in communities. Our multi-year investments of staff time tendered to projects include the following:

2004–2005	\$ 1,000,000	Sakai Project
2004–2006	\$ 500,000	Open Source Portfolio
2005–2007	\$ 2,000,000	Kuali Financial System
2006–2007	\$ 87,000	Sakaibrary (digital library)
2006–2008	\$ 430,000	Kuali Research Administration

Although these numbers may look large, they consist mostly of existing developer and functional specialists staff salaries that were assigned to the projects. They now represent investments in leverage with other institutions for a path of shared development. We have participated in almost \$20 million of pooled university investments and have received about \$4.5 million in grants to Indiana University (some subcontracted to partners).

We have perpetual-use rights to all of the software code—as would any non-investor—but we've been able to help shape the software to accommodate our requirements and processes. These investments and the accumulated experience of our staff essentially “buy down” the implementation and maintenance costs, since we know this software very well. And if we have problems, we also know the members of the community very well and can always get a little help from our friends.

The Community Source Model

The Community Source Model is a hybrid model that blends elements of directed development, in the classic sense of an organization employing staff and resources to work on a project, and the openness of traditional open-source projects like Apache. The resulting software is available under an Open Source Initiative (OSI) approved license. The code can be examined, changed, redistributed, sold, or incorporated into other products without fee. Anyone can make changes, and subject to quality review, those changes can be incorporated back into an open-source application for the benefit of all.

The distinguishing feature of the Community Source Model is that many of the investments of developers' time, design, and project governance come from institutional contributions by colleges, universities, and some commercial firms rather than from individuals. These contributions may be tendered as the first phase of a project, and then additional work may be contributed on an ongoing, voluntary basis by those institutions with a continuing interest in the project. The project often establishes a software framework and baseline functionality, and then the community develops additional features as needed over time.

Community Source Model projects generally operate as follows. Several institutions realize they are trying to solve a similar problem—need for a research administration system is a recent example. After some discussions and resulting agreement on project objectives, timelines, and philosophy, the institutions pool their resources under a project board of institutional leaders. The institutions are often agreeing to tender existing staff time to the direction of the project, and as such, this is not a new cash outlay but rather an aggregation of existing staff in a virtual organization. A grant from a foundation may provide cohesion among the investors. Typical recent projects have ranged from \$1 to \$8 million in funding and from twelve to thirty months in duration. Each investor signs a Corporate Contributor Agreement that grants a copyright license for the software to the project or foundation (modeled on the practice of the Apache Foundation). The project usually operates on a date-driven delivery schedule. This forces difficult decisions in the reality triangle of balancing features, resources, and time, but such a schedule is essential to the growth of community confidence.

The project board then establishes the appropriate structure for articulating the system requirements, the technical choices, and a project manager. It is essential that clear roles and responsibilities be established early, and the project participants will benefit from spending some face-to-face time together at the beginning of the project. Experience reveals that some staff members may not work well in distributed, virtual organizations, whereas others find the work to be career-renewing.

Early projects had to transition from an investor-based project to a community and a foundation. New projects can take advantage of the foundations' existing infrastructure and how-to knowledge and can begin as a project of a foundation. There is no rulebook for Community Source Model projects for every domain, but there is a growing body of accumulated wisdom on how to coordinate institutional investments and execute a development plan for quality software.

Rutgers University has written, and contributed to Sakai, some valuable code for cross-language Web services support because the university needed the code locally. Foothill College has developed and contributed its Melete lessons tool because the tool was essential for the institution's faculty and was missing in Sakai's early versions.

Likewise, openness is proving to be a strong attraction even within large organizations, where cooperation by administrative fiat would not be effective. The decentralized structure of the University of California (UC) has made cross-campus IT investments challenging. The openness of projects at the Kuali Foundation and Sakai is encouraging collaboration among UC institutions with the support of the UC Office of the President (UCoP) and the CIO. Three campuses and UCoP recently joined to invest \$1 million in the Kuali Financial System development. Cornell University and Cornell's Weil College of Medicine have joined their separate research administration system efforts as investors in the Kuali Research Administration.

Lessons

Lesson #1: There is a core model for how to coordinate, scale, and sustain collaborative development for higher education. Few know that what has now become the Kuali Financial System had an aborted start as a closed collaborative development among some members of the Committee on Institutional Cooperation (CIC) in 2002. The effort at that time would have been organized as a gated community held together by contract with specific commercial arrangements. The October 2006 release of the open-source Kuali Financial System—with investments from public, private, research, and community college institutions—is a better system and has a better future, with the backing of the independent Kuali Foundation and multiple commercial support providers.

The community has now explored some alternative collaboration models, with varied insights. MIT and members of the Coeus consortium for research administration software realize that the model of "MIT as vendor" is less than ideal for many. The initial, pure

open-source approach of “if we build it, they will come” of the OSPI (Open Source Portfolio Initiative) did not work. DSpace has seen widespread adoption of its software in many countries, but it continues to look for more contributors to improve the software as active members of the community. Experiences from each of the open-source projects provide insight for what works well for higher education. Although there is considerable heterogeneity among the teaching and learning, research, administrative, library, and infrastructure staff and needs at institutions and their respective communities, a core

important, however, than the choice of software license, which defines much of how a community will work together and will interact with other communities.¹¹

In the United States, forming an independent, tax-exempt 501C3 foundation is proving useful for project sustainability. Following in the example of the Apache Software Foundation, this formation provides a legal home to hold the copyright for contributions, to collect contributions and/or membership dues, and to coordinate community activities. The practice is to purposefully employ very few staff at the foundation to provide for the

was frustrated that the Sakai system, as implemented at Indiana, supported only official university e-mail addresses. Within two weeks, the development team modified the software, tested it on the staging server, deployed it into full production during a maintenance window, and committed the change back to be part of Sakai’s next, 2.3 release. The faculty member and the development team were delighted, and the entire community benefited from the work. This type of rapid improvement plays out over and over as dozens of institutions and developers learn to efficiently contribute their work.

Nevertheless, some decisions are mutually exclusive. Some users may argue strongly for a certain development path in the framework while others argue differently. At some point after much discussion, decisions must be made and adhered to for community leverage. A respected, benevolent dictator is essential when those

decisions must be made and consensus is not possible. A strong project board and good local managers are essential to ensure that decisions stick and that the community moves forward.


Open Source 2010

Looking ahead, what can higher education expect from open-source application software in 2010? What institutional changes, if any, should leaders consider for open source 2010?

Looking Ahead

First, all trends point to the continued mainstreaming of open-source application software in the commercial and academic markets.¹² The full lifecycle economics of system development and maintenance are favorable to this model; thus, an open-source course management or financial system will seem about as unusual as someone using Linux in a machine room today.

Second, the software foundations—such as Sakai, Quali, and others—will emerge as hubs of activity for coordinating institutional investments. They will grow in their competencies for managing all parts of the software lifecycle and



Looking ahead, what can higher education expect from open-source application software in 2010? What institutional changes, if any, should leaders consider for open source 2010?

model can coordinate resources, sustain software, and enable multi-institutional communities with commercial support. These lessons—pioneered for higher education by uPortal, refined by Sakai, and further developed by Quali—are briefly outlined in the sidebar about the Community Source Model.

Lesson #2: Community building is more than half the work. Developing enterprise-quality software alone is hard, and developing it with the help of a distributed community is really, really hard. Building a sustainable community to support it is even harder, but doing so is essential for any real confidence of adoption and of access to the motivating economics of the endeavor. Thus, community-building efforts must be engaged very early in a software project and should be a part of the project’s early budget.

Communities come together through collaboration infrastructure (e.g., e-mail lists, wikis, Web sites) and conferences. There are many decisions to be made regarding technical choices, release packaging, quality-assurance work, and communications, and someone or some group needs the authority to make these decisions. Perhaps no decision is more

coordination of member activities and some core responsibilities (e.g., security, framework, release packaging, member communications) and draw on development among the members (remember the “shortest distance” trend). Membership with equal rights for educational, commercial, or other not-for-profit entities also facilitates a frictionless combination of the best resources and talent to address the needs of the community.

Lesson #3: Institutional engagement is challenging and valuable. Early adopters and pioneers of community-source applications experienced considerable challenges and rewards. Major investors and early adopters—such as Foothill College, Indiana University, and the University of Michigan—have lived through the challenges of early-release software and the “forming and storming” stages of community development. Some staff members burned out under the pace, whereas others were exhilarated by the experience. Users are finding both joy with some benefits of the software capabilities and frustration with some things they wish would work differently.

One example comes to mind. A faculty member at Indiana University

servicing the needs of their members, and this will provide ready-made infrastructure for new projects. The foundations may have similar back-office operations, but their front-office, member-facing activities will be tailored to the needs of their respective communities. How many foundations are needed? Should each project create its own legal home? I believe that higher education would likely benefit from about four foundations, which could be umbrella organizations for projects to distinct communities:

- Teaching and Research: Sakai
- Administrative Systems: Quali
- Infrastructure: JASIG
- Scholarly Repositories/Libraries: ??

It is also possible that a meta-foundation might provide leveraged, shared services in hosting community communication services, software testing, integration work, release packaging, and legal services. By 2010, the maturity of these operations in the foundations may make

them timely for consolidation among a shared-services unit.

Third, commercial support offerings for open-source software will thrive alongside sharpened value propositions for proprietary software. When freed from the enormous costs of developing, maintaining, and marketing proprietary software, more firms will emerge with specialized and valuable expertise around open-source application software. A shifting of these software-development and -maintenance costs from a firm to the community will (partially) correct the marketplace failure of recurring monopolization.

Fourth, open-source applications will put colleges and universities back on a path of zero disruptive upgrades.¹³ Home-grown systems enabled ongoing and modest patching and improvement by local developers, but the monolithic ERP systems used by many institutions today have imposed expensive and disruptive upgrade cycles, with few alternatives. Open-source applications will restore

control of software evolution to colleges and universities and will force vended packages to realign offerings with the needs of the academy. This is a good outcome for everyone in higher education, no matter their software choice.

Finally, experiential differences will be apparent among institutions that have incorporated community-source engagement into their IT strategy. This will be true for both software-producing and software-consuming institutions. There is a learning curve for working effectively in open-source communities, and in a meritocracy environment, it takes time to establish credibility to lead through influence. Contributions of effort—often made in community activities other than writing software—lead to respect and being known in a community. Institutions and staff members who are known engender greater kindness from strangers when they seek help on a list or post a request. Virtual software communities need and value many skills, at institutions of all sizes. Thus, time invested in giving to

software communities, refining institutional and personal skills to leverage the work of others, and discerning community trends has a real institutional payoff.

Risks and Impediments

Risks remain and can impede community development for 2010. The greatest risk is the ever-present option to fork the code. “Forking” means that someone takes the software in a different direction apart from the community. This could be a commercial endeavor to fork and package the software in order to sell proprietary value, or it could be an effort by members of the community who no longer want to work together. Whatever the reason, a fork may splinter the leverage that is the economic basis of community support. A healthy community should find ways to avoid the incentives to fork the code. If large parts of the community follow various forks, however, then this is the marketplace speaking. The solution, of course, is for members of higher education to collectively force healthy com-

munities to meet their needs and to not reward or motivate efforts to fork.

Second, in the absence of meaningful software patent reform in the United States, 2010 will continue to be plagued by the chores of litigating and defeating overly broad and ill-advised patents. The unsavory aspects of questionable patents exploded onto the scene with Blackboard’s 2006 lawsuit. Blackboard is not alone in holding patents of interest to the core software operations of higher education, but its lawsuit opened a torrent of community protest and renewed concerns of how patents would affect collaborative software development within higher education. Even though many patent claims will not stand the tests of prior art, they will remain an unhealthy friction on the use of software in higher education. Again, colleges and universities have the marketplace power to not reward those who impose such friction, and perhaps early lessons can dissuade this behavior. IBM and other firms have demonstrated how they can obtain patents for defensive purposes and

still place the patents in a Patents Common for use in education.

The final risk is depriving open-source communities of the resources and engagement they need to flourish as efficient alternatives to homegrown and commercial software. Institutions that lament the absence of options in, or the vendor pricing power of, commercial software companies but that do not invest in shared communities fall prey to Steven Kerr’s classic “On the Folly of Rewarding A, While Hoping for B.”¹⁴

The Collaborative Capability

Open-source software—through directly leveraging the investments of others—is a potential remedy for the four challenges noted above: rising IT services demand; modest-to-declining resources; a marketplace failure of true alternatives; and the growing distance between developers and users to shape software. Yet to be precise, it is the institutions that develop the *collaborative capability* that will extract the greatest value from open source 2010.

Institutional capabilities represent bundles of reliable routines that enable an organization to do something well over time. For example, Procter & Gamble has a unique capability to manage consumer brands and products, whereas UPS has an outstanding capability to manage logistics for shipping. Colleges and universities often have refined capabilities to contract and procure software or perhaps to stage and test new systems before roll-out.

I believe that developing deep collaborative capabilities among staff and as part of institutional routines will be well rewarded by 2010. The routines will enable staff to freely participate in and lead open-source communities. Institutions that are known to be good partners will be given the best opportunities to partner on grants and leading projects with other good partners. They will develop informal networks for making the best use of shared knowledge and insights regarding system development, support, and innovation.

An institution's current collaborative capability can be assessed by reviewing the answers to a few questions:

1. Do staff members across the institution know how to achieve institutional objectives through work in multi-institution, distributed communities?
2. Is the institution or key staff invited to participate in important community work that shapes community and software directions?
3. Does the culture of the organization reward staff time and commitment to communities?

Institutional collaborative capabilities must be developed over time and cannot be easily bought or spoken into existence. IT leaders should assess now how their staff and institutional policies enable or impede development of collaborative behaviors, since these behaviors arbitrate access to the greatest value from the community.

Conclusion

Colleges and universities and commercial firms have demonstrated great progress in realizing the vision proffered for "Open Source 2007," and 2010 will mark even

greater progress. Although much work remains in refining open source for higher education applications, the signals are now clear: the collaborative development of software can provide one of the most potent tools for IT leaders as they wrestle with four challenging trends for IT services.

The outcomes, pace, and shape of open-source collaborations remain fungible and in our hands. Thus 2007 is the year to revisit if and how open-source application software fits as part of a comprehensive IT strategy for a specific college or university. Given a multi-year lead time to develop an institution's robust ability to leverage the investments of others, now is not too early to begin this work even as various applications mature. In my view, developing a collaborative capability is not an option for 2010: it is a necessity for effective college and university IT organizations. *e*

Notes

1. Brad Wheeler, "Open Source 2007: How Did This Happen?" *EDUCAUSE Review*, vol. 39, no. 4 (July/August 2004): 12–27, <<http://www.educause.edu/cr/erm04/erm0440.asp>>.
2. Gregory A. Jackson, "Open Source Is the Answer: Now What Was the Question?" *Chronicle of Higher Education*, September 24, 2004, p. B17; Lev Gonick, "Open Source Is the Answer: Now What Was the Question?" *Bytes from Lev*, January 30, 2005, <http://blog.case.edu/lsg8/2005/01/30/open_source_is_the_answer_now_what_was_the_question>.
3. "Scholarship 2.0" is a term used to describe the rising role of IT, digital repositories, and electronic collaboration in maintaining or improving the quality of the scholarly endeavors of research, teaching and learning, and service.
4. Thomas J. Hacker and Bradley C. Wheeler, "Making Research Cyberinfrastructure a Strategic Choice," *EQ (Educause Quarterly)*, vol. 30, no. 1 (2007, in press); A.J.G. Hey and A.E. Trefethen, "The Data Deluge: An e-Science Perspective," in *Grid Computing: Making the Global Infrastructure a Reality*, ed. Fran Berman, Geoffrey Fox, and Tony Hey (New York: John Wiley & Sons, 2003), 809–24; "Revolutionizing Science and Engineering through Cyberinfrastructure: Report of the National Science Foundation Blue-Ribbon Advisory Panel on Cyberinfrastructure," January 2003, <<http://www.nsf.gov/cise/sci/reports/atkins.pdf>>; "Our Cultural Commonwealth: The Report of the American Council of Learned Societies' Commission on Cyberinfrastructure for Humanities and Social Sciences," July 18, 2006, <<http://www.acls.org/cyberinfrastructure/acls.ci.report.pdf>>.
5. Anne K. Walters, "U. of Missouri System to Redirect \$20-Million," *Chronicle of Higher Education*, August 4, 2006, p. A21.
6. Paul N. Courant and Rebecca J. Griffiths, "Software and Collaboration in Higher Education: A Study of Open Source Software," July 26, 2006, <<http://www.ithaka.org/about-ithaka/announcements/ooss-study-final-report>>.
7. *Ibid.*, 4.
8. Fuchs, personal communication with the author.
9. Brad Wheeler, "The Inevitable Unbundling of Software and Support," *Syllabus*, March 1, 2004, <<http://www.syllabus.com/article.asp?id=9026>>.
10. James Surowiecki, *The Wisdom of Crowds: Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies, and Nations* (New York: Doubleday, 2004).
11. Paul B. Gandel and Brad Wheeler, "Of Birkenstocks and Wingtips: Open Source Licenses," *EDUCAUSE Review*, vol. 40, no. 1 (January/February 2005): 10–11, <<http://www.educause.edu/apps/cr/erm05/erm0517.asp>>.
12. Matt Asay, "Open Source = Helping Customers Pay for What They Want?" *InfoWorld*, November 29, 2006, <http://weblog.infoworld.com/openresource/archives/2006/11/open_source_hel.html>.
13. The phrase "zero disruptive upgrades" comes from John (Barry) Walsh, director of University Information Systems at Indiana University and executive director of Quali Financial System, who observed that the homegrown financial system at Indiana University had been through forty-seven releases over ten years and that only two of them had involved a substantial disruption to the university community. ERP upgrades that consume an institution and freeze other work are not the best strategy. (Walsh, personal communication with the author.)
14. Steven Kerr, "On the Folly of Rewarding A, While Hoping for B," *Academy of Management Journal*, vol. 18, no. 4 (December 1975): 769–83.