Publisher's Note

The following is the second part of an exchange of e-mail is a discussion Sakai products, direction, and strategy. The first covered the period 27 through 30 October.

The discussion represents several different perspectives.

Message authors include Craig Counterman from MIT, Joseph Harden and Chuck Severance from the University of Michigan, John Norman from the University of Cambridge, and Mark Norton from the Sakai Educational Partners Program.

Routing information and references from the original e-mail has been deleted. If needed this information is available from the individual email .emi files that have been retained as source.

 30 Oct 2005 23:59:18 -0500 (EST)
Date: Sun, 30 Oct 2005 23:59:14 -0500
From: Craig Counterman <ccount@MIT.EDU>
Subject: Re: Sakai SPB
To: Jim Farmer <jxf@immagic.com>
Cc: Craig Counterman <ccount@mit.edu>

I just realized, you didn't answer one of my questions: exactly what are the "about five major open source LMS integration projects next year."?

I had started talking with Patrick Masson [SUNY Learning Network] about the LMOS since he mentioned it on a Sakai list.

I don't know if I should send them Sakai <whatever> Bundle work, I was expecting to, before this blow-up.

I started sharing the Spring service injection work I did before the January meeting when the framework work was assigned to Glenn. But I'm not sure how to describe it for the public record. Some elements are now in Sakai, but not all.

It's out there at
<http://cvs.sakaiproject.org/cgi-bin/cvsweb.cgi/scratch/fw2/projects/>
anyhow.

Craig

Date: Mon, 31 Oct 2005 10:31:16 -0500
From: Jim Farmer <jxf@immagic.com>
Organization: instructional media + magic, inc.
To: Craig Counterman <ccount@MIT.EDU>
CC: Justin Erik Tilton <jet@immagic.com>
Subject: Re: Sakai SPB

The five major integration projects would be:

1. State University of New York SLN2

2. University of North Carolina TLTC

3. California State University procurement + Moodle "insurgency"

4. JISC following the standards developed in conjunction with JISC/DEST/SURF

5. Moodle partners' uPortal+ Moodle + "tools"

No implications of order. Moodle partners' will be the least in terms of resources, but may become the pattern for JISC. North Carolina will follow SUNY, but likely include Moodle as an alternative for the community colleges. The CSU Chancellor's office will likely be forced to ASP and support Moodle as well as Blackboard. Hardly a Blackboard victory since two of the campuses have already announced dropping Blackboard and declining to pay for the license, even at the sharply discounted rates.

JISC is the big giant now with a mission--implementation by 2008.. Whether they can move from decentralized development to a centrally guided implementation is not yet clear. Here LAMS would be the favorite since the money is coming from a strategic plan by Diana Laurillard when she was number 2 at DfES. The target is K-16. Laurillard was a strong LAMS supporter because it implemented sequencing with group activities.

Patrick [Masson, SUNY Leaning Network] would appreciate and deserves any assistance you can provide. Today Justin and I gave some of your background to Ken Udas to whom Patrick reports. All complimentary, of course. Patrick was under the impression the Sakai framework could be separated from the Sakai Enterprise Bundle. SUNY would like to have Sakai Samigo and Sakai grade book and are assuming these are "Sakai tools" that can be "plugged in" to the Sakai framework.

jf

Craig Counterman wrote:

> I just realized, you didn't answer one of my questions: exactly what
> are the "about five major open source LMS integration projects next
> year."?
>
> I had started talking with Patrick Masson about the LMOS since he
> mentioned it on a Sakai list.
>

> I don't know if I should send them Sakai <whatever> Bundle work, I
was
> expecting to, before this blow-up.
>
> I started sharing the Spring service injection work I did before the
> January meeting when the framework work was assigned to Glenn. But
> I'm not sure how to describe it for the public record. Some elements
> are now in Sakai, but not all.
>
> It's out there at
> <http://cvs.sakaiproject.org/cgi-
bin/cvsweb.cgi/scratch/fw2/projects/>
> anyhow.
>
> Craig
>
>

--
Jim Farmer
+1-202-296-2807
cell +1-405-408-9264

Date: Mon, 31 Oct 2005 15:04:46 -0500
From: Mark Norton <markjnorton@earthlink.net>
Subject: Re: JISC in Sakai (etc)
To: Charles Severance <csev@umich.edu>
Cc: John Norman <john@caret.cam.ac.uk>, Joseph Hardin
<hardin@umich.edu>,
 Jim Farmer <jxf@immagic.com>, Mark Norton <markjnorton@earthlink.net>

Charles Severance wrote:

> I "talked" to Mark Friday morning on my drive into work - after some
> long "opening statements" - we switched from loudly describing the
> problem to finding a solution to the problem :).

It was indeed a "vigorous" discussion. Fortunately, my phone has a
volume limiter. :)

> The key here is that we need to make sure that the SKB does not seem
> to be a way to "be Sakai compliant".

Chuck is right in saying that writing tools on the Sakai Kernel Bundle doesn't make them Sakai compliant, in part because we really haven't defined what that means. To be sure, tools written only to the kernel are not very integrated into Sakai since they wouldn't be fully tied into the security model (AuthGroups), worksites, and other essential services.

> At some level we want Boddington to improve Sakai, raising all
boats,
> rather than take our "kernel" and build a competing solution that is
> completely incompatible and not interoperable with the rest of
Sakai.

I'd agree that it would be far better to fold the Boddington services (etc) into Sakai where all may benefit. As Chuck points out later in this message, we are not really at a point where it is easy to do this, though.

> The primary value here of Mark's work is that this is a really easy
> way to understand the basic plumbing of Sakai - it is something that
> new developers should do first - they should spend two weeks with the
> Sakai Kernel Bundle - learning about components and APIs, Spring,
> etc. Without needing long compiles and re-deploys and getting very
> frustrated watching Samigo compile for the 50th time and deciding
> that Sakai sucks because it is too big.

It's quite refreshing to work in a Sakai environment stripped to it's essentials and allows developers to really focus on the core elements of Sakai without having to wade through thousands of files. Besides being a good platform for training, I am developing a new set of documentation that I hope will further aid developers coming up to speed with Sakai.

> But then before they get serious about writing a real Sakai tool -
> they should move from the SKB to the Real Sakai so they can use all
> of the Sakai APIs and any helper tools they want and ultimately build

> a tool that "fits in" rather than stands alone.

I don't wholly agree with Chuck here, John. Granted the SKB is doesn't have anything but the kernel in it, but I'd say it really depends on what you are building and what phase of development you are at. What we are trying to avoid is monolithic, independent applications that are difficult to integrate into Sakai. We already have too many of those. The end goal is to build Sakai tools (and new services) that all can benefit from.

> To the extent that Mark and the SKB can a do training throughout the
> UK and convince folks that Sakai is not so hard to develop for would
> be a wonderful thing.

I will be working towards improving the training and documentation offerings going into 2006 and would be happy to discuss training workshops for next year.

> Mark is going to add some clarification to the document to keep SKB
> from getting mis-interpreted.

Yes, just as soon as I catch up with my email. ;-)

> We debated on what to call it.
>
> SPB - Sakai Platform Bundle - Oversells the notion of what "it" is -
> implies that this is suitable for long-term production use and
> deployment and implies that integrating with the "Platform" is
> equivalent to integrating with Sakai. Dangerously splits the brand.
> Scares me to the core.
>
> STB - Sakai Training Bundle - Undersells the notion of what "it" is -
> implies that it is only for training - actually it is a great place
> for people to start their development and get the basic stuff wired
> together and then once their "project" complies, injects, and comes
> up - move to the Real Sakai release.
>
> SKB - Sakai Kernel Bundle - To me (like in Goldilocks) this feels
> "just right" - it accurately represents that this is a small thing
> and allows up to use the term SPB in the future when we have such a
> thing.

Much as I'd like to agree with Chuck's nomenclature above, I am strongly considering adding the AuthGroup, Person, Group, and Site services to the bundle over the next month or so. These are essential for truly training people on how to develop good Sakai tools, but also lie well outside of the kernel. "Sakai Kernel Bundle" as a term is a bit limiting also.

> I think that at some point we should work towards a SPB - but this
> needs to include not only the Kernel, but in addition all of the
> APIs, and all of the helper tools.

Hmm. I don't agree. It doesn't need to include all APIs and helper tools. It should be modular enough that the basics are there and other services can be easily added and removed as needed for good tool development. It shouldn't be necessary to have all of Sakai present to

build a well-designed and focused tool.

> To build an SPB right at this point would require some massive and
> painful refactor. It would be best done as part of a long 3.0
> effort - not as a series of disruptive mods to the 2.x series.

First of all, I don't think it will be quite as much effort as all that. Chuck is also including the re-design of several Sakai services, but that needn't be part of an early release. The short term effort is in re-modularization. Done properly, this should have little or no impact on existing code at all. Naturally, this is something that can be assessed as it unfolds.

> In my mental schedule - that talks to a June 2007 release of an SPB
> and having the SPB be a natural part of the Sakai 3.0 release.

I have trouble visualizing Sakai that far out into the future. Our track record hasn't been all that great so far, but I do have a lot of hope for a community based approach to Sakai development. Surely, it's not too soon to be thinking about this and perhaps conducting a few experiments, though we wouldn't want it to disrupt the near term plans for Sakai.

> As a matter of fact, with proper investment, the Sakai 3.0 release
> could be the Sakai SPB and people just add stuff as they see fit.
> The SKB could be seen as the early pre-cursor to the SPB.

My thoughts, exactly. OTOH, I think we could start putting together an early version of the SPB much sooner than 2007. Furthermore, I think there are people in the Sakai community who need it sooner. Perhaps there is a compromise position?

Joseph and I spoke a bit about this at Educause. One of the main reasons for developing this kernel bundle was to lower the barriers of entry to developing Sakai tools and services. The more people who know how to do this, the better off we will all be. However, there is another aspect to this that needs serious consideration. Sakai needs to be inclusive in it's development efforts going forward. Sakai can no longer be a closed club of core developers.

While considerable strides have been made in being open, we have a long way to go. Inclusive means a lot more than just just communicating what's going on, it means actively encouraging people to participate, it means finding places for people to contribute at their current skill level, and it means being open to new ideas. This attitude is not only desirable to expand the range of projects that we can tackle, I believe it is critical for the sustainability of the Sakai project itself, because if people don't feel they are part of the project, they won't contribute money to it.

- Mark Norton

Date: Mon, 31 Oct 2005 15:09:28 -0500
From: Mark Norton <markjnorton@earthlink.net>
Subject: Re: JISC in Sakai (etc)
To: hardin@umich.edu
Cc: Charles Severance <csev@umich.edu>, John Norman <john@caret.cam.ac.uk>,
Jim Farmer <jxf@immagic.com>

Joseph Hardin wrote:

> This sounds much better to me. Mark, where are your intro docs for
> training with the SKB?

They are not quite ready for distribution at this time. I'm still working on them. The short term goal is to have them ready for use in the Algeria Workshop (next week). Distribution will be limited to CDROM there. Do you want to see the drafts in progress?

- Mark

 01 Nov 2005 06:39:33 -0500
Date: Tue, 01 Nov 2005 06:39:24 -0500
From: Charles Severance <csev@umich.edu>
Subject: Re: JISC in Sakai (etc)
To: markjnorton@earthlink.net
Cc: John Norman <john@caret.cam.ac.uk>, Joseph Hardin
<hardin@umich.edu>,
 Jim Farmer <jxf@immagic.com>

We are in 90% agreement here. But there are some areas of
disagreement.

On Oct 31, 2005, at 3:04 PM, Mark Norton wrote:

> Charles Severance wrote:

[agreed-stuff-snip]

```
>> But then before they get serious about writing a real Sakai tool
>> -  they should move from the SKB to the Real Sakai so they can use
>> all  of the Sakai APIs and any helper tools they want and
>> ultimately build  a tool that "fits in" rather than stands alone.
>>
>
> I don't wholly agree with Chuck here, John. Granted the SKB is
> doesn't have anything but the kernel in it, but I'd say it really
> depends on what you are building and what phase of development you
> are at.  What we are trying to avoid is monolithic, independent
> applications that are difficult to integrate into Sakai.  We
> already have too many of those.  The end goal is to build Sakai
> tools (and new services) that all can benefit from.
```

**Mark, even the simplest of useful tools cannot deploy on the SKB.
SKB in its current form is way too small and even with the addition
of 3-4 more APIs it is still way too small.**

[snip]

```
>> We debated on what to call it.
>>
>> SPB - Sakai Platform Bundle - Oversells the notion of what "it" is
>> -  implies that this is suitable for long-term production use and
>> deployment and implies that integrating with the "Platform" is
>> equivalent to integrating with Sakai.  Dangerously splits the
>> brand.   Scares me to the core.
>>
>> STB - Sakai Training Bundle - Undersells the notion of what "it"
>> is -  implies that it is only for training - actually it is a
>> great place  for people to start their development and get the
>> basic stuff wired  together and then once their "project"
>> complies, injects, and comes  up - move to the Real Sakai release.
>>
>> SKB - Sakai Kernel Bundle - To me (like in Goldilocks) this feels
>> "just right" - it accurately represents that this is a small
>> thing  and allows up to use the term SPB in the future when we
>> have such a  thing.
>>
```

>
> Much as I'd like to agree with Chuck's nomenclature above, I am
> strongly considering adding the AuthGroup, Person, Group, and Site
> services to the bundle over the next month or so. These are
> essential for truly training people on how to develop good Sakai
> tools, but also lie well outside of the kernel. "Sakai Kernel
> Bundle" as a term is a bit limiting also.

The problem here is whether or not you intended to include these in their current form, or demand refactoring of these interfaces before they are ready for inclusion. If the need to evolve the SPB becomes something that forces a bunch of re-factoring when it is not the right time for that work, then we have to cover two independent thrusts or complex work with one set of resources. The question here is whether the need to expand the SKB trumps the rest of the needs for effort within Sakai.

I claim that the answer is "no" - we need to finish the 2.x work and then start on the 3.x work. Trying to make an SPB in the margins will not be successful. And claiming that we can add "one more thing" to make it better is true, but there is not enough little things to make it worthwhile as a separate deployable distribution.

>
>> I think that at some point we should work towards a SPB - but
>> this needs to include not only the Kernel, but in addition all of
>> the APIs, and all of the helper tools.
>>
>
> Hmm. I don't agree. It doesn't need to include all APIs and
> helper tools. It should be modular enough that the basics are
> there and other services can be easily added and removed as needed
> for good tool development. It shouldn't be necessary to have all
> of Sakai present to build a well-designed and focused tool.

Mark - Without thinks like helpers available and the rich APis people will just start solving their own problems and we will end up with "monolithic, independent applications" as you describe above. It is already difficult to get people to stop re-inventing the wheel for each application. You are proposing long-term use of a framework that effectively demands that they reinvent the wheel.

You keep sliding away from the notion that this is a "training tool" to "this is the way to completely develop a certain class of applications". This is a fine way to develop "monolithic stand alone applications".

>> To build an SPB right at this point would require some massive
>> and painful refactor. It would be best done as part of a long
>> 3.0 effort - not as a series of disruptive mods to the 2.x series.
>>
>
> First of all, I don't think it will be quite as much effort as all
> that. Chuck is also including the re-design of several Sakai
> services, but that needn't be part of an early release. The short
> term effort is in re-modularization. Done properly, this should
> have little or no impact on existing code at all. Naturally, this

> is something that can be assessed as it unfolds.

Right - but lets name the product after this "unfolds" not before.

>> In my mental schedule - that talks to a June 2007 release of an
>> SPB and having the SPB be a natural part of the Sakai 3.0 release.
>>
>
> I have trouble visualizing Sakai that far out into the future. Our
> track record hasn't been all that great so far, but I do have a lot
> of hope for a community based approach to Sakai development.
> Surely, it's not too soon to be thinking about this and perhaps
> conducting a few experiments, though we wouldn't want it to disrupt
> the near term plans for Sakai.

Our poor track record is a good reason *not* to invent new demands for scarce resources and then apologize for the next few years for not making progress on the promises we make about the "SPB".

>> As a matter of fact, with proper investment, the Sakai 3.0
>> release could be the Sakai SPB and people just add stuff as they
>> see fit. The SKB could be seen as the early pre-cursor to the SPB.
>>
>
> My thoughts, exactly. OTOH, I think we could start putting
> together an early version of the SPB much sooner than 2007.
> Furthermore, I think there are people in the Sakai community who
> need it sooner. Perhaps there is a compromise position?

Everything in Sakai is "needed sooner". This is just one more thing in Sakai that we "need sooner".

> Joseph and I spoke a bit about this at Educause. One of the main
> reasons for developing this kernel bundle was to lower the barriers
> of entry to developing Sakai tools and services. The more people
> who know how to do this, the better off we will all be. However,
> there is another aspect to this that needs serious consideration.

The problem here is that there are one of two ways to "reduce barriers to entry". Once is to make things better and clearer and another is to redefine "entry" to make it really easy. In its current form SKB is a training tool and not much more - it is not a form of "entry".

If we treat this as a "form of entry", we effectively fork the project.

/Chuck

01 Nov 2005 07:42:25 -0500
Date: Tue, 01 Nov 2005 07:42:17 -0500
From: Charles Severance <csev@umich.edu>
Subject: Sakai Platform Bundle - Not
To: markjnorton@earthlink.net
Cc: John Norman <john@caret.cam.ac.uk>, Joseph Hardin <hardin@umich.edu>,
Jim Farmer <jxf@immagic.com>

The one thing that our (and all other open source projects) license agreement prohibits against is taking the Sakai code base making modifications and separately distributing it with the same name. The one thing worth protecting in an open source project is the "brand" that software represents.

You cannot take Apache, remove 97% of it and call it the "Apache Platform" and distribute it or remove 97% of JBoss and call it the "JBoss Platform" or take MySql, remove 97% of it and call it the "MySql Plaform" or take uPortal 3.0 remove 97% of it and call it the uPortal Platform Edition.

I would suggest that if right now Mark was talking to the Moodle community, suggested that he remove 97% of Moodle and the distribute it as the "Moodle Platform Bundle" - they would not be particularly receptive to the notion.

Of course the uPortal Platform, JBoss Platform, MySql Platform, Apache Platform, Moodle Platform, and Sakai Platform are easier to use - they have 97% removed.

We are not debating whether a Platform Release is a good idea - it is a good idea and fits nicely into our roadmap with proper investment.   It has to be more than 3% of Sakai.

We are debating, whether to allow Mark Norton to immediately define the term "Platform Bundle" and then manage that release on behalf of the project going forward. This is a profound brand question. It effectively cedes half of the brand to Mark Norton.

Mark can take the Sakai code and make a copy of it and call it the "Norton Platform Bundle" with "portions copyright Sakai Foundation" - or John can take this and call this the "CARET Platform Bundle" with "portions copyright Sakai Foundation". This is fine - this is how Open source works.

But to call it "Sakai Platform Bundle" can only happen in the context of this project. We are making a commitment on the part of the project not on the part of Mark Norton.

This is similar to when Indiana wanted a "special branch" this Fall because the maintenance branch was so far behind trunk that it was useless. They wanted to call it "sakai-prime" - I said "no - that implies that the project is making a commitment" - so we renamed it to "indiana" to make it very clear what commitment was being done. This worked well and the real solution will be to fix the maintenance branch and have it more responsive to production needs rather than making a different branch.

The theme is to stick with something and *fix* its problems we have rather than creating something new because initially it seems to side-step the problems.   These side step moves never truly avoid the problems - generally they make things worse.

If we are going to produce two versions of Sakai in the market (one of which I do not control), I would prefer that the second version be named something other than "Sakai" - so that "Sakai" actually continues to mean "Collaboration and Learning Environment" and represents software that people can download and run in production.

Lets call it the "Norton Platform Bundle" with portions copyright Sakai Foundation. But not the Sakai Platform Bundle unless we are going to do it "right" and in a way that is coordinated with the rest of the project.

/Chuck

01 Nov 2005 13:36:13 +0000
Date: Tue, 01 Nov 2005 13:36:11 +0000 [08:36 –5000]
From: "John Norman" <john@caret.cam.ac.uk>
Subject: RE: Sakai Platform Bundle - Not
To: "'Charles Severance'" <csev@umich.edu>,
<markjnorton@earthlink.net>
Cc: "'Joseph Hardin'" <hardin@umich.edu>, "'Jim Farmer'"
<jxf@immagic.com>

I think we should shelve this issue for a few weeks. We are busy here with the release, I imagine Chuck is busy too and expect Mark will have a heavy testing and documentation load around the release. I will not be pursuing any such option until the concept is subject to greater consensus and I think we will have a better chance of reaching consensus when we are all under less pressure.

Best, John

```
> -----Original Message-----
> From: Charles Severance [mailto:csev@umich.edu]
> Sent: 01 November 2005 12:42
> To: markjnorton@earthlink.net
> Cc: John Norman; Joseph Hardin; Jim Farmer
> Subject: Sakai Platform Bundle - Not
>
```
[Text omitted]

Date: Tue, 01 Nov 2005 08:41:33 -0500
From: Joseph Hardin <joseph.hardin@gmail.com>
Subject: Re: Sakai Platform Bundle - Not
To: Charles Severance <csev@umich.edu>
Cc: markjnorton@earthlink.net, John Norman <john@caret.cam.ac.uk>,
 Jim Farmer <jxf@immagic.com>

The way this works is: if you can come to agreement, then we are all
happy.
If not, then it gets elevated to the Board. The Sakai Board is
naturally
very concerned about issues of branding.

Joseph

On 11/1/05, Charles Severance <csev@umich.edu> wrote:
>
> The one thing that our (and all other open source projects) license
> agreement prohibits against is taking the Sakai code base making
> modifications and separately distributing it with the same name.
> The one thing worth protecting in an open source project is the
> "brand" that software represents.
>

[Text omitted]

--

Joseph Hardin
School of Information
University of Michigan
Sakai Foundation
734-763-3266

Date: Tue, 01 Nov 2005 09:10:35 -0500 (GMT-05:00)
From: markjnorton@earthlink.net
Subject: Re: JISC in Sakai (etc)
To: Charles Severance <csev@umich.edu>
Cc: John Norman <john@caret.cam.ac.uk>, Joseph Hardin
<hardin@umich.edu>,
Jim Farmer <jxf@immagic.com>

> We are in 90% agreement here. But there are some areas of
disagreement.

If we ever got to 100%, world peace can't be far behind! :)

> Mark, even the simplest of useful tools cannot deploy on the SKB.
> SKB in its current form is way too small and even with the addition
> of 3-4 more APIs it is still way too small.

I feel like I've been manuevered into defending the SKB as a development environment, when I don't really advocate it. However, it could, in fact be used given the addition of a few more services, authorization in particular.

Let's take a case in point. The members of the Tool Development Exercise are creating a pedagogical tool right now to manipulate data tables. It will be deployed using the conventions of Sakai meaning it will be registered as a tool and operate correctly in the Charon Portal. It has an application service to manuplate its data. Currently, the only thing missing from that service not provided by the SKB is authorization. This is a real tool, Chuck.   The members of the group expect Sakai users will be intersted in using it.

So, while I am in no way trying to overemphasize what the SKB can do, I ask you not to trivialize it either. The SKB contains almost all of the basic support for a tool to operate in the Sakai environment. That is what the kernel does, Chuck. What is missing is the richness of the rest of Sakai services.

> Much as I'd like to agree with Chuck's nomenclature above, I am
> strongly considering adding the AuthGroup, Person, Group, and Site
> services to the bundle over the next month or so. These are
> essential for truly training people on how to develop good Sakai
> tools, but also lie well outside of the kernel. "Sakai Kernel
> Bundle" as a term is a bit limiting also.

> The problem here is whether or not you intended to include these in
> their current form, or demand refactoring of these interfaces before
> they are ready for inclusion.

What demands are you talking about, Chuck? This is a collaborative project. It's a proposal, not a demand.

> If the need to evolve the SPB becomes
> something that forces a bunch of re-factoring when it is not the
> right time for that work, then we have to cover two independent
> thrusts or complex work with one set of resources.

I guess that depends on priorities to some extent. If there is a need for such a bundle, shouldn't we be addressing it? If the refactoring (as you call it, I think that term is overloaded a bit) has minimal impact on existing code, where is the problem? If it goes call for a major impact, that would have to be balanced against other work. Finally, think about Sakai as a limited set of resources is a self-fullfilling viewpoint. Serializing development to a small set of developers excludes those who might be interested in working on a parallel effort.

> The question here
> is whether the need to expand the SKB trumps the rest of the needs
> for effort within Sakai.

I'm trying to understand why this has to be an either / or proposition. I'm not advocatiing a major redesign of Sakai. I'm suggesting that there is a way to reduce Sakai to a much simpler development environment. Furthermore, I think resources can be found to do it.

> I claim that the answer is "no" - we need to finish the 2.x work and
> then start on the 3.x work.

A serial development approach will not scale -- plain and simple. Parallel development efforts do require coordination and communication, but isn't that the essence of an open source project? Properly organized, it should be possible to do more than one thing at once, even more than one major thing at once. However, as Chief Architect, this is your call to make and I respect that.

> Trying to make an SPB in the margins will not be successful.

Yeah, that's likely to be the case. Why does it have to be done on the margins? Why not make it a project now for distribution a year from now, or sooner if it can be done?

> And claiming that we can add "one more
> thing" to make it better is true, but there is not enough little
> things to make it worthwhile as a separate deployable distribution.

Well, that is a value judgement, I guess. You could be right, but I'd be interested in hearing what the rest of community thinks about it's potential value.

> Mark - Without thinks like helpers available and the rich APis people
> will just start solving their own problems and we will end up with
> "monolithic, independent applications" as you describe above. It is
> already difficult to get people to stop re-inventing the wheel for
> each application. You are proposing long-term use of a framework
> that effectively demands that they reinvent the wheel.

No, what I am proposing is to allow people to bring in code when they need it and not be burdened with it if they don't. Yes, people should be encouraged (even required) to follow Sakai conventions, especially UI ones that call for helper apps. Modular as opposed to monolithic.

> You keep sliding away from the notion that this is a "training tool"
> to "this is the way to completely develop a certain class of

> applications". This is a fine way to develop "monolithic stand alone
> applications".

Well, I do admit that I thought of the idea and caused it to happen (in it's current, limited form). It is a good training tool. How people use it beyond that is the debate here. Your attempts to minimize the part of Sakai designed to provide support for tools causes me to defend it. Let me state this clearly: there is no difference between the code in this bundle and what is distributed with the Sakai Enterprise Bundle. It is purely a matter of what is included and what is not. The code it includes is identical to Sakai mainline.

> Our poor track record is a good reason *not* to invent new demands
> for scarce resources and then apologize for the next few years for
> not making progress on the promises we make about the "SPB".

It is not my intention to distrupt the current development schedule at all.

>> Perhaps there is a compromise position?

> Everything in Sakai is "needed sooner".   This is just one more
> thing in Sakai that we "need sooner".

So there is no compromise solution then?

> The problem here is that there are one of two ways to "reduce
> barriers to entry". Once is to make things better and clearer and
> another is to redefine "entry" to make it really easy. In its
> current form SKB is a training tool and not much more - it is not a
> form of "entry".

So how will we make it easier to develop tools for Sakai? Throw a million lines of code at every new developer?

> If we treat this as a "form of entry", we effectively fork the
project.

I have no desire to fork the Sakai code. I strongly believe that keeping community together is important to the success of Sakai. It would be sad to think that new ideas and approaches must be done outside of Sakai.

- Mark