Designing for innovation: Endeavor Information Systems, Inc., and the Law of Unintended Consequences –- a meditation and manifesto in the year of Endeavor's 10th birthday

Jim Robertson, Assistant University Librarian
New Jersey Institute of Technology
james.c.robertson@njit.edu
web.njit.edu/~robertso

Transcript of a talk presented at EndUser 2005, the annual users group meeting of Endeavor Information Systems, Inc.
Session 4, Thur, Apr 28, 2005, 1:15 pm.

Abstract:

A "think piece" looking at how market and industry concepts such as community coding, disruptive technologies, open and transparent systems, the Law of Unintended Consequences, and loosely coupled systems illustrate lessons for Endeavor (and other ILS vendors) as they develop and evolve their products and services. Discussion on how ILS vendors must embrace these realities in order to provide the right environment and platforms in which innovation can flourish in their customer's libraries. Stagnated innovation equals irrelevance and obsolescence; libraries and ILS vendors must innovate or die. It can be achieved given a paradigm shift that embraces the market and industry concepts reviewed in this talk.

Introduction:

The seed for this talk was planted a year ago at EndUser 2004. It was the third year in a row I had been to EndUser and, like usual, I was blown away by the creativity of the Endeavor user base. The creative ways in which these librarians were employing Voyager to provide great services to their users. The lengths to which they were "hacking" their Voyagers to get the innovations they wanted for their users.

And … the ways in which Endeavor doesn't always make it easy for us librarians to do those things. The ways in which the Voyager system hasn't been designed from the outset – from the ground up -- to allow for some of these processes. These new needs. New services.

We had librarians chomping at the bit to really employ and extend Voyager to do new things – newly conceived things – and going to great lengths to get their systems to do them. Or to do some approximation of what we wanted to do.

After a few days of the conference talking to other like-minded librarians … I figured I would commit to giving this talk at EndUser 2005. It would force me to get my thoughts together on this issue and weave together other concepts I had floating around in my head. Concepts such as unintended consequences, loosely coupled systems, the Lazy Web, gift economies, and more.

It also seems appropriate as Endeavor celebrates its 10th year to step back and do some reflection on the direction of their ILS and other products and the paradigms behind them.

Before I start, three points.

One: I think Endeavor is great and I enjoy our partnership with them. Almost nothing I'm about to say can't be equally applied to any other ILS vendor … or BlackBoard or Peoplesoft or whomever. This talk is meant to be "critical" in the sense of "critical thinking."

Low cost & high impact (concentrate efforts here)	Low cost & low impact
High cost & high impact	High cost & low impact (ignore!)

Second: I'm not sure if the future action will be in the ILS anyway. Roy Tennant, among others, questions the wisdom of continuing with our inventory-focused, MARC-based catalogs.

Nonetheless, its inventory we all have. And here we are in Chicago. So, I'll soldier on. Really: I think some of the things I'll describe here are in the right "box" on the low-cost/high-impact square. For a little investment and a bigger shift in thinking, we can increase the value our ILS systems greatly for our users and ourselves.

And third: you're going to see lots of PowerPoint presentations over the next three days here at EndUser. But not here. I'm going to try something different. I'm just going to talk, and maybe we can think together.

Our catalogs as symphonies:

In casting around for an analogy that illustrated this vision I have for what Voyager could and should be, I hit upon the idea of our catalogs as symphonies. It may not be a perfect analogy, but bear with me.

Think of a symphony. You've got an audience, a concern hall, a conductor, sheet music, an orchestra with players, and instruments.

The audience is in place. The conductor has his sheet music. Lines and dots and squiggles. Encoded information. Hmmm – encoded information? We'll come back to that in a minute.

The conductor reads the sheet music – the script, if you will – and conducts the orchestra on the fly. These notes tell him to bring in the violins. So, he cues the violins and in they come, turning the encoded information into something real. Enjoyment, mood, feelings, whatever.

OK, here comes the cue for the flautist. So, in comes the flautist, adding dimension to the song. Does the sheet music (the "script") call for the timpani? No? No matter. The concert doesn't come to a screeching halt. This particular script doesn't provide any information for the timpani. No matter; perhaps the next piece in the concert does.

So, there was have it – as the audience – knowledge assembled on the fly for us from the encoded information. For our enjoyment and enrichment.

As the audience, we don't need to know how to interpret the sheet music. To us, it's just dots and lines. And, we don't need to know how to play the instrument. The links in the chain – sheet music to conductor to player to instrument – assemble the whole.

Making our catalogs sing:

Now, think of the sheet music as the MARC record. Think of the conductor as Voyager. Think of the players as web services. And the instruments as the remote systems and databases behind the web services.

So, Voyager (the conductor) puts some encoded information in front of us. Instantly and on-the-fly, it "conducts" the "assembly" of that information into "knowledge." Does the MARC record have encoded author information? Great: display it, but also use the author's authority information to query another web service (perhaps the trumpet player in our little analogy) and pull back the author's bio from a free or paid author bio service.

Similarly, does the information provide an ISBN? Great! Query these other five web services (our violin section, perhaps) to pull back information on other versions of this book via the xISBN web service, the book cover from our consortium's book cover image repository, usage statistics on this book from our local book usage data mine, professor ratings and comments on this book from our university system's rate-this-book service, and SIP information from Amazon's Statistically Improbable Phrases API.

Does the information also provide an ISSN so we can query the newly-launched service that provides the tables of contents for research journals via RSS feeds? No, not this record? OK, no problem. The ISSN-RSS web service will sit this one out. Maybe the next record will have an ISSN and call this service. No big deal.

Can we take the keywords the user entered and display to the user that there are 18 articles in the Scopus database and 14 articles in Lexis-Nexis matching those keywords in which they might be interested?

Thought of in this way, our catalogs are less about displaying the static information residing in the dusty MARC record and more about using the proper "triggers" in the MARC record to launch value-added web service processes that – on-the-fly – grab and deliver information back to the catalog. And, to the user.

The result that the user sees has been conducted and assembled from the best data available at that time. The whole is greater than the sum of its parts.

Amazon.com's Statistically Improbable Phrases, or "SIPs", are the most distinctive phrases in the text of books in the Search Inside! program. To identify SIPs, our computers scan the text of all books in Search Inside. If they find a phrase that occurs a large number of times in a particular book relative to all Search Inside books, that phrase is a SIP in that book.

» Amazon.com

Six categories of value-added:

1. ease of use
2. noise reduction
3. quality
4. adaptability
5. time savings
6. cost savings

» Taylor, Robert

But, wait, the Syndetics Solution:

Wait, some might say. Voyager has allowed something like this since version 2001 via the HTMLColumnC=ISBN tag in the opac.ini and the HTML:020||a: tag in the displayX.cfg files.

Oh … you mean the 14 pages resigned to Appendix A at the back of the 900 page WebVoyage manual? Of course … Not!

In a great injustice, what I think has been the most far-reaching, breaking-the-box, empowering-the-librarian enhancement since my library joined the Endeavor family four years ago was stuck at the end of the manual almost as an afterthought.

Now, I don't mean any disrespect to anyone in Endeavor. Designer. Documentation writer. Whoever. Endeavor has a great product. Its employees work hard. I've gotten lots of great help and partnership from Endeavor employees.

But, the lack of imagination shown in the launching of that enhancement is a jumping off place for my next point.

The Law of Unintended Consequences:

If you read Appendix A, you'll see that feature documented in very specific way. It is written in a very narrow way that only addresses Syndetics Solutions. At the risk of being wrong, I'm going to go out on a limb and suggest that's the way it was conceived and designed by Endeavor, as well.

Perhaps it was: hey, we've got a big customer or potential customer requesting the ability to use Syndetics Solutions book cover and tables of contents data. Here's our solution.

Perhaps it was: hey, Syndetics has contacted us asking us to find a way for its customers to use its service within Voyager. Let's design it.

Or, perhaps it was: hey, we've got this enhancement request to enable Syndetics data. He you go.

In each case, the result was a specific enhancement designed for a specific need. "You asked for this. Here is it."

No, no, no.

That little "escape hatch" … that modestly positioned "hole in the system" or "leak in the dyke" … is just the foot in the door to actually doing a whole lot more with Voyager.

At NJIT we have used that little bit of code to suck in book cover images from our **own** book cover image repository. No need to contract with Syndetics. In fact, if you read the book *Amazon hacks*, you can use Amazon's API syntax to suck into Voyager a book cover image for virtually **every** book in your catalog. Direct from Amazon's server. At NJIT, we actually did just that, if only for five minutes one afternoon just to prove the concept.

The only reason we turned it off was because of Amazon's web services licensing agreement. They encourage such active use of their "platform," but only at less than one hit per second and no more than 10,000 requests per 24 hours period. If I left this live in our Voyager catalog, I really had no way to enforce such limits and figured we'd definitely go over on the "less than one hit per second" limitation during many hours of the day.

Once we figure out how to do our own local implementation of a book cover repository, we realized the tremendous power and potential of this feature. And, the ways in which we are exploiting that "hole" in Voyager at NJIT are no doubt unanticipated by Endeavor when they created it.

We are using the ISBN to link out to a web service that gathers book usage data from the Voyager Oracle tables on the fly and presents it to our users. We are using it to display item status information to our users (only an enhancement that has been on the table for Voyager to implement for the last three years at least, perhaps longer!). We are using it to generate persistent human-readable "shortcut" URL links back into Voyager for that specific book (bypassing all the Voyager cgi-bin syntax). We are using it to point our users at book reviews in Amazon. We are using it to throw a user into a search in the catalog of two other university libraries in our town to which our students have borrowing

You may make calls by ASIN, browse ID, Listmania(r) ID or keyword (or by any other method Amazon may permit in the future) to Amazon Web Services 24hrs a day, 7 days a week, provided you: (i) do not exceed 1 call per second or send files greater than 40K, and (ii) in regards to obtaining the Alexa Data, do not exceed a maximum of 10,000 requests per 24 hour period. If you build and release an Application, the 1 call per second limitation applies to each installed copy of the Application.

» Amazon.com Web Services Licensing Agreement

privileges. Our copy is being borrowed? Here's one-click check to see if you can walk a few blocks and get it.

We are using the ISSN to point our users at our Serials Solution-provided A-to-Z list of e-journals and database holdings so they can find alternatives to our print copies. We are using it to resolve against a locally-created index of journal tables-of-contents available as RSS feeds.

All these are the **unintended consequences** of Endeavor's attempt to provide a Syndetics solution.

iPods:

iPods are beautiful things. Well designed, robust, and fun. I want to talk about them for a minute in light of what I've been discussing about integrated library systems.

Did you know, you can go to a web store like everythingipod.com and purchase hundreds of different add-on products for iPods? Cases, external speakers, clips, water-resistant enclosures, FM transmitter. You can buy a plug-in that turns your iPod into a voice recorder. Another one turns your iPod into a flashlight and laser pointer.

All of these leverage the inventive and simple design of the iPod. **None** of these ad-ons were designed, manufactured, or sold by Apple.

Richard Doherty, research director at the Envisioneering Group, estimates that the typical iPod user spends up to half the cost of his or her iPod in add-ons. That's another $150 for a $300 iPod to the ad on market.

Did Apple anticipate this "downstream" or "ancillary" market for IPod add-ons? I honestly don't know, but the point is that the IPod was designed from the outset to **allow** such third-party products to be conceived, designed, created, and find a market. It's simple, elegant, and "open" design made this possible.

Another similar example is the ring-tone market for cell phones. That "micro-content" market has been estimated at over $3 billion worldwide in 2003 (Kaser). When Motorola or Nokia designed their cell phones, it was genius to allow the user to add their own ring tones.

Some popular and rap artists have achieved "best seller" status on the ring-tone market. Others have negotiated deals for music specifically written and recorded for the ring-tone market.

Skins for these phones are also another popular "downstream" market. The ability for a user to customize and personalize their phone was another example of design that supported an unintended consequence.

Filmmaking and web design:

Well, some might say, "if you are able to do all this kind of stuff that you've done at NJIT – the linking to Amazon book reviews, the linking to usage statistics--what's the problem? You can also embed OpenURL links. I mean, aren't you really describing a resolver like LinkFinderPlus?" There are two responses to that.

One, this was accidental. Incidental. By chance. So, what I'm pushing is for a more **purposeful** and **intentional** design of Voyager capabilities – in all modules -- with this in mind from the outset. From the ground up.

And two. There a caveat of playwriting, movie writing, directing, acting. Don't **tell** the audience. Don't **explain** it to them. Just **show** them.

Haven't you read a review that complained about the sheriff who shows up in act three just to act as an audience surrogate so the protagonist can explain the story thus far? Or, a review that criticized the director's choice to use the hero's voiceover to explain his emotions instead of letting the audience see his emotions through his acting?

Similarly, Jakob Neilsen – the web usability guru – has written in his *Homepage usability* book about using **examples** to reveal a web site's content. He writes that on e-commerce sites, a category and link labeled "Sale items" its not so nearly compelling as seeing a few actual products that are on sale.

> "You stand a better chance of piquing user interest if you provide something concrete to read or look at instead of just abstract category names." Also, "examples can help users

> successfully navigate, because they show what lies beneath the abstract category names." (p. 16)

Now, our catalogs are not e-commerce sites, but we're still competing for eye-time and mindshare. So, applying that logic, don't give me a link that **tells** me "click here for author bio" and sends me out through a resolver somewhere or **explains** to me "click here for usage stats" and throws me out into another database.

Show me: "Stephen Edwin King was born in Portland, Maine, in 1947, the second son of Donald and Nellie Ruth Pillsbury King. After his parents separated when Stephen was a toddler …" (click for more). And: "This book borrowed 56 times this year. Ranked #18. More."

Also, resolvers make information two clicks (or sometimes one click) away. What I'm describing pulls content in with zero clicks.

Now, how is it possible to accomplish some of these ideas?

The programmable web – web services and APIs:

The web has really evolved from its infancy as a series of linked static documents into a giant programmable interface. Some of the leaders in pushing the envelope of the programmable web are Amazon, e-Bay, and Google.

They have all created and exposed ways for people to pull information out and link information back into their web sites. Books like *Amazon hacks*, *e-Bay hacks*, and *Google hacks* document and demonstrate these tricks.

Writing of Amazon and eBay, Aaron Ricadela wrote in *Information week* magazine: "Now they're emerging as hubs of software-development activity, where openness breeds innovation and innovation generates sales." (Or **use**, perhaps, for us non-commercial world.)

And, "There's an enormous amount of information and interesting processes that are out there, and if it was all exposed via Web services so that developers could integrate that stuff together, people would do incredible things with it."

(By the way, I was very encouraged to hear Roland Dietz speak about web services in his Endeavor Update presentation earlier this morning.)

Amazon's web services licensing agreement (of which I spoke earlier when I had to dismantle our Amazon book covers test) even defines itself as: *a **platform** which enables the creation of websites and applications that perform various functions, such as enabling and completing transactions, retrieving information about Amazon products or adding a product to an Amazon shopping cart, wish list* ... etc.

Not a bookstore. Not an e-commerce site. A **platform**.

Wouldn't it be great if our ILS systems were platforms, too? Instead of the "designed-for-a-purpose" "dead-end-destinations" they are currently.

Indeed, they also call it the "Amazon.com ecosystem." Wouldn't it be great if our catalogs were similarly key parts of our universities' "knowledge ecosystem?"

The *Amazon hacks* book writes:

> "*Amazon hacks* is much more than a guide for getting the most out of Amazon.com as it is today – it is a call to all true hackers out there to innovate on the platform … the ingenious applications that developers have built in the last year using the rich content and features is just the tip of the iceberg of what is to come." (Bausch, Foreword, p. xiii)
>
> "By lowering the barriers to entry and experimentation on top of the Amazon platform, we invite true hackers to extend and enhance the platform for all to enjoy – including us!" (Bausch, back cover).

"Lowering the barrier." "Experimentation." "Extend and enhance." "All to enjoy."

Wow.

One example of an innovative service that has taken advantage of Amazon's platform is junglescan.com. That third-party website tracks the daily sales rank

Library systems in a broader network environment; four emphases:

- public platforms
- Personal and interpersonal spaces are major focuses
- Service orientation and on-demand platforms
- The web all the way

» Dempsey, Lorcan. http://orweblog.oclc.org/archives/000622.html

of books on Amazon over time and puts them into stock-market-like charts. Amazon doesn't offer this themselves, but via the API they've published, a third party can hack in and get this data out for its site for its own use.

Another is allconsuming.com. It tracks the books people mention in blog posts. It tracks "hot" books, people's comments, and pulls details information from Amazon (via the ISBN) to build a rich website of "books with buzz."

Other creative users include Grokker.com, iPilot.net, monsoonretail.com, naturallyopen.com, and yes.net.

> While companies such as Google and Microsoft are also experimenting with the idea of letting outsiders tap into their databases and use their content in **unpredictable** ways … none is proceeding more aggressively than Amazon. The company has, in essence, **outsourced much of its R&D**, and a growing portion of its actual sales, to an army of thousands of software developers, who apparently enjoy nothing more than finding **creative new ways** to give Web surfers access to Amazon merchandise—and earning a few bucks in the process. (Roush)

Data and services need to be made available in ways which better facilitate their recombination in different user contexts. This touches on what I have called intrastructure, the applications tissue that allows us to more easily stitch together systems and services. RSS feeds, URL-based web services, bookmarklets, data import and export: these are all boundary crossing services which enable better stitching.

» Dempsey, Lorcan. http://orweblog.oclc.org/archives/000585.html

These web site examples -- as well as many others -- are using techniques like API's (Application Programming Interface) and web services (able to communicate through XML over HTTP and via SOAP, SOAP is Simple Object Access Protocol) to communicate. These are ways for two different computer systems to ask each other questions and get structured answers in return -- without having to know the internals of the other's technology architecture or business structure.

Equally important is that these web services exist in a loosely coupled fashion.

Loosely coupled systems:

www.looselycoupled.com writes that the concept is:

> "The friction-free linking enabled by web services. Loosely coupled services, even if they use incompatible system technologies, can be joined together on demand to create composite services, or disassembled just as easily into their functional components. Participants must establish a shared semantic framework to ensure messages retain a consistent meaning across participating services."

Assembling on-demand services to automate business, commerce, and the sharing of knowledge

» www.looselycoupled.com/blog/2002_07_14_lc.htm

Loose coupling is tolerant of failures. Because the web will always be a little bit broken, according to inventor Tim Berners-Lee (Weinberger, p. 76). It was **designed** to be somewhat broken.

> The imperfection of the web isn't a temporary lapse; it's a design decision. It flows directly from the fact that the web is unmanaged and uncontrolled so that it can grow rapidly and host innovations of every sort. The designers weighed perfection against growth and creativity, and perfection lost. The web is broken on purpose. (Weinberger, p. 79)

So, broken links, down servers, network traffic jams and so forth are expected and accepted. The web is robust. An experiment in loose coupling will not collapse the web or your core system.

Interestingly, before we leave that last quote, let me emphasis part of it again: the web was **designed for innovation**. And, perfection was pushed aside in order to allow for such a design. Perfection didn't matter; **innovation and openness mattered**.

That's another lesson for us.

And, since lack of perfection means failure sometimes, failure sometimes requires loose coupling. A way to anticipate and expect failure. To account for it. To "roll with the punches."

As I tell my own programs and applications: fail, but fail gracefully. In the symphony example from the beginning of my talk: is the violinist ill today? No problem. The whole orchestra doesn't stop playing; we play on with the rest of the orchestra and manage pretty well.

Some of the key benefits of loosely coupled web services are (Kaye, pp. 29-31):

- Independence
- Standardization
- Modularity and granularity
- Reusability
- Lower costs
- Reduced brittleness
- Scalability

Because when exposed web services and link them in loosely coupled ways, great things can happen.

*Innovation will come from APIs that support '**unintended consequences**'* -- Tim O'Reilly (also at http://www.oreilleynet.com/cs/weblog/view/wlg/1707)

*Web services could … create a loosely coupled architecture in which people could build **new functionality** out of **small, independent tools**.* -- Tim O'Reilly

In writing about blogs, Amazon, Google, and the BookWatchPlus service, Phil Wainewright wrote:

> By exposing their output via APIs or as RSS feeds, all of these participants have plugged into a network of innovation, where the potential outcomes are beyond the imagination of any individual participant. And remember, this is just what people have done for fun, as an experiment. Think what they'll be capable of doing for money, or for power.

The web gets its value not from the smoothness of its overall operation but from its abundance of small nuggets that point to more small nuggets.

» Weinberger, p. x

Make MARC sweat:

With all this information available out there on the web and exposed via web services, ripe for the picking, we simply don't make our MARC records work hard enough. MARC's got just about all the info it needs to do some serious work, to put some added value into the catalog. But, lazy MARC just sits there.

He's got to sing for his supper. Up off your feet. Get to work. Use your author information to add value. Leverage your ISBN to gather additional intelligence. Exploit your LC classification to create visual data maps. Use your imagination.

We don't need to keep stuffing more and more information into the MARC record. There's plenty there now. We just need to make that information work smarter.

There are rich machine-to-machine "conversations" happening every second on the web. And **our ILSs are not participating**. They are deaf and mute. We need to be participants.

Can you imagine that they used to have libraries where the books didn't talk to each other?

» attributed to Marvin Minsky (perhaps apocryphally)

Fast, cheap, and out of control:

Remember NASA's new "fast, cheap, and out of control" philosophy from a few years back?

Actually, NASA got the idea from computer scientist Rodney Brooks and Anita Flynn. Rodney was featured in the Erol Morris documentary of the same name.

His idea was: instead of investing in one large, super-sophisticated, super-expensive, greatly redundant, eggs-in-one-basket Martian explorer, create a "swarm" of many small, independent, simpler, more focused Martian explorers. Instead of one 1,000 lb. robot, send 1,000 one lb. robots.

Make them fast. Make them cheap. Give them their freedom. Exploit their smallness and cheapness. The failure of one, two, or even 40 won't compromise the mission.

Instead of one robot to search for water, atmosphere, rocks, sediment, wind, and so forth, create 30 to search for water, 30 to check the atmosphere, 30 to dig for rocks. Modularize. Specialize. Simplify.

The whole will be greater than the sum of its parts.

Some of the ideas behind fast, cheap, and out of control come from the world of biology, bio-inspired computing, smart mobs, and swarm logic.

The "rules" of such bio-inspired computing are (Steinberg):

- The whole can be more than the sum of its parts.
- Simple rules rule.
- No one needs to be in control.
- Size matters (e.g., not too small; not too big; about 150 for groups of people (Gladwell, p. 184).

What are the lessons of fast, cheap, and out of control in our context?

In 1999, much effort was put into developing a web standard called Information and Content Exchange (ICE). It aimed at allowing major content providers (such as the New York Times and the BBC) the ability to "syndicate" headline content to mid-sized publishers and portals. The specs ran to 154 pages. It cost about $50,000 to implement. (Valdes)

And it died.

Also in 1999 an RSS standard was developed by Netscape. This was also aimed to syndicate content– in this case into the My Netscape Network portal. It was eventually abandoned by them. A "competing" RSS standard was also developed independently.

Despite competing "standards" and the lack of even a unified definition for the RSS acronym (take your pick of "Really Simple Syndication," "Rich Site Summary," or "RDF Site Summary"), these RSS standards were **simple** (two pages) and **open** and **independent**.

And when blogging "hit" in 2003, RSS was positioned for a comeback. RSS ties the "blogosphere" together into a whole greater than the sum of its parts.

> "The lessons for vendors and companies is that simple, 'good enough' technology, such as RSS, often prevails against more-complex approaches that attempt to be all things to all people, even when the good-enough approach suffers from significant limitations in process and scope." (Valdes)

By the way, RSS is a great example of a "syndication" service and technology. Don't wait for your users to come to you. Offer to go to them. It fits in great with some of the ideas I've been exploring for getting our OPACs to "sing."

So, what are some of the lessons to learn here? Make the whole more than the sum of the parts. Think simple and modular and loose. Give up control. Endeavor should give more control to librarians. Librarians should give more control to automated content and services via the web service model.

The Lazy Web, or, "Given Enough Eyeballs" Theory:

> "Given enough eyeballs, all bugs are shallow." (Eric Raymond).
>
> "If you wait long enough, someone will write/build/design what you were thinking about." (Matt Jones)
>
> "I describe a feature I think should exist in hopes that someone else will code it." (Written by Clay Shirky, whom Cathy DeRosa mentioned in her Keynote speech earlier this morning)

These are all quotes from web designers or application developers. We see this in action on our Voyager-L listserv in regard to Access and SQL queries. Someone writes: "I'm trying to get a list of BIBs with XYZ in the 123 field. Anybody written something already?"

Most times, someone pipes up. Sometimes they've already written such an SQL. Sometimes they know enough to write it on the fly for the person. Community in action. Supporting the sentiment behind the quotes above.

My corollaries to those above are these:

- Given open software (not necessarily open source) and enough eyeballs, all bugs will eventually be **fixed**.
- And, given open software and enough eyeballs, **all enhancements will be created, published, and available**.

Now, why would the customers do the work of the vendor, some of you might wonder? I'll give you a couple of good reasons.

- We're desperate.
- We're creative.
- The gift economy is real.

We're desperate. That's true. Who among us hasn't had needs in our libraries that we've been desperate for our ILS to solve. Who among us doesn't do X, Y, or Z, just a **little** bit differently than do most other libraries. If only. If only I could tweak my circ module to s how me just that little bit of information missing. If only I could tweak the overdue notices to calculate just that other little fine we charge in our library. If only.

We're creative. Just look around at all the outstanding presentations to be done in the next three days here at EndUser. And the body of work from previous EndUser meetings archived on SupportWeb. (Remember, that's what got me started thinking last year.) We are a highly creative group of librarians.

When desperate and creative come together, watch out.

Frankly, desperate and creative produce more **incentive**. There is more incentive in the user community to create new tools for our ILSs than there is in Endeavor.

Use librarians are not looking to launch a new product or capture a new market. We're not looking for return on investment. We're looking to solve a need that was bugging us … or roll out a creative new idea we've been burning to get into our users hands.

Look, I've seen Rochester go to extreme lengths to hack the opac.ini and header.htm files in order to put table elements in their WebVoyage so that it matches their library's web pages. I've heard the University of Kansas talk about inserting pre- and post-parsers in front of and behind their LinkFinderPlus products so they could "massage" data before and after LFP touched it because LFP itself didn't have enough internal flexibility. I've seen Hamilton College flail away in trial and error until they could figure out how to write an application from their campus portal that logged on a user to the WebVoyage patron screens. I've seen Southern Methodist University hide HTML code in their MARC records so they could trick Citation Server into putting WWII images alongside metadata.

When desperate and creative come together, what results is this: if it **can** be hacked, it **will** be hacked. That's reality. So, **expect** to be hacked. **Hope** you get hacked. **Build** and **plan** for being hacked.

The gift economy:

What motivates a scientist to "give" away a paper she writes with her latest research? What motivates Rick Prelinger to give away his ephemeral film archives of industrial, educational, and advertising films?

Why does Alan Manifold spend hours answering Access questions on Voyager-L? Not because he's paid to do it. Why does Ken Herold share his code for presenting patron fine information in his campus portal? Not because he thinks he's going to hoard it for a commercial product.

Because status in the community is not measured by how much is hoarded, but by how much is shared.

> No matter what your company is, most of the smart people work somewhere else.
>
> **» attributed to Bill Joy**

"Reputation capital" is accumulated. If you give -- if you add value to the community -- you get an increased reputation.

That pays off in both non-tangible and sometimes tangible ways: recognition, favors in return, self-esteem, and so forth.

Writing about Rick Prelinger's film archives in the *Village Voice*:

> "There's the karma factor. If you give something away for free, you will profit from it in a very real, tangible way down the line. If he gives footage away, more people see it. There's increased name recognition. Everything increases in value. People see his footage, and they come back to him, either to work on a project with him or get more footage – and he profits in the end. Because while you can get it for free if you're an artist, have a cable access show or want to put together a little film festival project, if you want some footage for a high-end project – a film or commercial – then you have to pay for it. The goodwill and generosity has come back to him in spades." (Knipfel).

Besides, developers have social itches. They want to write software that other people will adopt. (Shirky)

Stefano Mazzocchi writes: good ideas and bad code build communities, the other three combinations do not. (Shirky)

Good ideas + bad code = community

Good ideas + community = good code

To do the "algebra," if then follows that good ideas and a community (both of which we've already proven we have in abundance here in the hotel this week) equals **good code**.

Open innovation:

In order for innovation to take place – real innovation – the environment needs to be open. Its not about one person or institution controlling the process or the direction or the developments.

Tim Berners-Lee developed a fairly simple way to encode documents and link them in a user-empowering, user-friendly, decentralized way across the internet. He could not have anticipated all the hundreds and thousands of creative people who came after him, jumped on his ideas, expanded them, and created the web as we know it today.

But he did, as I mentioned before, make some conscious decisions about valuing growth and creativity and openness and innovation in the design of the web.

And what has that environment wrought? Google and Amazon. arXiv.org and The American Memory Project. SETI@home and iTunes.

For some shark species, they need to keep moving to survive. If they stop swimming, they can't breath. (Or, I've also heard it as: they sink. In either case the analogy is sufficient.)

Similarly, ILS vendors need to keep moving to survive. Innovate or die in the market.

And guess what? I'd say so do libraries, if they are to stay relevant to their users. All libraries, but particularly academic libraries and public libraries, need to stay relevant to the Millennial generation now upon us.

Millennials:

- Format agnostic
- Nomadic
- Multitasking
- Experiential
- Collaborative
- Integrated
- Principled
- Adaptive
- Direct

» Abram and Luther. ***Born with the chip***

"Companies that don't innovate, die," wrote Henry Chesbrough in his book *Open innovation* (p. xxvi).

In it, Chesbrough writes about the traditional **closed** model for research and design. And the new, **open** model for research and design.

The closed model says: we must gather (hire) all the best brains in the business and lock them away to develop. The open model says: that's doomed to failure; we must acknowledge that we may only, in fact, have 1% of the best brains in the world on staff. We must figure out how to work with those other 99 brains **outside** our company.

Open innovation. He writes:

> Customers also have important information that can be vital to open innovation. The most advanced, most demanding customers often push your products and services to the extreme. In doing so, they themselves attempt to create new combinations with your offerings as part of the building blocks … People may use your technology in ways you never expected … The [vendor] mind-set was 'We know what they want, and they'll wait until we say it's ready.' … Here [in open innovation], the mind-set shifts to 'Here are some of our thoughts and here's a product that features them. What can you usefully do with it? What can we do to help you do something even more useful?' (Chesbrough p. 56)

Endeavor values its customers. We know that. We seem dozens of our Endeavor colleagues here every year – from the CEO to the customer support rep. They support our user group. They put us on their task forces. They allow us to upload resource sharing records into the Knowledge Base.

But … it's is still a relationship limited in many respects by the structure of the product, by the design intent of the product.

But, it doesn't have to remain that way. At ALA midwinter, Roland Dietz, Endeavor's CEO, "off-handedly remarked that perhaps Endeavor should publish

the code to Voyager and let librarians make their own enhancements, freeing the company to pursue digital initiatives. No one took Dietz seriously, as it's unlikely that there's a library in the country with the resources to work with the enormous Voyager code—and even so, it would still need service and support." (Rogers).

I like that sentiment. Yes, publish the code. Expose and open up the products. I'm not saying abandon it to the customers, but to **leverage** the customers. Yes, the Voyager code is unwieldy and huge. But, equally yes, some customers will do create something innovative and unintended with a disruptive technology.

A thought experiment:

If we wrote an enhancement request today for Endeavor to design a new feature that would suck in an RSS feed for a journal table of contents from a remote server using the ISSN in the MARC record, it would optimistically take at least a year to get into the enhancement schedule, at least another year before getting beta tested, and at least one last year before making it into an official update release.

Not only is creating a completely new ILS unrealistic, but Roland Dietz, Endeavor's president and CEO, suggests that even "incremental functionality improvements [to existing systems] are more and more expensive." And "scalability is not a trivial thing."

» **Pace, Andrew K.**

And even then, users would probably see limitations to its design and implementation and seek refinement.

Oh, and by the way, by the time those three years passed, RSS would be passé and the great new IT killer app would be some as-yet-unthought-of BlueSmokeMagicInfo standard version A-28.

Our ILS vendors just can't chase these innovations. They simply cannot respond to them one-by-one quickly enough.

Instead, Endeavor should design a way to pass "smart" info out of Voyager and suck info back into Voyager. And release **that** kind of upgraded product. Release that idea it into the wild of its customer base.

Don't worry; we'll figure out some cool, interesting, useful, value-added services to hook into it.

Home-grown, home-coded. Third party. Cobbling existing software together. Whatever. We're smart. We're creative. We're librarians. We'll figure it out.

Sure, perhaps Endeavor could even develop some "canned" applications to "hook into" this new capability. Some libraries– both large and small– simply don't have the time, resources, or expertise to do it on their own. But, it should be modular.

So, we're not asking Endeavor to design a "solution" from A-to-Z. To consider all the possible uses and all the possible implications. To imagine all the potential applications.

No. Quite the opposite.

Just enable the **possibility**. Give us an **infrastructure**. A **platform**. A **portal**. We'll take it from there.

Really.

Now, I've talked about unintended consequences as unforeseen and unpredictable. The sister concept to unintended consequencesis disruptive technologies.

Disruptive technologies:

Bower and Christiansen coined the phrase "disruptive technology" in their 1995 Harvard Business Review article.

Sustaining technologies reinforce the status quo, build incrementally upon past developments, and fit within a company's vision and game plan.

Disruptive technologies are often "ground up" developments and are initially dismissed as lacking sophistication or advantages to the current worldview. But they gradually sneak up on the sustaining technologies, eventually overtaking them, turning the status quo on its head and changing the rules of the game.

Disruptive technologies improve much faster than sustaining ones and are far more efficient. They are typically cheaper, simpler, smaller, and more convenient to use.

Some examples include blogs, the web itself, PCs over mainframes, the steam engine, and digital cameras. Perhaps Mp3's and open-source software, too.

Christensen wrote in his *Innovator's Dilemma* book:

> Generally, disruptive innovations were technologically straightforward, consisting of off-the-shelf components put together in a product architecture that was often simpler than prior approaches.

We hear echoes again of simpler is better. Component pieces.

Now, the word "disruptive" has connotations of undesirability. Indeed, in a recent Google pro-or-con ACRL panel in Minnesota, Steven Bell said: "How are we going to harness these disruptive technologies, so we can become a sustainable technology ourselves that will be resistant to all these types of disruptive ones?" (Lederman)

I say, we **want** disruptive technologies. We just want to **expect** them (even if we don't know when they're coming), **plan** for them (even though we don't know what they look like), and **plug them in** when they get here.

To do that, **we** need to be agile. Our **systems** need to be agile. And modular. And independent. And loosely coupled. And fast, cheap, and out of control.

Because things are coming. The web is a platform of knowledge greater than the sum of its parts.

I've spoken a couple of times about RSS feeds. Here's a good example of a disruptive technology. How are our ILSs positioned to use RSS? Because RSS is just one early example of what will surely be many, many more "syndication" protocols yet to come. Syndication of content. The new "Three R's" will be re-use, re-package, re-context. Content will be broken out of its "boxes" and will by flying around the web without our control.

> … there is a tendency in the library world to think of the ILS as a single unified set of functionality. Think of it instead as some data, and a bunch of services built upon that data … The raw data of a web page may be static, but what can be done with that data before it is display to the user is limited only by your imagination.
>
> **» Akerman**

Did anyone see Amazon's search-inside-the-book coming? How about their Statistically Improbable Phrases? Anyone? Didn't think so.

Instant messaging been around for years, but some of us librarians are just responding to it now, judging by recent library-land articles.

Why can't we have a book IM a student: "hey, I've just been returned." Why can't we have our ILS push out an RSS feed for our books: "hey, list of the top 10 most borrowed books so far this semester."

If you tell Gnod.net what authors you like or are currently reading, it will visually map a constellation of other authors that people like you are also reading.

If you give Amazon your zip code, it will tell you the most popularly purchased books in your neighborhood.

If you throw GIS or latitude and longitude coordinates at Google Maps or MapQuest you can get street maps and satellite photos of the location.

Why aren't we taking advantage of these things on the programmable web? We aren't our ILSs **positioned** to allow us to?

I can't possibly imagine all the things coming down the pike. And all the cool uses we can employ to toward in our library systems for our users. Neither can Endeavor. I don't want them to. I just want them to position us with the **capabilities** to respond.

Here's another proof why: **Kurzweil's Law**. Ray Kurzweil says that each technological paradigm shift accelerates subsequent innovation logarithmically, or at an exponential rate of change. Change happens faster and faster. He actually writes of change approaching a "singularity" where all change happens all the time … all at once. (see kurzweilAI.net)

So we've got to be ready.

Conclusion:

Well, the title of this talk promised "a meditation and manifesto." I think I've given enough meditation. (And, we may all need some medication at this point).

So, I'll leave you with the manifesto:

1. If it **can** be hacked, it **will** be hacked. Even if is users have to go to extraordinary lengths. The desire to create an improvement or fix a problem is a strong one. So, don't resist it. Embrace it; plan for it; facilitate it.

2. A passionate, diverse, creative, and oft-times desperate user base above a critical mass will **always** out-power and out-pace vendor-based developers in pushing innovations. Recall Margaret Mead: "Never doubt that a small group of thoughtful people can change the world. Indeed, it's the only thing that ever has.") Thus, use the leverage of this user base. Exploit the gift economy.

3. Innovation thrives in open environments. Indeed, innovation may even **require** openness. **Open up**. And reap the rewards many times back.

4. Don't focus on creating new **features**. Instead, focus on creating new **capabilities**, **possibilities**, and **potentials**.

5. Don't create solutions to problems. Create solution **systems**.

6. Bigger, more detailed, do-everything systems take longer to develop. Focus on small pieces, loosely joined. Allow librarians to link them in infinite, **unforeseen** ways. (Metcalf's Law says the value of the whole network will logarithmically increase with each node added to the network).

7. Create an ILS **platform** that allows for **unintended consequences**. We **want** unintended consequences. You want unintended consequences. Unintended consequences are **good**.

8. Create an ILS platform that expects **disruptive technologies**. We want them, too.

Thanks for indulging me for 45 minutes and letting me "work out the kinks" in public.

I look forward a continuing conversation.

Suggested readings and citations:

Abram, Stephen, and Judy Luther. *Born with the chip.* Library journal, May 1, 2004, pp. 34-37.

Akerman, Richard. *Take control of the library interface.* Science Library Pad weblog, Mar 15, 2005. (http://scilib.typepad.com/science_library_pad/2005/03/take_control_of.html)

Bausch, Paul. *Amazon hacks: 100 industrial-strength tips and tools*. Cambridge: O 'Reilly; 2003.

Bower, Joseph L., and Clayton M. Christensen. *Disruptive technologies: catching the wave*. Harvard Business Review, Jan-Feb 1995.

Brooks, Rodney A., and Anita M. Flynn. *Fast, cheap, and out of control: a robot invasion of the solar system*. Journal of the British Interplanetary Society. 42(10), pp. 478-485. Oct 1989.

Burton, John. *The emergence of web services: web services technology is helping IHEs use data and services more easily, quickly, and cheaply*. University Business, Apr 2005.

Chesbrough, Henry W. *Open innovation: the new imperative for creating and profiting from technology*. Boston: Harvard Business School Press; 2003.

Christensen, Clayton M. *The innovator's dilemma: when new technologies cause great firms to fail.* Boston: Harvard Business School Press; 1997.

Christensen, Clayton M., and Michael E. Raynor. *The innovator's solution: creating and sustaining successful growth.* Boston: Harvard Business School Press; 2003.

Dempsey, Lorcan. *The integrated library system that isn't*. OCLC weblog, Feb 22, 2005. (http://orweblog.oclc.org/archives/000585.html)

Dempsey, Lorcan. *Systems in the networked world*. OCLC weblog, Apr 6, 2005. (http://orweblog.oclc.org/archives/000622.html)

Gladwell, Malcolm. *The tipping point: how little things can make a big difference*. Boston: Little, Brown, and Co.; 2000.

Johnson, Steven. *Emergence: the connected lives of ants, brains, cities, and software*. New York: Scribner; 2001.

Kapor, Mitchell. *How is open source special?* Educause review, Mar-Apr 2005, pp. 72-73.

Kaser, Dick. *One ringy-dingy*. Information Today, Jul/Aug 2004, vol. 21, num 7, p14.

Kaye, Doug. Loosely *coupled: the missing pieces of web services*. Marin County, CA: RDS Press; 2003.

Knipfel, Jim. *Stock footage psyche*. New York Press. Oct 13-19, 2004, p. 34.

Krug, Steve. *Don't make me think: a common sense approach to Web usability*. Indianapolis, Ind.: New Riders; 2000.

Lederman, Doug. *Google: friend or foe?* Inside Higher Ed, Apr 11, 2005 (http://www.insidehighered.com/news/2005/04/11/google)

Morris, Errol (director). *Fast, cheap, and out of control* (documentary film). Columbia Tri-Star, 1997.

Nielsen, Jakob, and Marie Tahir. *Homepage usability: 50 websites deconstructed*. New Riders: Indianapolis, IN; 2002.

OCLC's xISBN project at http://www.oclc.org/research/projects/xisbn

O'Reilly, Tim. *Amazon Web Services API*. O'Reilly Developer Weblogs, Jul 18, 2002. (http://www.oreilleynet.com/cs/weblog/view/wlg/1707)

Pace, Andrew K. *Dismantling integrated library systems*. Library Journal, Feb 1, 2004, pp. 34-36.

Pinchot, Gifford. *A sense of community*. Executive Excellence, 15(10), pp. 19-20, Oct 1998.

Ricadela, Aaron, and John Foley. *New face of e-commerce*. Information Week, Jul 26, 2004.

Rogers, Michael, and Brian Kenney. *ALA Midwinter Meeting Offers Mixed Tech Bag in Boston*. Library Journal; Feb 15, 2005, vol. 130, no. 3, p23.

Roush, Wade. *Amazon: giving away the store*. Technologyreview.com (MIT), Jan 2005 (http://www.technologyreview.com/articles/05/01/issue/roush0105.asp?p=1)

Seidl, F. Andy. RSS: *Disruptive technology hiding in plain sight*. MySmartChannels, Sep 7, 2003 (http://myst-technology.com/mysmartchannels/public/item/12485)

Shirky, Clay. *LazyWeb and RSS: given enough eyeballs, are features shallow* too? OpenP2P.com (O'Reilly), Jan 7, 2003 (www.openp2p.com/lpt/a/3086)

Steinberg, Daniel H. *Swarms and mobs at this year's ETech*. OpenP2P.com (O'Reilly), Apr 25, 2003 (www.openp2p.com/pub/a/p2p/2003/04/25/swarms_mobs.html)

Taylor, Robert. *Value-added processes in information systems.* Norwood, NJ: Ablex Publ.; 1986.

Valdes, Ray. *RSS is a case study in unintended consequences*. Gartner Group document. Oct 7, 2004.

Wainewright, Phil. *Loosely coupled invention*. Looselycoupled.com, Jul 20, 2002 (http://www.looselycoupled.com/blog/2002_07_14_lc.htm)

Weinberger, David. *Small pieces loosely joined: a unified theory of the web*. Cambridge, MA: Perseus Publishing; 2002.