Editorial Board Thoughts: The Checklist

Mark Cyzyk

At my home institution, Johns Hopkins, we have a renowned professor who made a keen insight several years ago. Dr. Peter Pronovost, professor of anesthesiology and critical care medicine, of public health policy and management, of surgery, and of nursing, took note of the fact that many careless errors were occurring in hospital intensive care units, errors that were due not to ignorance, but to overlooking simple, mundane processes and procedures that maximize the safety of patients and increase the likelihood of positive medical outcomes. How to remedy this situation? As is often the case, the simplest solution is the most profound, the most effective, the most brilliant. Pronovost implemented straightforward checklists of processes and procedures for doctors and nurses to routinely follow in the intensive care unit (ICU). There was nothing on these checklists the medical staff did not already know; indeed, they were so basic that what was listed almost went without saying. But the brilliance of Pronovost's insight was that the very implementation of a "just saying" checklist resulted in an immediate and significant improvement in patient outcomes.

So simple. So easily done.

Might we here in the library IT world implement just such a checklist? In particular, I'm wondering whether we might begin to construct a fairly comprehensive checklist of what I'm calling "software genres" for use in libraries.

It doesn't take much insight to see that software packages map to services and that groups of these packages cluster around such services. Such clusters, I'm thinking, are actually genres of software. So, for instance, let's think about a standard library service: a service that fulfills the need of academic institutions to store, archive, and provide access to faculty research. We see software systems emerge to fulfil this need, e.g., DSpace, CONTENTdm, Fedora, EPrints, and Islandora. We can think of each of these systems as the fulfilment of similar sets of requirements and these requirements as being dictated by the satisfaction of a need. These systems individually represent concrete instantiations of clusters of similar requirements. But these systems also cluster among and between themselves insofar as the requirements and the needs they satisfy are similar. They form a genre of software.

Now, we've identified one genre of software useful in libraries: institutional repository software. What other genres might there be? How about the Granddaddy of them all: a software

Mark Cyzyk (mcyzyk@jhu.edu), a member of the *ITAL* Editorial Board, is the Scholarly Communication Architect in The Sheridan Libraries, John Hopkins University, Baltimore, Maryland.

system/concrete-instantiation-of-requirements fulfilling the need in libraries to enable the creation and collection of bibliographic metadata, to provide access to that metadata to library patrons, to enable the management of the circulation of physical and electronic objects mapping to that metadata and to manage acquisition of materials including serial publications? Why it's the venerable integrated library management system, of course! And here we'll find such open-source examples as Koha and Evergreen alongside their commercial counterparts.

Continuing to compile, we'll begin to gather a table of data similar to what's below.

But what good is this? Don't we all already know about these software "genres"? Aren't we just in some sense stating the obvious?

To the extent that these software genres and the software packages classified by each are obvious, this table, this checklist, resembles the checklists that Professor Pronovost presented to doctors and nurses in the ICU. Those doctors and nurses certainly knew to wash their hands, to properly sterilize around an area of catheterization, etc. And if they were about to overlook a hand-washing or sterilization, the checklist brought this need and requirement immediately to their attention. So too we might be aware of, say, the fact that many libraries implement systems to manage visual resources, electronic images. We may work in a library that does not. Yet scanning the list below we'll see that it is a need with requirements that cluster to such an extent that a genre has developed around it. We may say to ourselves, "We too have a significant collection of electronic images that we've just been putting into our local repository, but maybe the repository is not the right place for this content, maybe we should look into these other systems designed specifically for storing and serving up images?" More, suppose you work in a really small and resource-constrained library, yet one that has an important archival collection of materials related to local history? You now scan the table below and determine that there is indeed a genre of archival information systems. "Maybe we should set up an instance of ArchivesSpace to facilitate the cataloging and provision of access to our archival treasures?" Maybe you should!

Now some of these systems can do double duty. DSpace, for instance, could be used as a serviceable image management/data management/ETD management system. But cramming so many disparate kinds of content into a single system and expecting it to behave like a bona fide image management or data management or ETD management system is probably a bad idea. The whole reason these genres emerge in the first place is that they satisfy the idiosyncratic needs of various types of content; their functionalities are tailored to the types of services one would want to provide surrounding such disparate types of content. Using the right tool for the job is surely a best practice if ever there was one.

That said, some of these software packages, by design, cut across the different genres in much the same way that, say, the music of Ryan Adams cuts across the genres of rock and alt country or the novels of China Mieville cut across all manner of literary genres. Islandora provides a good example of this. It can be used as a repository, but its various "solution packs" allow it to be

expanded to serve well as, for example, an image management system. Islandora, interestingly, is itself built on two of the software packages listed below—Drupal and Fedora—combining the strengths and functionalities of both, resulting in a genre-busting (genre-extending?) software package.

Viewing software packages as types of genres is useful insofar as, when it comes time to assemble requirements for a new project or service, thinking of existing software systems as concrete instantiations of such requirements speeds the accomplishment of this task. Rather than begin to gather requirements *ex nihilo*, we should be thinking, "Here are some prime examples of actual working systems that fulfil the satisfaction of needs similar to ours. How about we compile a list of them, install a few, and see what they do? Maybe there is something here that fits our specific need?"

So simple. So easily done.

Check!

Software Genre	Software Packages
ILMS	Koha Evergreen Aleph Millennium Polaris Symphony Horizon Voyager
Indexing and Discovery	Blacklight Summon WorldCat Local Ebsco Discovery Services Encore Synergy Primo Central LibraryFind eXtensible Text Framework
Institutional Repository	DSpace Fedora EPrints Greenstone CONTENTdm Islandora Digital Commons
Image Management and Access	MDID ARTstor Shared Shelf Luna
Publishing: Journals	Open Journal Systems

	Annotum Digital Commons
Publishing: Monographs	Open Monograph Press BookType
Online Exhibition Software	Omeka Pachyderm Open Exhibits Collective Access
Archival Information System	Archon Archivists Toolkit ArchivesSpace ICA-AtoM
Faculty Research Portfolio/Showcase	BibApp SciVal
Personal Bibliography Management	Zotero RefWorks EndNote Mendeley
Web Content Management	WordPress Drupal Joomla
Data Management	Data Conservancy Archivematica Dataverse
ETD Management	Vireo OpenETD
Wiki	MediaWiki Confluence Sharepoint Tiki Wiki
GIS	ArcGIS Server MapServer MapStory