Statistical Improvements for v1.6

Plan of Attack

I've begun the process of putting in place a series of modules and core changes to DSpace to support the inclusion of external statistics and reporting systems. Much of this work is coming from code donations from @MIRE and represents components of @MIRE products with we feel will improve the health of the DSpace ecosystem by exposing. Most specifically, by exposing and donating these components, we seek to show "how" modularity needs to be modelled and apporached not only in the future DSpace+2.0, but more immediately now in 1.6. --Mark Diggory 04:25, 3 July 2009 (EDT)

The breakdown of the projects is as follows:

UsageEvent improvements:

Adjustment of the UsageEvent API to support exposing richer detail about the DSpaceObject in the implementations please see:

http://jira.dspace.org/jira/browse/DS-243

GEOIP, SOLR and DNS modules

Creation (and/or publishing into Maven) of Support Modules for features that may be common in more than one statistics/reporting implementation: Please see.

http://maven.dspace.org/release/org/dspace/dnsjava/dnsjava/2.0.6/

https://scm.dspace.org/svn/repo/modules/dspace-geoip (Self installing library for supporting GeoIP lookup)

https://scm.dspace.org/svn/repo/modules/dspace-solr (Self deploying SOLR webapplication pluggable into dspace distributions)

SOLR Based Statistics Logging and Reporting

```
https://scm.dspace.org/svn/repo/modules/dspace-solr-stats
```

(Soon to come: Self deploying Solr Client and Statistics Loggers for above Solr Instance)

```
https://scm.dspace.org/svn/repo/modules/dspace-solr-reporting
```

(Soon to come: Self deploying Solr Client and Statistics Report/Query API for above Solr Instance)

DSpace+1.6 ServiceManager Support

Finally, We are working on a slimmed down version of the DSpace+2.0 ServiceManager that should allow dynamic registration of Asyncronous services like StatisticsLogging and StatisticsReporting into DSpace without the need for explicit configuration via dspace.cfg/PluginManager.

https://scm.dspace.org/svn/repo/modules/dspace-services (Soon to come: Service API and Utility classes to support dynamic registration of services)

Raw descriptions of ideas presented so far : please elaborate

"Page Views" versus Downloads

- In my opinion (Tim Donohue) , there's really two main types of practical statistics that may be of interest to different individuals using a repository
  - *Page Views* (i.e. individual visits to individual splash pages within a repository) - From experiences at U of Illinois, most of our individual faculty members or departments don't really care about how many people out on the web visit their item "splash page". These "Page Views" however are still usually of interest to the Repository Administrators, as it can sometimes help them determine how users are using the site and what they are visiting, etc. Reports about Page Views could be generated by something like Google Analytics (or AWStats), as that is what both do quite well – Tim Donohue 10:00, 29 May 2009 (CDT)
  - *Downloads* (i.e. download counts of individual files in the repository) - From experiences at U of Illinois, this is what our faculty/staff /departments really want to see from a repository. They are very interested in reports of actual downloads over time, including "Top 10" lists of downloads at Community/Collection levels over different time frames (e.g. "Top 10 Downloads in ___ Community in last month /year/overall"). These download reports would be much more difficult to generate using something like Google Analytics, as Google Analytics has no concept of the hierarchical structure of a given repository (i.e. Community / Collection / Item / Bitstream) – Tim Donohue 10:00, 29 May 2009 (CDT)
- I agree with Tim, and would only add this: We have had numerous requests to collect and present information about referring site – in other words, whether the link to the DSpace item or bitstream is coming from Google, Google Scholar, Yahoo, or wherever, so we've begun to capture that information. Jim Ottaviani 13:00, 29 May 2009 (EDT)

    - Just to back up what Jim has said, U of Illinois has also had numerous requests for information about where downloads are coming from. This includes wanting to know if they are coming from Google, a direct blog link, etc (like Jim mentioned), but also where they may be coming from in the world (on-campus, from the USA, from elsewhere in the world, etc.). Again, the requests we've heard all have to do with generating these reports based on *downloads* and not based on "page views" – Tim Donohue 12:30, 29 May 2009 (CDT)
- *Yes, agreed. Its always the downloads that people after, but once you have an implementation to support downloads, adding hooks for page views is trivial and the door is open to many other reports as well.* --Mark Diggory 14:12, 9 June 2009 (EDT)

COUNTER compliant usage statistics

- PIRUS final report (research for Counter compliant statistics) http://www.jisc.ac.uk/publications/documents/pirusfinalreport.aspx --Bluyten 06:06, 29 May 2009 (EDT)
    - From this report it seems that monthly file downloads per item are the requirement for COUNTER-compliance.

leverage Google Analytics instead of collecting our own data

- *I believe Google Analytics can't track direct bitstream downloads very accurately.. can anyone confirm/discuss further?* --Kshepherd 17:21, 28 May 2009 (EDT)
- *I am not a fan of using Google Analytics as it places us dependent on an external service for statistics collection.* --Mark Diggory 17:25, 28 May 2009 (EDT)
- *The great thing about GA Stats is that many institutions already have a backlog of GA Stats history, and that they generally implement the "counting" in the same way (script in the footer of the page). Important drawbacks are that it doesn't take into account bitstreams that are directly downloaded without accessing the web pages (for example, direct download through google). Also, it's impossible to aggregate data directly from Google analytics: most popular items per community, collection, ... It would be great if the ultimate solution will allow to "draw in" google analytics data, and let the repository administrator decide whether he or she wants to visualize GA stats data, or internally measured data. GA stats data might always come in useful, if people want to compare their data to institutions running another platform (eprints or fedora), if those institutions are also running GA stats.* --Bluyten 06:14, 29 May 2009 (EDT)

offline management reports using a nightly copy of observations

- *when a dataservice is available to query against for statistics, providing reports will become independent of being "offline" or "cron" jobs, we should be striving for a new approach using existing reporting tools and datastores for querying against. Certainly NOT perl scripting.* --Mark Diggory 17:41, 28 May 2009 (EDT)
- *Fair comment, but how scalable are these new approaches? Can we feel safe running adhoc live queries when we have >1000000 usage events stored? The length of time taken to generate a single report is the only thing stopping my own stats systems from being real-time.* --Kshepherd 17:51, 28 May 2009 (EDT)
    - *I feel that our stats engine is up to this challenge.* --Mark Diggory 14:13, 9 June 2009 (EDT)

A Case for Google Analytics

I'll address the points already raised (and try to update if more are added), and provide my own opinion on this matter.

It's correct to say that if you simply follow the most basic integration guidelines provided, then Google Analytics will only track HTML page views (and then, only those that have the tracking code integrated!). However, the data collection API that is provided is capable of tracking more than simple page views. It's a relatively simple task to add a javascript event to the bitstream download links, so that when a user clicks on them, that download request is tracked.

That won't help the case of a direct link to the bitstream from outside the repository (although as they don't have proper persistent identifiers / URLs, that possibly should be discouraged). If you need to track such downloads, then the most obvious means would be to detect referers that are outside of the repository, and deliver a 'your download will start automatically' page in place of the actual bitstream - which would then include the GA tracking code. Alternatively, it's just a service API, and it wouldn't be that hard to construct a call from the Java code directly (although if you bypass the ga.js javascript library, you may be exposed to the API changing).

Having dealt with statistics gathering in a variety of ways (direct database logging, offline log analysis) for some moderately high volume sites, I'm aware of how problematic it can be. The volume of data generated is huge, and has scalability problems in storing, parsing and reporting - for either style of statistics gathering. Then you have to deal with determining and removing robot / invalid accesses. Recognising when a user may have double-clicked on a link.

And the flipside to having a(n external) service API for collecting statistics data is that it already provides for tracking events - AJAX operations, Flash controls, redirects to other sites, can deal with local / content caches or content delivery networks (and that could be an important point going forward with DuraCloud). All things that you would need to add additional javascript calls and local service collection points to cope with for local statistics gathering.

Historically, there has been a problem with Google Analytics not being able to provide statistics integrated with the repository / to non-registered users, but now that there is an API for retrieving data from GA, that is an issue that can be solved.

My personal view is that we have enough to deal with in terms of delivering repository functionality, and making the repository itself scale, to really not need to be dealing with all the problems that come with statistics gathering / generation. Moreover, if you do keep the statistics internally, then the scalability of your repository will be compromised by the requirements to provide statistics.

If you look at the reasons why Google purchased Urchin, provide Analytics for free, their capability to provide scalable services, size, stability and general ethos of the company, then I would consider the risks of relying on them to be much lower than the cost of maintaining your own statistics. That said, I would still advocate a framework and collection of data points within the repository for supporting integrated reporting (ie. downloads on the item page), where that data can be supplied from either Google Analytics, or an internal data collection service.

--grahamtriggs 12:03, 29 May 2009 (BST)

The importance of "Hooks" for specific data collection points and use of a statistics logging backend should not be undervalued. We have an opportunity here to gather critical data that can be used to facilitate a richer browse, search and viewing expereince. Throwing it off to GA and suggesting its out of scope of "Repository Concerns" is a bit heavy handed IMO. --Mark Diggory 14:06, 9 June 2009 (EDT)

Also consider that many IR administrators are probably expecting this addon to work with the local data they've already collected – people who have been requesting statistics have been sitting on years worth of logs, but do not necessarily have any accurate historical data with GA. (I realise that the logs aren't in the UsageEvent format we want to use now, anyway, but converting them is not a big deal). I think there would have to be more of these cases than IRs having years of GA data, but no logs saved. --Kshepherd 22:02, 9 June 2009 (EDT)

Technical Requirements

- Separation of Statistics implementation from core dspace-api.
- Modularization of Statistics as Addon to JSPUI and XMLUI
- UsageEvent based event recording to standalone service in place of dspace.log analysis.
- Independent Persistence of datasets separate from dspace database.
- Queriable Service API for Accessing Statistics.

Existing 3rd party statistics systems

Please add your statistics reports/tools here – if reports are not publicly accessible, screenshots or video demonstrations would still be great to see.

- Minho RepositoriUM Statistics (example item-level report)
- University of Rochester Statistics (example homepage 'hit counter' report)
- LCoNZ Statistics (example full site report) (example collection report) (example item report) (example author report)
- @mire Statistics Addon Module (example collection report) (example item report)
- Stuart Lewis's Blog (example Google Analytics 'hit counter' widget)
- Simplestats (A subversion repository, username: anonsvn password: svn) (example community report) (example collection report)
- Example of U of Rochester stats in DSpace 1.5 XMLUI (IDEALS @ Illinois):
    - Top 10 Downloads on Homepage
    - Example Community Download Stats (see bottom of page)
    - Example Item Download Stats (see bottom of page)
    - Example Community Download Report
- Leiden University Statistics (example item-level report; click on "More statistics for this item") (screenshots of admin stats:)