Enhancing Statistics: Google Analytics and Visualization APIs

Overview

Usage statistics have been an important topic in the repository community for some time. From Minho's DSpace additions, through Interoperable Repository Statistics, to @mire's Solr based contribution do DSpace 1.6, there have been many approaches to providing statistics.

One technique that has been used in a few places is to set up a Google Analytics account. These have several advantages – free, independent of repository (and it's architecture), proven scalability, excellent tools for visualizing the data.

But it has historically had its problems too – doesn't understand the structure of the repository (for displaying totals or top views/downloads for an arbitrary grouping of the content), doesn't track downloads without additional work (or those directly linked from search engines), and the reports are locked behind an authentication wall and can't be opened up to general repository users.

With the [April 2009] release of an API to retrieve data from Google Analytics, that has changed. Data that has been calculated in Google Analytics can be pulled back into the repository, so that it can be viewed within context, and by anyone that can access the repository (or not, depending on implementation).

This presentation shows how Google Analytics can be integrated with the repository, techniques for capturing data that wouldn't normally be available with Analytics, and making the data comprehensible through visualizations.

Whilst the implementation presented here was initially conceived using a DSpace repository, the techniques can be replicated in any repository software. Further, the visualization methods are independent of the analytics data themselves, so can be adapted for other sources of data.

Integrate analytics data into the repository

In order to retrieve data from Google Analytics, you form a query asking for a set of metrics (pageviews, unique visitors, etc.) broken down by various dimensions (URL, page title, country, city, etc.). This can be requested over a defined time range, filtered to a subset of the data and sorted in various ways.

For example, in a DSpace repository, item pages would have 'handle/1234/567' in the URL, and the bitstreams for the item would have 'bitstream/1234/567' within the URL. If you want to retrieve the total number of views for an item, or downloads for that item, then you would request the number of 'pageviews', filtered to a 'pagepath' containing 'handle/1234/567' or 'bitstream/1234/567'.

If that item had multiple files, and you wanted to retrieve the download counts for each file separately, you can do this in one request, simply by adding the 'pagepath' as a dimension – you would then get a row for each path (each path would be a different file), with it's 'pageview' (download) count. Applying a sort on 'pageview', you can show the list in order of popularity.

Sometimes with query parameters, the path may be recorded in multiple formats. This can make it harder to see overall totals for item views or file downloads in the Google Analytics interface – but it's something we can easily cater for when integrating the data into the repository. When we parse the data from the API, we look for the common elements in the page path (ie. handle/1234/567), and accumulate the metrics (views) returned for them. We can also filter out anything that we don't want – so when we need data for an arbitrary set of items, we can blindly request data for all of them, and ignore the values that don't belong to this set.

By careful selection of dimension and metric combinations, and building up the techniques described above, this presentation will show how we can create statistics pages for an item that will show the number of views for that item, the downloads of attached files, the recent activity by month, and the countries and cities where those accesses are coming from.

And for a grouping of multiple items (ie. a repository collection), the views of a collection landing page, the number of item views and file downloads for contents of that grouping, the recent activity, top item and top file download lists, and the geographical breakdown.

Bring statistics to life with Google Visualizations

Simple tables and number displays can be unappealing and hard to understand. It's much easier to see relative performance with bar and pie charts, or trends with line or area charts.

In 2008, Google launched a visualization api – a collection of charts, graphs, maps and more. These are easily integrated into any web page, by using Javascript to create a dataset and calling the visualization to render into a selected element on the page.

The data returned by Google Analytics is a natural fit to become a dataset for the visualizations. In the examples provided (see references), bar charts are used to show overall item views and file downloads (or the top item / top download views for collections and communities – if you click through to 'view all'). A line chart is used to plot the trends of activity of the recent six months.

The nicest example of synergy is in the country / city names returned by Google Analytics for the geographical breakdowns – which are immediately recognised the Geo Map visualization. So, on each page of the example, there is an interactive flash map showing the number of accesses from each country.

Whilst there is a natural use for these visualizations with the Google Analytics data, they are separate – so these techniques can be used to enhance the display of statistical data retrieved from other sources too.

Capturing more information in Google Analytics

The other drawback to Google Analytics is its use of Javascript for tracking page views. Whilst it's arguable that the number of non-Javascript users is small enough to not significantly impact the statistics, it is a problem for non-html content – ie. file downloads.

Typically, you would get round this by adding javascript events to download links in the repository. However, this only works for downloads that come from a page in the repository, if a user accesses a PDF through a direct link from Google, that won't be counted.

A better technique would be to have the server trigger the logging in the case of file downloads. Primarily aimed at the mobile device market, Google provides an example of triggering a pageview from the server side (via a simple HTTP request). From this starting point, we will see how to log a file download from the server, accurately capturing all downloads – regardless of whether the user came directly from a search engine or not.

What's more, Google Analytics has an additional feature called Event Tracking. This was envisaged mainly as a way of tracking activity within an interactive page element (ie. Ajax or Flash). Events consist of Categories, Actions and Labels, and can be assigned arbitrary values.

These events can also be generated server side, using the same techniques as the download tracking. This gives us a powerful means of getting other interesting data out of the repository application and into Google Analytics – where it can be manipulated by their UI, or retrieved back into the repository via the export API.

For example, we can use the event tracking to log how many bytes are being transferred out of the repository by each request. In doing so, we can see the overall (outbound) bandwidth taken by the repository over a period, but we can also break it down to the individual paths – showing how costly each page (or element of a page) is.

References

1: Enhanced Statistics, Open Repository

http://www.openrepository.com/products/enhanced-statistics

2: Google Analytics API

http://code.google.com/apis/analytics/

3: Google Visualization API

http://code.google.com/apis/visualization/interactive_charts.html

4: Example Implementation within a DSpace repository

Entire repository: http://www.e-space.mmu.ac.uk/e-space/displaystats
Community: http://www.e-space.mmu.ac.uk/e-space/displaystats?handle=2173/1012
Collection: http://www.e-space.mmu.ac.uk/e-space/displaystats?handle=2173/552
Item: http://www.e-space.mmu.ac.uk/e-space/displaystats?handle=2173/7889