CHAPTER 17

Putting It All Together

So far, we've looked at the many kinds of data you need to monitor, both on your site and elsewhere on the Internet. It's a deluge of data, and you can spend your entire day analyzing it, leaving you little time for anything else.

The final step in complete web monitoring is putting it all together. This is hard work: while many organizations are now recognizing the need for complete web monitoring, few understand how to assemble it into a single, coherent whole.

But assemble it you must. Silos of web visibility lead to the wrong conclusions. Imagine, for example, a website redesign. If conversions go down following the design, you might assume that the design was a failure. If your service provider had a problem that made pages slow at the same time, however, you can't tell whether the redesign was at fault or was, in fact, better.

With many moving parts behind even a simple website, having one part affect others is commonplace: conversions are down because people can't understand a form field; a page isn't usable because third-party content isn't loading; a synthetic test is pointed at part of the site visitors never frequent; a survey gets no responses because the SSL certificate is broken.

In the end, you need to tie together performance, usability, web analytics, and customer opinion to get a complete picture. Here are some of the reasons for doing so.

Simplify, Simplify, Simplify

With so much data at your disposal, your first instinct may be to collect and aggregate as much as you can in hopes that some sort of pattern will emerge, or that someday, it will all make sense. Thankfully, the places & tasks model should protect you from wasting your time chasing useless numbers. By using the questions that you've written down as a metrics blueprint, you'll go on missions to find relevant numbers within any arbitrary tool that might provide it. This differs greatly from using a tool's number to try and figure out what's important to you. Places and tasks helps you avoid information overload by seeking out only the metrics that matter (and avoiding those that distract).

Drill Down and Drill Up

Many of the tools we've seen start with an aggregate view of what's happening on a site. This makes sense, because with so much happening on a website, you should begin with a view of traffic as a whole.

When viewing aggregate data, however, you want to drill down quickly to individual visits. If you're seeing a pattern of clicks on a page, you may want to replay mouse movements for that page to better understand what's happening. If you're seeing errors, you will want to replay visits to see what visitors did just before arriving at that page. When you notice a performance problem, you will want to analyze individual visits that experienced the problem to see what caused it.

You will also want to drill up. Given an individual visit, you'd like to know what was happening to the rest of your visitors. When someone calls the helpdesk with a web problem, the first question should be, "Are others affected?"

Drilling up is part of the diagnostic process. If you have a visitor who's experiencing slow page loads, you want to analyze all other visitors who experienced similar conditions to see what they have in common. For example, if 10 percent of all users are having very slow page load times, analyzing all of their visits together can show what they have in common. You might notice that they're all in the same city, or all have the same browser type, or all bought more than $500 in products.

Drilling up means creating a custom segment based on attributes of an individual visit (for example, "All visits from Boston on Sprint using Firefox"). It may also mean moving across several data sources in the process.

Visualization

Even if you're not trying to drill down or up, you may want to see everything on a single screen. This is easier to do than actually tying individual visits to aggregates, because you can just pull aggregate information from many sources and display it on a single chart with the same time range. Typically, graphs and data will be grouped by time, site, network, or region, since these attributes are universal across almost all kinds of monitoring.

For example, you could display performance, community mentions, page views, and conversion rates over time. You could also show them by region, network, or web property for a given time period (e.g., the last week). You'd quickly see correlations and be able to compare traffic sources.

Segmentation

As we've discussed, web monitoring isn't just about collecting visit data, it's about segmenting those visits in useful ways to see what's working better or worse. Many of the dimensions along which you want to segment are contained in a single tool—you can segment goals by referring sites within an analytics tool directly. But what if you want to segment data in one tool by a dimension that's only available in another?

Let's say you'd like to know whether people who submit feedback on your site through a VOC survey are less likely to buy something. You can approach this in several ways. You can store both VOC and analytics data in a central data warehouse, then analyze it later. Or you can modify your VOC tool to send a message to the analytics service that marks the visit as belonging to someone who has been surveyed, then segment conversion rates by this marking.

Measuring the impact of page load times on conversions is another obvious example that requires cooperation between several systems. You may want to answer questions that require even subtler segmentation—for example, do people who scroll all the way to the bottom of the page have a faster page load time?

To do this after the fact, you need to compute the new segments from individual visits and pages that your systems monitored, which means storing all visit data somewhere.

Efficient Alerting

Another reason for consolidating your monitoring data is alerting. You can't watch everything all the time. You'd like to feed monitoring data into a system that can baseline what "normal" is and let you know when something varies.

Sometimes the first clue that there's a problem is a drop in sales—indeed, anecdotal evidence suggests that sales per minute is one of the key metrics huge online retailers like Amazon look at to determine whether something's wrong (though none of them would confirm this for us).

Complete web monitoring implies complete web alerting, bringing all of the problems detected by all of your monitoring tools into a single, central place where you can analyze and escalate them appropriately.

Getting It All in the Same Place

The amount of consolidation you're going to get depends on how much work you're willing to do and how much you want to invest. You can buy a solution from a single provider that collects many kinds of data, or you can build your own data warehouse and generate your own reports. Even if you don't want to invest a lot of time and effort, you can still herd all the web monitoring data you're collecting into a few sensible views.

Unified Provider

The simplest way to get a consolidated view is to use a vendor that provides it. Since much of the data we've seen is collected in the same way—through JavaScript—it makes sense that vendors are broadening their offerings to include a more comprehensive perspective.

Here are some examples:

- Omniture's Genesis offering is a central tool for aggregating data, and many third-party products can push their information into Genesis, where you can analyze it. You can also use Omniture's segments to query Tealeaf sessions.
- Gomez and Keynote offer both synthetic testing and JavaScript-based RUM in a single service.
- Google Analytics includes a site overlay view that shows WIA click analysis.
- AT Internet combines many WIA and analytics functions in a single offering.
- Tealeaf captures WIA data for analysis and replay, and also captures page performance information (RUM) and allows you to specify goals (analytics).
- ClickTale records page timings of the visitors whose interactions are recorded, combining RUM with WIA.
- Analytics provider WebTrends is partnered with community monitoring tool Radian6.
- Coradiant is primarily focused on RUM and user performance, but can extract custom fields (such as checkout price) from pages for segmentation later. It can also export page records to analytics packages as a form of inline collection.
- Clicky includes FeedBurner and Twitter follower information within its display.

We expect to see considerable consolidation and collaboration among vendors in this manner as a complete web monitoring model is more broadly adopted. While unified provider tools often cost more money, they may have a smaller impact on page loading because they require only a single monitoring JavaScript on the client.

Data Warehouse

The other way to get a complete picture is to paint your own. Web analytics is really a subset of a much broader category known as business intelligence (BI), which is dominated by firms like IBM (Cognos), MicroStrategy, and Oracle (Hyperion). Companies use BI to crunch numbers, from employee performance to which products are selling best in which stores.

You can think of BI as a gigantic version of the pivot tables you see in spreadsheets, where any piece of information can be segmented by any other. BI tools query data from a data warehouse for analysis. The data warehouse is essentially a very large database that's specially optimized so the BI tool can segment and query it along many dimensions.

Every time you make a purchase with a grocery store loyalty card, book a hotel with a frequent flier card, or use a bookstore discount card, data about your transaction winds up in a data warehouse for analysis by a BI tool.

If you put web monitoring data into a data warehouse, you can use a BI tool to generate custom reports that span multiple data sources. If you load aggregate data (visits per hour, comments per hour, performance for an hour) into the BI tool, you can get aggregate reports. But the real value of a BI tool comes from loading individual visit data into the tool, which allows you to segment it in new and creative ways.

Merging visitor data: The mega-record

Before you can load anything into a data warehouse, you have to understand the structure of the information you're inserting.

Imagine that you have a visit to your site, and you're running five kinds of visitor tracking. After that visit, you have four different vantage points from which to consider the visit:

- The web analytics system knows each page that was visited, as well as all the analytical metadata about pages and the visitor (such as browser type, ad campaign, and whether the visitor abandoned a transaction). It also knows which online community referred the visit, if any.
- The WIA system knows about pages it watched, such as a home page's clickmap and a checkout form's field completion.
- The RUM appliance knows the performance of each page the visitor saw, as well as any errors that occurred. We have excluded synthetic data from this list because it doesn't offer per-visit information; it's used for aggregate views instead.
- The VOC tool keeps a copy of the survey the visitor completed, along with any per-page feedback rankings.

Some of this data, such as browser type, applies to the visit as a whole. Other data, such as RUM performance or VOC ranking, applies to individual pages. There might even be information on individual objects (such as an embedded Flash element) and actions taken on those objects (such as clicking play or pause). You can even think in terms of the visitor, because this visit may be one of many that the visitor has made to your site.

In the end, the data you can collect is hierarchical. Different metrics from different perspectives apply to different parts of this hierarchy, as shown in Table 17-1.

Table 17-1. Metrics from four kinds of monitoring at various levels of the user/visit hierarchy

Hierarchy	Analytics data	WIA data	RUM data	VOC data
User	Username, number of visits	Where visitors most often click on arrival	Percentage of visits that have errors	Number of surveys completed
Visit	Number of pages seen, whether a goal was achieved, entry page, browser	Events that happened during the visit	Total network time on session, total bytes downloaded, number of errors encountered	Whether the visitor provided feedback this visit
Page	Time on page, previous page visited	Mouse movements on page	Total page load time	VOC page vote
Object	Whether component rendered in browser	Whether component was clicked	Object size	-
Action	In-page events or labels	On-page movements and clicks	Refresh latency	-

This user-visit-page hierarchy is just one of many that are useful when defining how to structure data on web activity. We've found it works well for problem diagnostics, helpdesk, conversion optimization, and replay. There are many other hierarchies, however. For example, a SaaS company that has to report SLA performance might have a hierarchy that organizes information first by each subscribing customer, then by section within that site, then by page. This would make it easier to provide SLA reports.

Choosing the right hierarchy for a data warehouse is one of the reasons BI is such a complex topic: how you organize data affects how efficiently you can analyze it afterward.

Once you have specified a hierarchy, you need to determine the unique elements of visit records that you'll rely on to assemble the many data sources. For example, all four monitoring tools can record three basic facts: the time of the visit, the visitor's IP address, and the URL of the page the visitor was on. You can use this information to merge the data from each system into one mega-record. Then you can pick a metric at any level of the hierarchy (i.e., the VOC page-level information on whether a visitor provided feedback on a particular page) and tie it to another metric (i.e., analytics page-level information on whether that visit yielded a conversion).

Unfortunately, relying on only time and IP address is seldom enough. You need to consider many other factors, such as unique cookies or data within the DOM that all tools can collect.

Taken together, the hierarchy you choose and the rules you use for creating a gigantic mega-record of visits make up the schema for your data warehouse. Figure 17-1 shows one way this can happen:

1. A visitor surfs the website, visiting four pages. Each page has a session cookie that identifies it as part of the same visit, as well as a referring URL and timestamp from

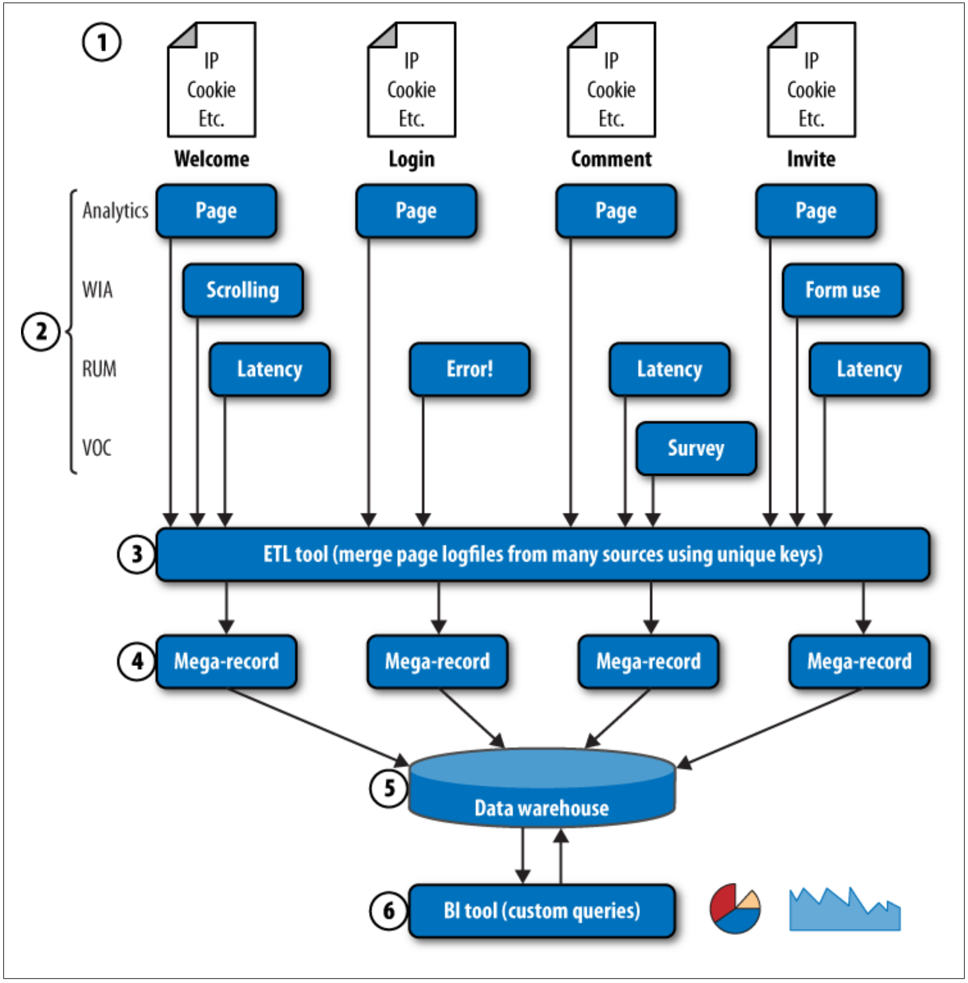


Figure 17-1. Pulling together disparate data according to common unique keys

the previous page, from which we can understand the sequence of the page requests.

2. The various monitoring tools collect their information. While analytics and RUM collect data on every page, some other monitoring tools may only collect information on certain pages—a survey on the comment page, for example, or form analytics on an invitation page.
3. At regular intervals, log data from all of the tools is merged according to the unique session cookies. This is a process known as *Extract, Transform, and Load* (ETL).
4. The result is a mega-record of each page.
5. The records are stored in a data warehouse, where they're indexed in useful ways (by time, geography, page URL, or visitor name, for example).

6. You can use a BI tool to query the data warehouse for aggregate or individual data.

ETL: Getting the data into the warehouse

ETL is the most complex part of this operation, since it requires merging many sources of nonstandard data for further analysis and dictates which kinds of analysis you'll be able to do. The ETL task is performed by a software package that gathers, modifies, and inserts data into a data warehouse according to the hierarchy and schema you provide, using certain unique keys to merge sets of data together.

As its name implies, ETL has three steps:

1. *Extracting* data from various sources (analytics, RUM, WIA, VOC, and communities) through RSS feeds, export tools, or APIs. Your ability to do this will depend on the product you choose.
2. *Transforming* data by normalizing it (for example, one tool may measure performance in milliseconds, another in seconds) and associating disparate data sources (for example, determining which VOC feedback goes with which analytics data by looking at unique cookies the two data sources have in common).
3. *Loading* data into the data warehouse, which means writing it to some kind of storage in a structured fashion and indexing it according to the dimensions by which you want to group and analyze.

Once the data is in the warehouse, you can use the BI tool, spreadsheets, or even advanced analysis tools like Maltego (*www.paterva.com/maltego/screenshots/*) to generate reports. Merging records into a mega-record like this is the only way to perform drill-down and drill-up across multiple data sources.

The example above deals with individual user records. If you're not concerned with drilling into individual records or segmenting based on visitor data, you can still put aggregate data into your data warehouse. For example, you might store site latency, conversion rate, number of community mentions, and number of VOC surveys completed every hour, then overlay them. While useful, this won't let you ask many questions of your data.

Be warned: BI is a complex field, and your organization probably has other priorities for its very expensive BI systems. Even if you get access to the BI tool and a data warehouse in which to store every visit, you're effectively building a new analytics tool of your own.

The Single Pane of Glass: Mashups

You can get a long way by simply displaying many information sources on the same screen. Where the BI approach first merged data, then reported it, this approach first generates a report within each monitoring tool, then merges those reports into a single view.

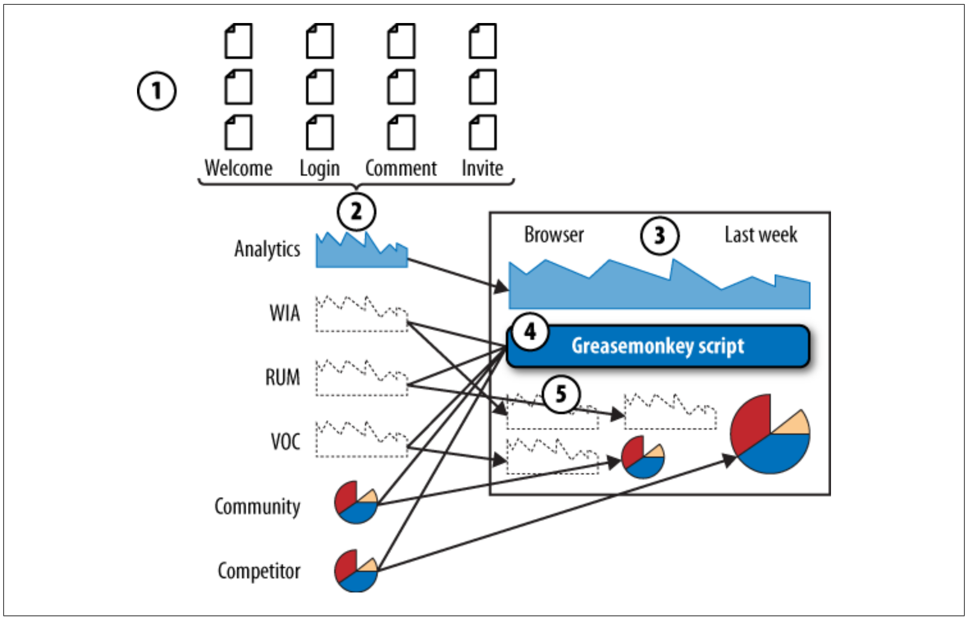


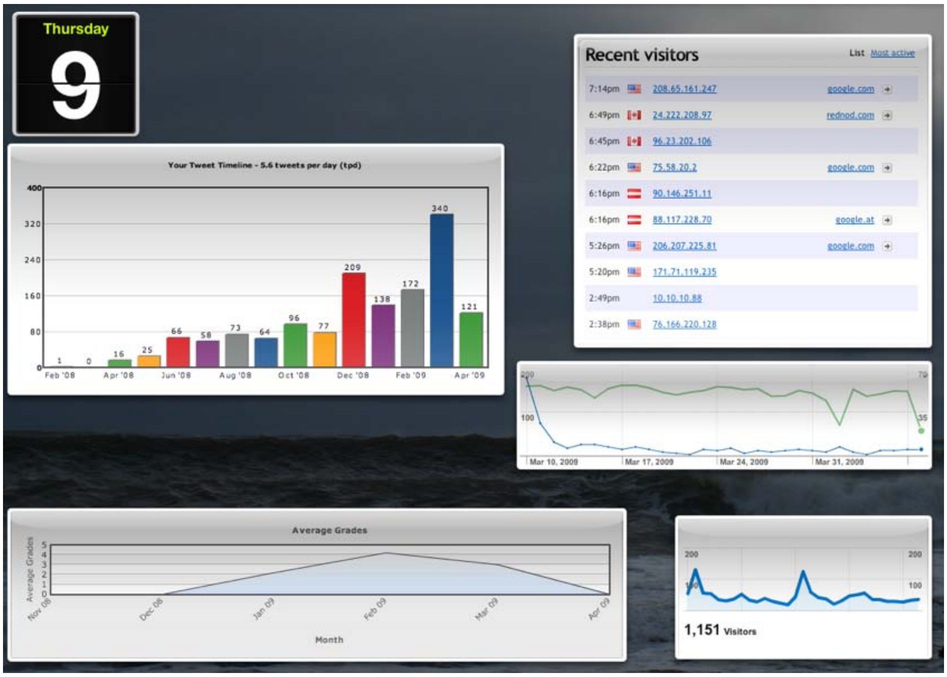
Figure 17-2. Using Greasemonkey and a browser mashup to display data from many sources at once

You won't be able to create your own segments easily—rather, you'll have to settle for the segmentation your tools offer—but you will be able to see the big picture and drill into individual tools. This is a good start, and often enough to satisfy executives who want regular updates. There are several ways to accomplish this.

Browser mashup

You can bring together several data sources within a browser using client-side scripting such as Greasemonkey (Figure 17-2). This is how Kampyle, which collects VOC data on whether visitors like a page, interacts with Google Analytics.

1. Visitors visit your site.
2. The various monitoring tools collect relevant information from their activities, both on your site and off. They may also collect data on competitors. Anything that can be correlated by time is a candidate.
3. You visit the analytics portal (such as Google Analytics) and select a particular time range or segment.
4. The Greasemonkey add-in runs a script that's triggered by your original request to Google Analytics. This script reads the time range from the Google Analytics page (e.g., "last week"), then rewrites the contents of the page that has just loaded so that it contains elements from other sites.
5. The data from each service—charts and graphs all aligned for the same time range—display in a single view.



Recent visitors

List Most active

7:14pm	208.65.161.247	google.com
6:49pm	24.222.208.97	rednod.com
6:45pm	96.23.202.106	
6:22pm	75.58.20.2	google.com
6:16pm	90.146.251.11	
6:16pm	88.117.228.70	google.at
5:26pm	206.207.225.81	google.com
5:20pm	171.71.119.235	
2:49pm	10.10.10.88	
2:38pm	76.166.220.128	

Figure 17-3. A desktop mashup of TweetStats, GetClicky, FeedBurner, Kampyle, and Google Analytics data displayed through the Mac console. Note the lack of consistent time ranges across components

You can create your own Greasemonkey scripts to consolidate data like this. One of the advantages of this approach is that it runs on your browser, where you probably have already saved logins to the various tools that are required. A third-party website can't sign in on your behalf and retrieve all of the data from all of the tools, but in this case it's your browser, not the third-party site, that's retrieving it.

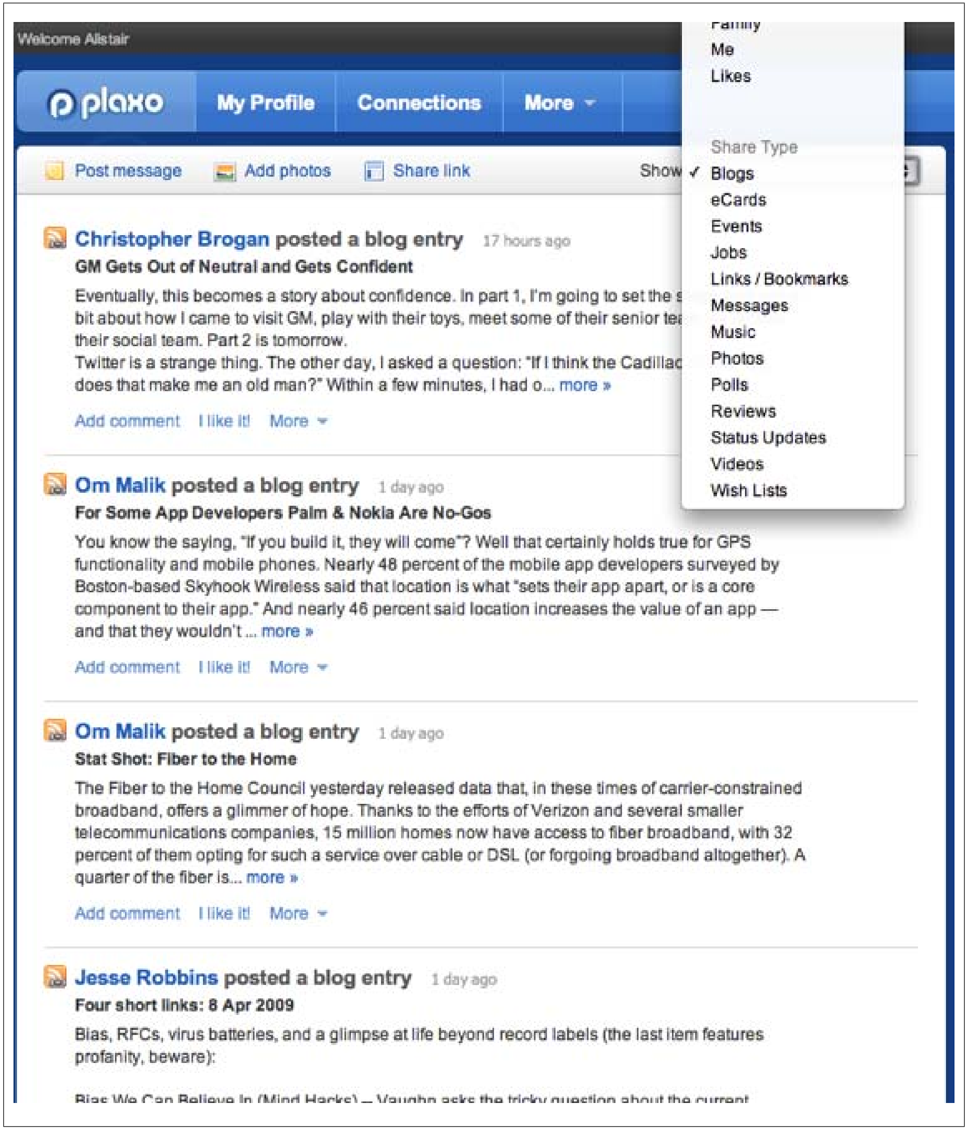
Desktop mashup

Some desktops allow you to embed page snippets from several sources (Figure 17-3). On the Mac, for example, the Console view can display data from many sources by querying several websites and displaying specific `DIV` tags.

On Windows, you can use Active Desktop or widgets to embed data like this. Desktop mashups work well for a wall display or other visualization that won't change much.

Site mashup

A number of websites let you pull together snippets of other sites, RSS feeds, and social network traffic. Social network consolidators like FriendFeed and Plaxo (Figure 17-4) subscribe to several data sources and build a timeline of activity across all of them.


Welcome Alistair

plaxo | My Profile | Connections | More

Post message | Add photos | Share link | Show

Family
Me
Likes

Share Type
✓ Blogs
eCards
Events
Jobs
Links / Bookmarks
Messages
Music
Photos
Polls
Reviews
Status Updates
Videos
Wish Lists

Christopher Brogan posted a blog entry 17 hours ago
GM Gets Out of Neutral and Gets Confident
Eventually, this becomes a story about confidence. In part 1, I'm going to set the s
bit about how I came to visit GM, play with their toys, meet some of their senior tea
their social team. Part 2 is tomorrow.
Twitter is a strange thing. The other day, I asked a question: "If I think the Cadillac
does that make me an old man?" Within a few minutes, I had o... more »
Add comment | I like it! | More

Om Malik posted a blog entry 1 day ago
For Some App Developers Palm & Nokia Are No-Gos
You know the saying, "If you build it, they will come"? Well that certainly holds true for GPS functionality and mobile phones. Nearly 48 percent of the mobile app developers surveyed by Boston-based Skyhook Wireless said that location is what "sets their app apart, or is a core component to their app." And nearly 46 percent said location increases the value of an app — and that they wouldn't ... more »
Add comment | I like it! | More

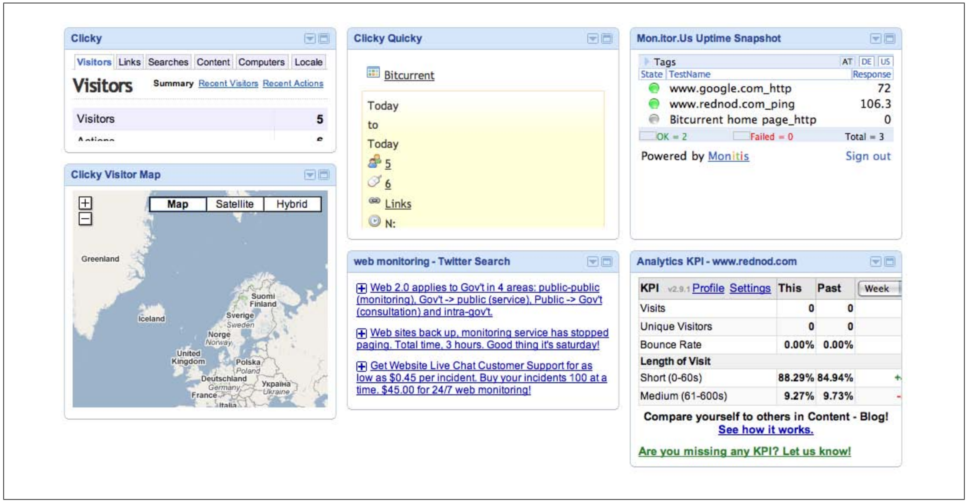
Om Malik posted a blog entry 1 day ago
Stat Shot: Fiber to the Home
The Fiber to the Home Council yesterday released data that, in these times of carrier-constrained broadband, offers a glimmer of hope. Thanks to the efforts of Verizon and several smaller telecommunications companies, 15 million homes now have access to fiber broadband, with 32 percent of them opting for such a service over cable or DSL (or forgoing broadband altogether). A quarter of the fiber is... more »
Add comment | I like it! | More

Jesse Robbins posted a blog entry 1 day ago
Four short links: 8 Apr 2009
Bias, RFCs, virus batteries, and a glimpse at life beyond record labels (the last item features profanity, beware):
Bias We Can Believe In (Mind Hacks) — Vaughn asks the tricky question about the current

Figure 17-4. A Plaxo feed filtered for blogs; status update aggregators can combine notifications across many different message sources into a single river of news

Another class of consolidated view is the personal dashboard provided by sites like Pageflakes or Google's iGoogle home page (Figure 17-5).

Many of the web mashup platforms that are available for free on the Internet limit what you can embed, so for more control, you may want to build a consolidated view of



Visitors	5

State	TestName	Response
●	www.google.com_http	72
●	www.rednod.com_ping	106.3
●	Bitcurrent home page_http	0
OK = 2	Failed = 0	Total = 3

Web 2.0 applies to Gov't in 4 areas: public-public (monitoring), Gov't -> public (service), Public -> Gov't (consultation) and intra-gov't.

Web sites back up, monitoring service has stopped paging. Total time, 3 hours. Good thing it's saturday!

Get Website Live Chat Customer Support for as low as $0.45 per incident. Buy your incidents 100 at a time. $45.00 for 24/7 web monitoring!

KPI v2.9.1 Profile Settings	This	Past	Week
Visits	0	0	
Unique Visitors	0	0	
Bounce Rate	0.00%	0.00%	
Length of Visit			
Short (0-60s)	88.29%	84.94%	+
Medium (61-600s)	9.27%	9.73%	-

Compare yourself to others in Content - Blog!
See how it works.

Are you missing any KPI? Let us know!

Figure 17-5. A combination of widgets and RSS feeds from searches assembled in iGoogle helps put many views in one place

what's going on either in your own web page or on a shared intranet server such as SharePoint.

Some vendors' dashboards also behave like personal dashboards, letting you add third-party dashlets from other sources to their displays (Figure 17-6).

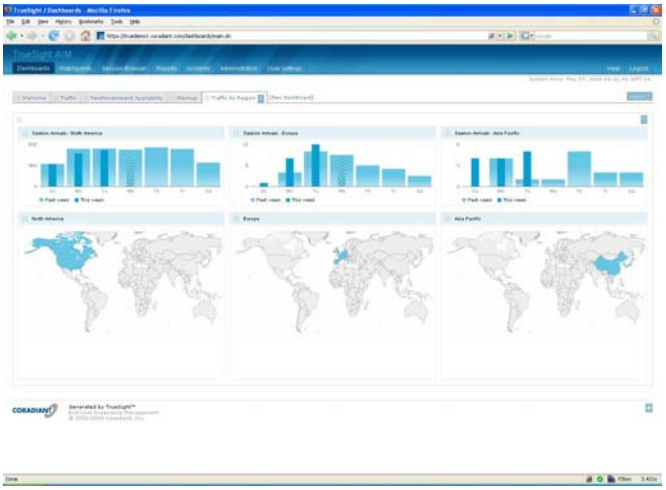
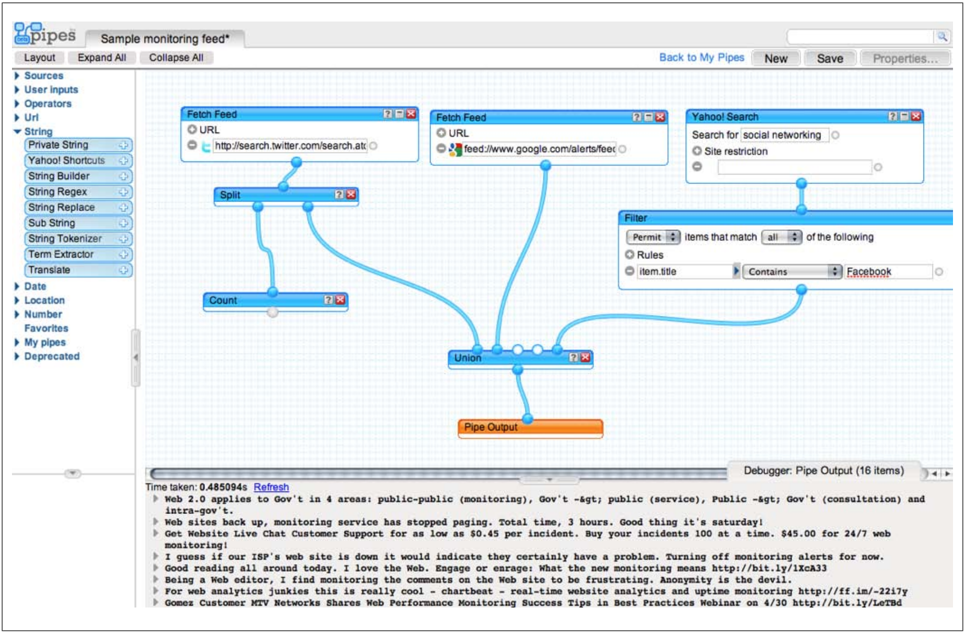


Figure 17-6. A TrueSight dashboard showing RUM data above dashlets from a third-party source (in this case, maps from Google Charts)



```
Time taken: 0.485094s Refresh
▸ Web 2.0 applies to Gov't in 4 areas: public-public (monitoring), Gov't -&gt; public (service), Public -&gt; Gov't (consultation) and
  intra-gov't.
▸ Web sites back up, monitoring service has stopped paging. Total time, 3 hours. Good thing it's saturday!
▸ Get Website Live Chat Customer Support for as low as $0.45 per incident. Buy your incidents 100 at a time. $45.00 for 24/7 web
  monitoring!
▸ I guess if our ISP's web site is down it would indicate they certainly have a problem. Turning off monitoring alerts for now.
▸ Good reading all around today. I love the Web. Engage or enrage: What the new monitoring means http://bit.ly/1XcA33
▸ Being a Web editor, I find monitoring the comments on the Web site to be frustrating. Anonymity is the devil.
▸ For web analytics junkies this is really cool - chartbeat - real-time website analytics and uptime monitoring http://ff.im/-22i7y
▸ Gomez Customer MTV Networks Shares Web Performance Monitoring Success Tips in Best Practices Webinar on 4/30 http://bit.ly/LeTBd
```

Figure 17-7. A Yahoo! Pipes example blending the RSS feed of a Twitter search, a Google Alert feed, and filtered Yahoo! Search results into a new feed

One of the advantages of a site-based single pane of glass is that it's accessible to anyone and available from anywhere. You're not tied to a particular desktop or browser installation. On the other hand, some monitoring services may not let other sites pull in data on your behalf, and each component will stand alone, without being able to take time ranges or segmentation cues from other components.

Roll your own mashup

An increasing number of sites and tools are integrating APIs into their products. Google Analytics is opening up its data to third-party tools, and social networks like Twitter have wide-open APIs that have created the wide variety of Twitter analysis sites we saw in Chapter 14.

Building a feed with Yahoo! Pipes

Yahoo! Pipes and Microsoft Popfly let you link feeds of data, text processing, filters, and searches together to consolidate information. For example, you can pull several RSS feeds into one master feed, then filter that feed for certain keywords or authors (Figure 17-7).

These tools are easy to configure using drag-and-drop connections between processing steps. When you consider that you can subscribe to a Twitter search term, a competitor's website, and a blog, all through RSS, the power of tools like Pipes and Popfly becomes quickly apparent.

As these APIs become more common, it'll be easier than ever to pull in raw data from many sources and render it in your own application, should you have the time and resources to do so.

Whether you use a client, browser, or site mashup, or decide to roll your own, make sure the data you display is clickable—either to drill down into more detail or to link back to the service from which the data came. That way, when you see something that's broken, you can quickly link to the individual tool that generated that part of the consolidated display and investigate further.

Comparing apples to apples

The individual tools from which you're collecting data generate reports in their own ways. Your goal is to line up multiple data sources to display them sensibly so you can see the big picture. You can do this based on data they all share.

Time, for example, is something all tools' data has in common. You can get graphs from several monitoring tools for a particular day. There are other dimensions you may be able to segment by: page name or URL, visitor name, campaign name, region, carrier, and so on.

If you're creating a browser-side mashup, your "anchor" data source will likely be web analytics. With the Greasemonkey approach outlined above, the analytics application dictates the segmentation—a particular time, a certain URL, or a single city, for example. The JavaScript you execute with Greasemonkey finds this segment by parsing the page, then it queries all the other tools' reports for the same segment. If you're viewing a week-long graph of conversion rates, the client-side JavaScript could send a request to a synthetic monitoring tool and retrieve a graph of performance that week, then insert it as a component within the analytics page container.

Desktop mashups can't do this. Each component of the desktop mashup stands alone because the various components can't "take their cues" from data on the parent page. Every chart and graph is unrelated to the others on the screen.

If you're using a website mashup such as Pageflakes or iGoogle, you probably won't have this much programmatic control. The best you can do is to define tabs that make sense and then collect related reports from each tool. Here are some examples to get you started:

- Your "regions" tab might show the site's performance by region alongside the analytics tool's view of traffic by region.

- Your "performance" tab might blend DNS uptime from a synthetic testing service with checkout page host latency from a RUM product and show conversion rates to help tie it back to business outcomes.
- Your "awareness" tab might include a chart of Twitter followers, new visits, Google Insights search trends, FeedBurner subscriptions, and Technorati ranking.
- Your "URLs" tab could show the top URLs in terms of poor performance, arrival and departure, bounce rate, and time on page.
- Your "virality" tab could show the top search results for certain terms, the number of people who have completed an invitation goal, bounce rate data from an email system, and which social networks and messages are being forwarded most.

Alerting Systems

You don't just want to consolidate for visualization. If you want to be sure your PDA isn't constantly vibrating with alerts, you need a way to bring together alerts and metrics from many sources, detect problems, and let you know about the important ones.

IT operators deal with hundreds of servers and network devices, and over the years vendors have built sophisticated algorithms for baselining and detecting problems. Their jobs revolve around a queue of incidents flowing in, each of which needs to be triaged and repaired. To handle the deluge, many monitoring tools include algorithms that dynamically baseline measurements, then detect problems when those baselines are exceeded.

The good news is that many of these systems don't care what kind of data you feed into them. Companies like Netuitive, ProactiveNet (now BMC), and Quantiva (now NetScout) can take in lots of different data sources and form an opinion about what's normal.

You can pump in conversion rates, the number of feedback forms visitors have submitted, page satisfaction rankings, and network latency. Monitoring tools will chew on all this information and let you know when anomalies occur. Some RUM vendors, such as Coradiant, build this technology into their products.

Consolidating information for alerting involves four steps:

1. *Normalize the data*. These systems work best when they receive numeric information at regular intervals. If you're using RUM data, for example, you don't just send web requests to the analysis tool. You need to feed a metric such as "95th percentile host latency for the last minute," which can take some computation.
2. *Collect the data*. These days, simple protocols like RSS feeds make it relatively easy for one system to pull data from another. You can even use web services like Yahoo! Pipes to pull many sources together, or you can build a collection service using a cloud platform like Google's App Engine.

3. *Do the math*. Even if you don't have access to sophisticated algorithms, you can still derive relatively simple ones. We know of one web operator whose main metric is the average rate of change in host latency—a gradual slowdown or speedup isn't a problem, but a sudden spike is easy to identify. You could do the same thing with checkout prices on the site.
4. *Send alerts*. This may involve sending an email or SMS (Short Message Service) message, turning a light red, or creating a trouble ticket. Much of this already exists in an organization's Network Operations Center (NOC), but the things they're monitoring generally don't include business data like conversions or form abandonments.

The point here is that you may need to pull in data from various monitoring services at regular intervals through whatever APIs or RSS feeds those services provide, then decide how to treat that data as a set of signals about the overall health of the website.

Tying Together Offsite and Onsite Data

One of the most difficult types of data integration involves linking what happens elsewhere in the world to your web analytics data. Whether it's a poster on a bus that contains your URL, a mention on a social network, or a call to your sales department that began on your website, it's hard to monitor the long funnel.

There are two main ways of tying offsite activity back to your website. The first is to gather information from social network sites where visitors opt in and share their identities. The second is to mark your offsite messages with custom URIs that you can track with your analytics tool.

Visitor Self-Identification

The first approach, which is the basis for systems like Open Social, Beacon, OpenID, myBlogLog, and Disqus, requires that you enroll yourself in the social network somehow. Then visitors who want to share their identity with you can do so.

This is the simplest way to track users. You may even be able to analyze inbound links from certain social networks to see who's sending traffic. For example, referrals from *www.twitter.com/acroll* indicate that the visit came from the profile page of Twitter user "acroll." In practice, traffic of this kind is only a fraction of Twitter traffic, as most people view links in their own Twitter feeds and with third-party desktop clients like Tweetdeck and Twhirl.

Many social networks want to monetize the analysis of their traffic; indeed, "selling analytics" is the new "ad-driven" business model. It's therefore unlikely that you'll get visitor analytics for free at the individual visitor level unless you ask visitors to log in. But don't write this possibility off: in many situations, your market will gladly tell you who it is. Musicians, for example, have legions of fans who crave the attention and

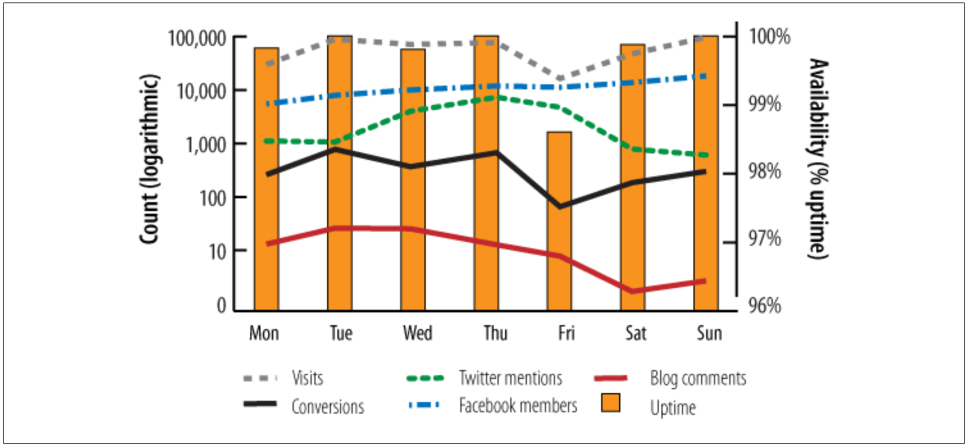


Figure 17-8. The most basic way to consolidate data manually is to display multiple data sources in a spreadsheet using time as a common key

acclaim of the artists they support, and may be willing to divulge their identities in return for access to a celebrity or early notification about tickets or appearances. There are often ways you can encourage visitors to self-identify if you're creative.

Using Shared Keys

Any data analysis requires a key of some kind. The more specific that key, the more closely you'll be able to tie together the data. All interactions have some basic attributes: where they happened, when they happened, any keywords that were mentioned, and in some cases, the person who said it.

The easiest way to compare two sets of data, then, is by time. Lining up graphs of mentions on Twitter, Facebook friends, web traffic, and so on, is an easy way to see which factors are related (Figure 17-8).

If keys like date or geography aren't enough to consolidate your information—particularly if you want to link individual actions to individual outcomes, rather than just comparing aggregate data over time—you'll need to make your own keys. These may be a unique URL on printed materials, a URI parameter hidden in a shortened URL, a unique phone number for each visitor to call, or some other way of linking together the many offsite interactions your visitors have with you.

In the end, you'll have to take whatever data you can get, assemble it according to common keys like time or geography, and track it over time to determine how your many sources of web monitoring information are related.