



WHITE PAPER



SENDING DATA TO SITECATALYST

Tracking Online and Offline Data

August 21, 2008

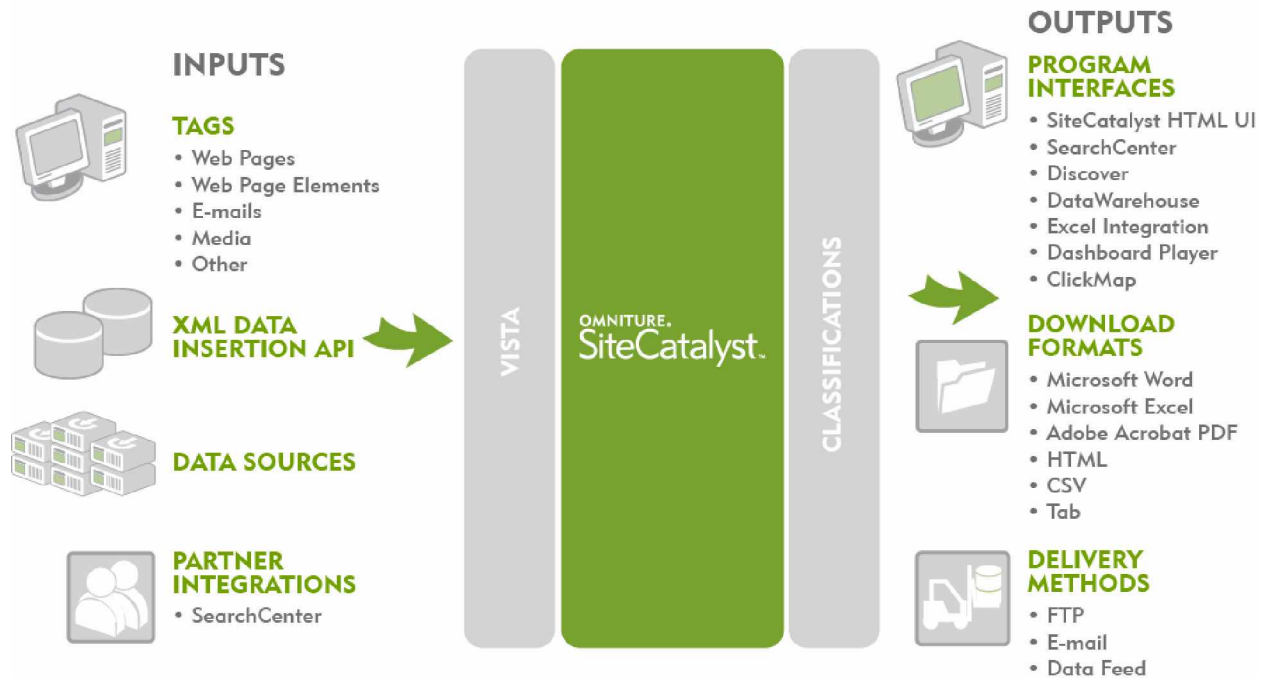
Version 2.0



1 Overview

In order to track information in as many environments as possible, Omniture has created multiple ways to send data into SiteCatalyst. These methods include tracking information in real-time from web sites, emails, mobile devices, client-server applications, and most applications that can access the Internet as well as reporting on information from offline systems via the Data Sources and partner integration platform. The image below provides a visual overview of the data collection and output options and this document provides a more detailed overview of each of these methods.

Figure 1-A: The Omniture Input/Output Data Workflow



2 Tracking Online Data

SiteCatalyst enables you to track various forms of online data, including web pages, links and links as web pages, page elements, and rich media. The process and syntax for tracking each of these online data formats are described in the following sections.

2.1 Tracking Web Pages

The SiteCatalyst web page data collection process is fairly simple. As a web page with SiteCatalyst code loads, the SiteCatalyst code collects certain variables at the page and browser level and calls the SiteCatalyst JavaScript file. These variables (and other identifiers) are used to facilitate the data collection process, and can be dynamically populated with server or application variables.

The JavaScript file builds an image request, or web beacon. The image request uses a transparent 1x1 pixel image to pass data from the web page to the Omniture data center. The process is repeated every time the visitor accesses a page that contains SiteCatalyst code. The following example shows a simple HTML page with SiteCatalyst code.

```
<html>
  <head></head>
<body>
<!-- SiteCatalyst code version: H.17.
Copyright 1997-2008 Omniture, Inc. More info available at
http://www.omniture.com -->
<script language="JavaScript" type="text/javascript" src="http://INSERT-
  DOMAIN-AND-PATH-TO-CODE-HERE/s_code.js"></script>
<script language="JavaScript" type="text/javascript"><!--
/* You may give each page an identifying name, server, and channel on
the next lines. */
s.pageName="Test Page"
s.server=""
s.channel=""
s.pageType=""
s.prop1=""
s.prop2=""
s.prop3=""
s.prop4=""
s.prop5=""
/* Conversion Variables */
s.campaign=""
s.state=""
s.zip=""
s.events=""
s.products=""
s.purchaseID=""
s.eVar1=""
s.eVar2=""
s.eVar3=""
s.eVar4=""
s.eVar5=""
/***** DO NOT ALTER ANYTHING BELOW THIS LINE ! *****/
var s_code=s.t();if(s_code)document.write(s_code)//--></script>
<script language="JavaScript" type="text/javascript"><!--
if(navigator.appVersion.indexOf('MSIE')>=0)document.write(unescape('%3C')+'\!
-'+'-')
//--></script><noscript><a href="http://www.omniture.com" title="Web
  Analytics"></a></noscript><!--/DO NOT REMOVE/-->
<!-- End SiteCatalyst code version: H.17. -->

Sample HTML document.<br>

</body></html>
```

A variety of mechanisms can be used to populate the variables, including server-side code, JavaScript functions, or event plug-ins within the SiteCatalyst JavaScript file. The following example shows a custom plug-in residing in the JavaScript file that populates the campaign variable based on query string parameters in the URL.

```
function s_doPlugins(s) {
    /* a,b,c = list of query string parameters to look for and
    populate the "s.campaign" variable. */
    s.campaign=s.getQueryParam("a,b,c")
}
```

2.2 Tracking Links & Page Elements

In addition to tracking the page load, SiteCatalyst also tracks specific actions on a page. The SiteCatalyst JavaScript file attaches a “listener” function to the onClick event (using either the attachEvent method for Internet Explorer browsers or the addEventListener method for Netscape/Mozilla browser) within the document body. This enables ClickMap, exit link tracking, download link tracking, form analysis, as well as use of the link handler plug-ins. Specific business requirements can be met by passing in SiteCatalyst variables on link clicks. Additionally, custom link tracking can be used to track additional information about a link when the standard methods are not sufficient. More detail on each of these is provided below.

2.2.1 ClickMap

ClickMap is a browser plug-in, and a module of SiteCatalyst™, that allows Omniture clients to visually measure traffic (page views), conversion, and success metrics within the pages of a web site. These metrics are overlaid on top of the page’s links and answer critical questions, including the following.

- What is the value of an individual page on your site?
- What is the value of an individual element on a web page?
- What is the most valuable “digital real estate” on your page?

A visual display highlights the most relevant elements on your web page and calculates the overall value of that page to your site’s success, based on any other success metric you define. ClickMap provides all this data by tracking the links clicked on a page. Please see the *ClickMap – Implementation and Use* white paper for more information

2.2.2 Exit link & Download link tracking

The exit link report shows links a visitor clicked on to leave your site and go to another site. The download link report helps you understand how often your visitors download files from your site. SiteCatalyst will *automatically track file downloads and exit links* based on configuration parameters set in the JavaScript file. For exit link tracking, any click of a link that does not match a list of internal URL filters will generate an image request to SiteCatalyst. For download link tracking, any click on a link that matches a list of download file types will generate an image request to SiteCatalyst (please see the *Implementation Manual* for more information). Links that need additional information passed about them can be manually tracked using custom link code as explained in the *Tracking Custom Links* section of this document.

2.2.3 Link Handler Plug-in

The linkHandler plug-in was created to automate custom link tracking and conditionally track custom links. The link handler plugin takes advantage of the fact that the SiteCatalyst JavaScript file attaches to the onClick event. Instead of manually adding functionality to the onClick event of a link (see *Tracking Custom Links* in this document), the linkHandler plug-in may be used to identify links, set a link type (exit or download), set variables, or fire events whenever a link is clicked. Specific examples include the following.

- Identify links that go thru a redirect and should be considered Exit Links, even though they match the linkInternalFilters variable.
- Set an event or SiteCatalyst variable on a link click.
- Give friendly names to file downloads based on filters.
- Copy the URL of the link clicked into a prop variable to allow for classification on those values.
- Reporting exit, download, or custom links as page views in SiteCatalyst.

2.2.4 Tracking Custom Links

Custom link code allows file downloads, exit links and custom links to be tracked in situations where standard link tracking is not sufficient or applicable. Custom link code is typically implemented by adding an onClick routine to a link, or adding code to an existing routine; however it can be implemented from essentially any JavaScript event handler or function.

The basic code to track a link using custom link code is shown in the following example.

```
<a href="index.html" onClick="var s=s_gi(report_suite_id); s.tl(this,'o','Link Name');
">My Page</a>
```



NOTE: The highlighted text "report_suite_id" must be replaced with the appropriate report suite ID. Also note that the "s_account" variable has also been initialized and can be used in this function. This example is for H-code. Please check with your Implementation Consultant if you need G-code. Refer to the *Link Tracking* white paper for more information.

2.3 Tracking Rich Media - the t() and tl() functions

The standard Omniture JavaScript code contains two functions used to track user interaction in a non-HTML environment such as AJAX. These functions, t() and tl(), have different properties and uses.

Table 2-A: The t() and tl() Functions

Function	Description
t()	This function is used to track a "virtual" web page load. Calling this function results in a page view.
tl()	This function is used to track links. Calling this function results in a Custom link, Download link, or Exit link

These functions are for H-code. Please check with your Implementation Consultant if you need G-code. Each time the t() function is called, SiteCatalyst increments the total page views for the site. Additionally, any variables (Omniture JavaScript variables) that have a value (other than an empty string: "") at the time the function is called, are sent to SiteCatalyst. Use t() when an event should be considered a page view, for example, when moving from one "page" to another in a multi-"page" AJAX application. Calling t() will add another page to the path reports (Next Page Flow, Fallout, etc.) as well as impact the time-spent-on-page calculations.

The previous section showed that `t()` sends data in the same way that a page load does. The `tl()` function, however, sends data in the same way that custom link tracking does. In other words, the total page views to the site are not incremented when calling `tl()`, but non-page view data is recorded. Calling `tl()` will NOT add another page to the path reports (Next Page Flow, Fallout, etc) and will NOT impact the time-spent-on-page or time-spent-on-site calculations. However, if a custom variable is passed in a custom link, pathing for that variable will be calculated if enabled. Refer to the *AJAX – Tracking Rich Media Applications* white paper for more information.

2.3.1 Page views vs. links

One of the most commonly asked questions with rich media is how to track micro-level activity separate from macro-level activity, and when it is appropriate to do either. For example, say you have an application that allows customers to uniquely configure a product. The application may have significant steps users are exposed to. Are these steps considered “page views”? In addition, there are micro level activities within each step. Should these activities be tracked as page views? What if you want to understand the flow between activities, or which features get the most activity? The trouble with rich-media is that each application is unique. They are designed to mimic realistic actions and because actions are relative to the situation, the possibilities are endless. However, most applications have a few major components: mile stone steps, features, and micro-level actions within features. So, while each application requires some consideration as to what specifically should be measured, there are some generalizations that can be applied to rich-media tracking.

Table 2-B: Micro and Macro-level Activity

Activity	Description
Macro-level Activity	This usually constitutes the loading of the application, which provides information on visits, visitors, instances, value to future actions, etc., but it can and should also represent major steps in the process. A good rule of thumb is that if a rich media action changes the application more than 50% (or whatever is considered significantly changing the user experience or content), then it is macro-level, and should be tracked as a page view.
Micro-level Activity	This includes any changes less than 50% (or not considered as significantly changing the user experience or content). Toggling between color selections, for instance, would be considered micro-level activity. Omniture recommends that micro-level tracking be related to features. For example, in the case of toggling between colors, is it really important to understand which colors were considered? Or is it more important to know that the color selection feature was used? Perhaps both are important, and if so, capture both; but, when measuring the effectiveness of rich media, consider the feature level activity as being more valuable. All micro-level activity should be tracked as “custom links” with specifics measured through associated “traffic” variables. This will ensure that page views are not inflated by micro-level activity, and allows for path analysis through the traffic variable.

2.4 ActionSource

Omniture ActionSource is a patent-pending method for natively tracking Flash applications used with the ActionScript programming language. This solution removes dependencies on JavaScript for Flash tracking through the use of native ActionScript components.

Historically, tracking Flash has been dependent on two programming languages: ActionScript and JavaScript. Communication between these technologies complicated the implementation process and introduced technical barriers. For Flash developers who did not understand the nuances of JavaScript, it was difficult to put analytics

solutions in place. Even for those who could deploy the JavaScript solution, limitations based on language and browser communication caused performance and data integrity problems including the following.

- Cancellation of analytics when Flash attempted to communicate through the browser to JavaScript for analytics and to link out to another browser page
- Limitations of 508 characters per data transmission from Flash through the browser to JavaScript
- Playback delays for animation while JavaScript is executing
- Localized tracking only – JavaScript must be present on the same page as the Flash file to execute analytics tracking

Omniture ActionSource was developed to simplify the implementation process and to improve performance and accuracy. Its native ActionScript engine removes the dependency on JavaScript and maximizes application portability and accuracy. Some of the key benefits of this solution are listed in the table below.

For more information on ActionScript, refer to the *Omniture ActionSource AS3 Implementation Manual*.

2.5 Media Tracking

Media on your site is tracked by gathering basic information from the media player and building a session of events that are sent to Omniture's collection servers for processing. The following basic information is either collected automatically or provided to ActionSource or JavaScript:

- • Media Name
- • Media Length (in seconds)
- • Media Player Name

Both ActionSource for AS3 and Omniture's JavaScript collection file version H.15 and up support the Media tracking module. The media module is option when using JavaScript, but is always present in ActionSource. If "s" is the name of your JavaScript object or ActionSource instance, you reference the media module as "s.Media". In both implementations you can manually track a media session by calling four functions. Alternatively, media may be automatically tracked by setting the "s.Media.autoTrack" variable to true.

Media tracking is accomplished by collecting data about which portions of the video are viewed and sending that data at the end of the video. Because you a media player or browser window may be closed before the media has finished playing, the collection code buffers the media session in a cookie or flash shared object that is passed in with the next page view. The cookie and shared object are named "s_br". You can disable the cookie or Flash shared object by setting the s.disableBufferedRequests variable to true. This cookie and shared object are retained for 30 minutes because the data must be associated with the visit in which it occurred.

For more information on media tracking, refer to the *Media Tracking Implementation Manual*.

2.6 Tracking Links as Web Pages

There may be times when you want to report a link clicked as a page view. For example, you want to add the link to the path reports or more accurately report time-spent-on-page metrics. There are two alternatives to count links as page views.

2.6.1 Links that Leave the Page

If the link leaves the page, simply use standard link tracking (exit link & download link) or custom links, or an explicit call to s.tl(). Each of these functions will ensure that the image request is fired before the user leaves the page by inserting a 500 ms delay. VISTA can then be used to convert these links into page views.

2.6.2 Links that do NOT Leave the Page

Alternatively, you may have links that simply change elements on a page, e.g. DHTML or open a new page in a new window. In these cases, the 500ms delay is not needed, since the page doesn't change. To record these links as page views, simply use the linkHandler plug-in (contact your Implementation Consultant for configuration instructions). For rich media applications, call s.t() directly. To record links as links without incurring the 500ms delay, a VISTA rule can be written to convert a page view (sent via s.t()) or the linkHandler plug-in) into a custom, download, or exit links.

2.7 Using the Data Insertion API

The XML Data Insertion API is a tool that facilitates sending page views into the SiteCatalyst Data Processing Engine via an XML request (instead of standard web beacon data collection). It has been designed to facilitate the integration of SiteCatalyst with applications that cannot be easily tagged with JavaScript or when a browser is not used. Examples include the following.

- Orders/Pages where the URL of the image exceeds 3kb
- Actual file downloads, rather than clicks on links. For example, direct links from Google to non-HTML files
- Order confirmation from a backend system
 - Orders are verified up to 30 minutes after purchase due to fraud or credit checks
 - Any confirmation that happens outside of the web site, like email confirmation

For more information on the Data Insertion API, refer to the *Web Services User Guide*

3 Tracking Offline Data

SiteCatalyst enables you to track various forms of off-network and third-party applications with Data Sources and various partner integration formats.

3.1 Tracking off-network and third-party applications – SiteCatalyst Data Sources & Partner Integration

The feature-rich capabilities of SiteCatalyst coupled with its ease of use make it a valuable tool for understanding website statistics and performing website analytics. As a result, many customers have requested the ability to import other types of time-based data to leverage the features of SiteCatalyst. With Data Sources, data may be easily imported, and immediately made available to users worldwide. Even better, the graphical features of SiteCatalyst may be used to analyze this data either separate from, or in addition to, SiteCatalyst data collected directly from the web site. Examples of data that can be imported via data sources including the following items.

- Email or ad vendor data such as sends, opens, and impressions
- Off-line sales
- Off-line returns

Also, SiteCatalyst has a robust integration with many other data providers including the following items.

- DoubleClick for integrating email and ad-server integration
- Salesforce.com for integrating CRM/SFA data
- LinkShare for affiliate marketing
- Advertising.com for integrating ad network data
- Vignette for content management
- CheetahMail for email remarketing
- Offermatica for A/B testing, multivariate analysis, and optimization
- OpinionLab for user experience & feedback data
- Various others

For more information, refer to the *Data Sources* white paper or Omniture partner directory (<http://www.omniture.com/partners/showcase>).

3.2 Tracking additional attributes – SiteCatalyst Classifications

To make your analytics data more actionable, SiteCatalyst provides customers with a powerful classification system. The classification system allows you to associate SiteCatalyst data with additional attribute dimensions. The Omniture SiteCatalyst classification system can be configured to pivot around any SiteCatalyst variable including custom insight variables, custom conversion variables, campaign variable, domains, zip codes, and many others. Examples of using classifications include the following.

1. Classifying campaign tracking codes with:
 - a. Creative type (such as link, image, or keyword)
 - b. Creative content (like an image of a father and daughter or an email to a friend)
 - c. Campaign type (such as email, banner, or pay-per-click)
 - d. Campaign vendor (such as CheetahMail, DART, Google, or Overture).
 - e. Cost
2. Classifying productID with:
 - a. Manufacturer
 - b. Size
 - c. Color
3. Classifying ad IDs with:



- a. Advertiser company
- b. Advertiser content
- c. CPC

For more information, refer to the Classifications white paper.



CALL 1.877.722.7088
1.801.722.0139

www.omniture.com
info@omniture.com

550 East Timpanogos Circle
Orem, Utah 84097

