FatWire | Analytics 2.5

Installing and Configuring FatWire Analytics

Document Revision Date: Jun. 26, 2009

FatWire® SOFTWARE

Installing and Configuring FatWire Analytics
Document Revision Date: Jun. 26, 2009
Product Version: 2.5

FatWire Technical Support
www.fatwire.com/Support

FatWire Headquarters
FatWire Corporation
330 Old Country Road
Suite 207
Mineola, NY 11501
www.fatwire.com

Table of

Contents

About This Guide

This guide contains procedures for installing and configuring FatWire Analytics to capture and process data on visitors who browse published Content Server sites, including Engage assets.

FatWire Analytics is a Content Server plug-in that monitors and statistically analyzes website traffic. Its reporting capability enables you to track visitors' interactions with published content from the time visitors start browsing your site, up to the time they leave your site.

Audience

This guide is for installation engineers, Content Server administrators, and anyone else who plans to install and configure FatWire Analytics. Requirements include:

- A thorough understanding of how Content Server is installed.
- Experience installing application servers and the Oracle database management system, creating a database, and using Content Server's administrative interface and utilities (such as Catalog Mover).
- Experience with Java, JSP, and Content Server elements.
- For enabling data capture on Engage reports, a thorough understanding of recommendations and segments is required.

Related Publications

The FatWire library contains many publications for Content Server users, administrators, and developers. The publications are provided as product manuals with your Content Server installation kit. They are also posted on the Web, by version number, at the following URL:

`http://e-docs.fatwire.com`

Check the site regularly for updates.

Other publications, such as case studies and white papers, provide information about Content Server's feature set and business applications. To obtain these publications, contact `sales@fatwire.com`.

Chapter 1

Architecture Overview

This chapter provides an overview of the components that make up the Analytics suite, and outlines the scenarios that you can choose to implement when installing Analytics.

This chapter contains the following sections:

Components of a FatWire Analytics Installation

Analytics is a modular system allowing for a high degree of scalability. An Analytics installation consists of the following components, which communicate with each other through JDBC for database access, connections for HTTP, RMI, and proprietary Socket protocols:

- **Hadoop** – provides distributed data storage (HDFS) and distributed data processing (`Map/Reduce`). The **Hadoop Distributed File System (HDFS**) stores input and output files of Hadoop programs in a distributed manner throughout the Hadoop cluster, thus providing high aggregated bandwidth.
- **FatWire Analytics:**
  - **Analytics data capture application (also called 'Analytics Sensor')** – web application that captures data on the activities of visitors as they browse your online site, and stores that data on the local file system. (For data capture to work, you must embed a special tag, `AddAnalyticsImgTag`, into the pages that you wish to monitor. The tag triggers the data capture process.)
  - **Hadoop Distributed File System (HDFS) Agent** takes the raw data collected by the data capture server and copies it from the local file system to HDFS.
  - **Hadoop Jobs (Scheduler)** runs jobs in a parallel and distributed fashion in order to efficiently compute statistics on the raw data that is stored in HDFS.

    Hadoop implements a computational paradigm named `Map/Reduce`, which divides a large computation into smaller fragments of work, each of which may be executed or re-executed on any node in the cluster. `Map/Reduce` requires a combination of `jar` files and classes, all of which are collected into a single `jar` file that is usually referred to as a "job" file. To execute a job, you submit it to a `JobTracker`. Hadoop Jobs then responds with the following actions:
    - Schedules and submits the jobs to `JobTracker`.
    - Processes raw data captured by the data capture server into statistical data and then writes it to the Analytics database.

    Hadoop provides a web interface to browse HDFS and to determine the status of the jobs.
  - Analytics database – stores the aggregated and statistical data on the raw data captured by the data capture server.
  - **Analytics reporting and administration web applications**
    - The reporting component provides the user interface, used to generate reports.
    - The administration component provides the administration interface, used to integrate Analytics with your Content Server system.

  Typically, the reporting and administration components reside on the same computer.
- **Load balancer** – used to link multiple data capture servers in order to increase performance. Load balancing is also recommended for failover.

A firewall is highly recommended, to protect your Content Server and Analytics systems from intrusion.

Due to the modular nature of Analytics, you have the option to implement a number of scenarios when installing Analytics. "Installation Scenarios," on page 11 describes the more common approaches.

Installation Scenarios

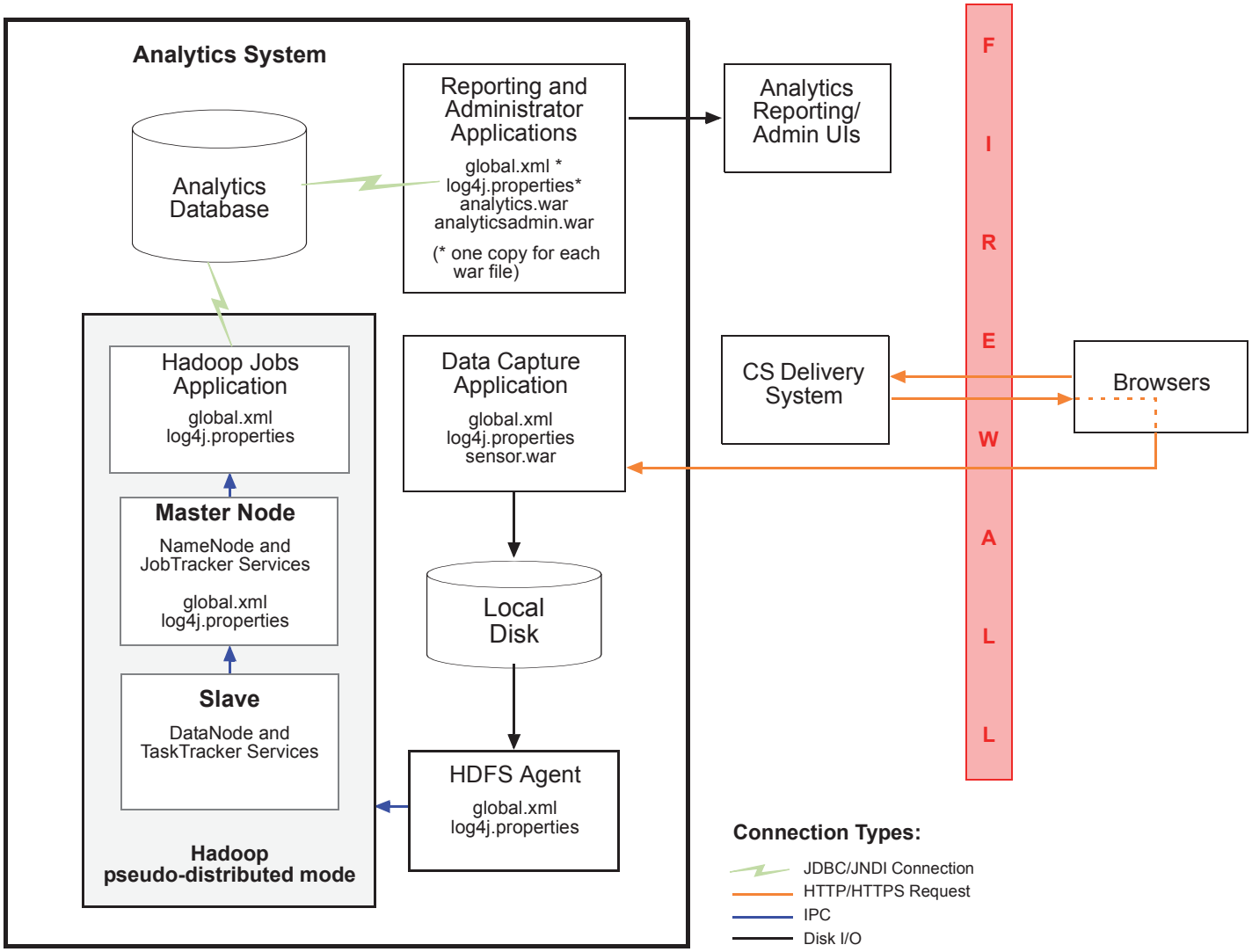
This section describes the different scenarios that you can choose to follow when implementing Analytics on your site. The scenarios are:

- Single-Server Installation: Analytics and Its Database on a Single Server
- Dual-Server Installation: Analytics and Its Database on Separate Servers
- Enterprise-Level Installation: Fully Distributed

Single-Server Installation: Analytics and Its Database on a Single Server

In this scenario, all Analytics components reside on a single, dedicated computer. This scenario works best in situations when you need to test and experiment with Analytics. Figure 1 illustrates a single-server Analytics installation and indicates where configuration files reside and services run. Arrows represent data flow.

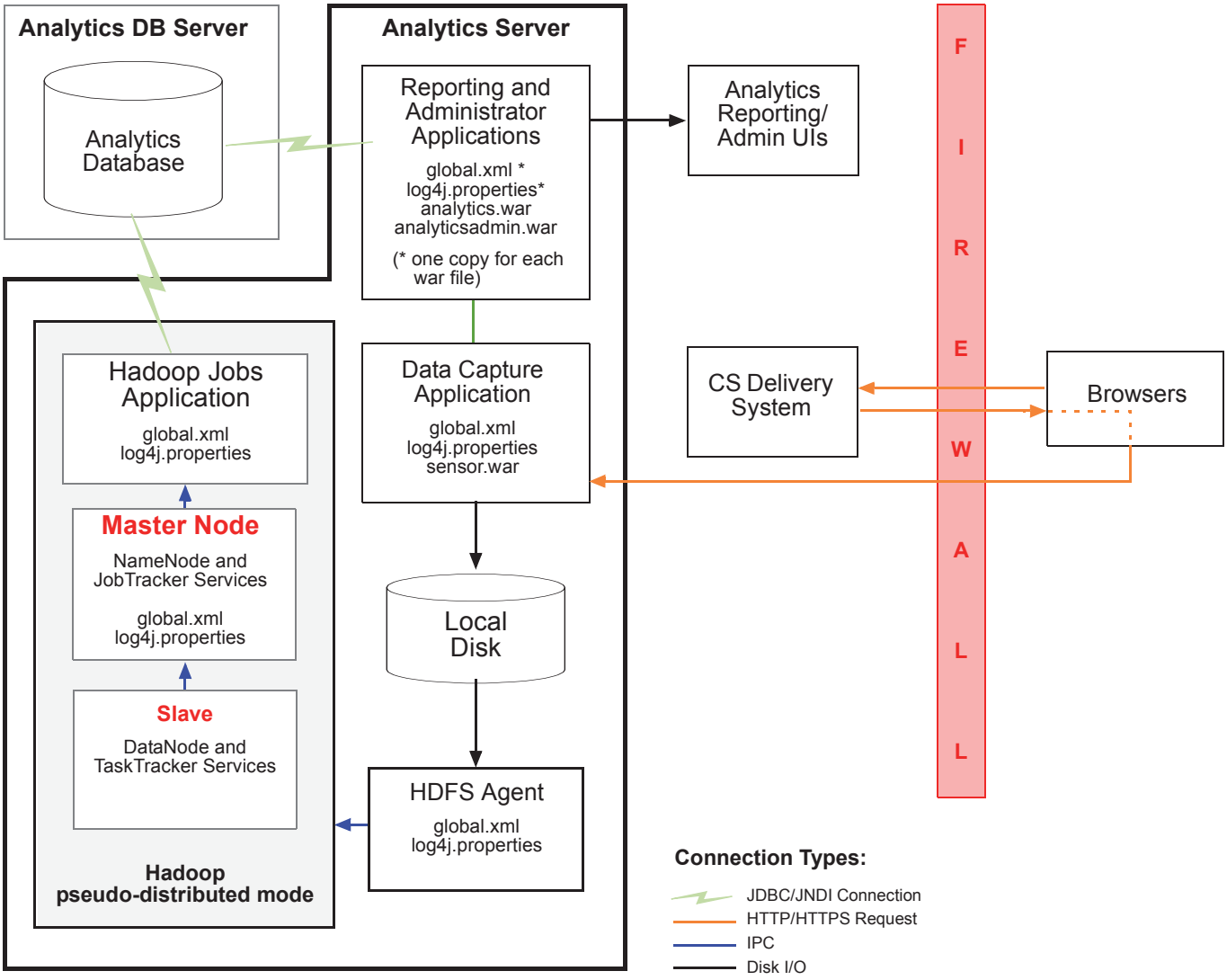
Figure 1: Single-Server Analytics Installation



Dual-Server Installation: Analytics and Its Database on Separate Servers

In this scenario, Analytics components **except for the Analytics database** are hosted on a single, dedicated server; the Analytics database is installed on its own server. This scenario works best in situations when you need to test and experiment with Analytics under increased performance conditions (isolating database transactions from Hadoop jobs minimizes their competition for resources). Figure 2 illustrates a dual-server Analytics installation and indicates where configuration files reside and services run. Arrows represent data flow.

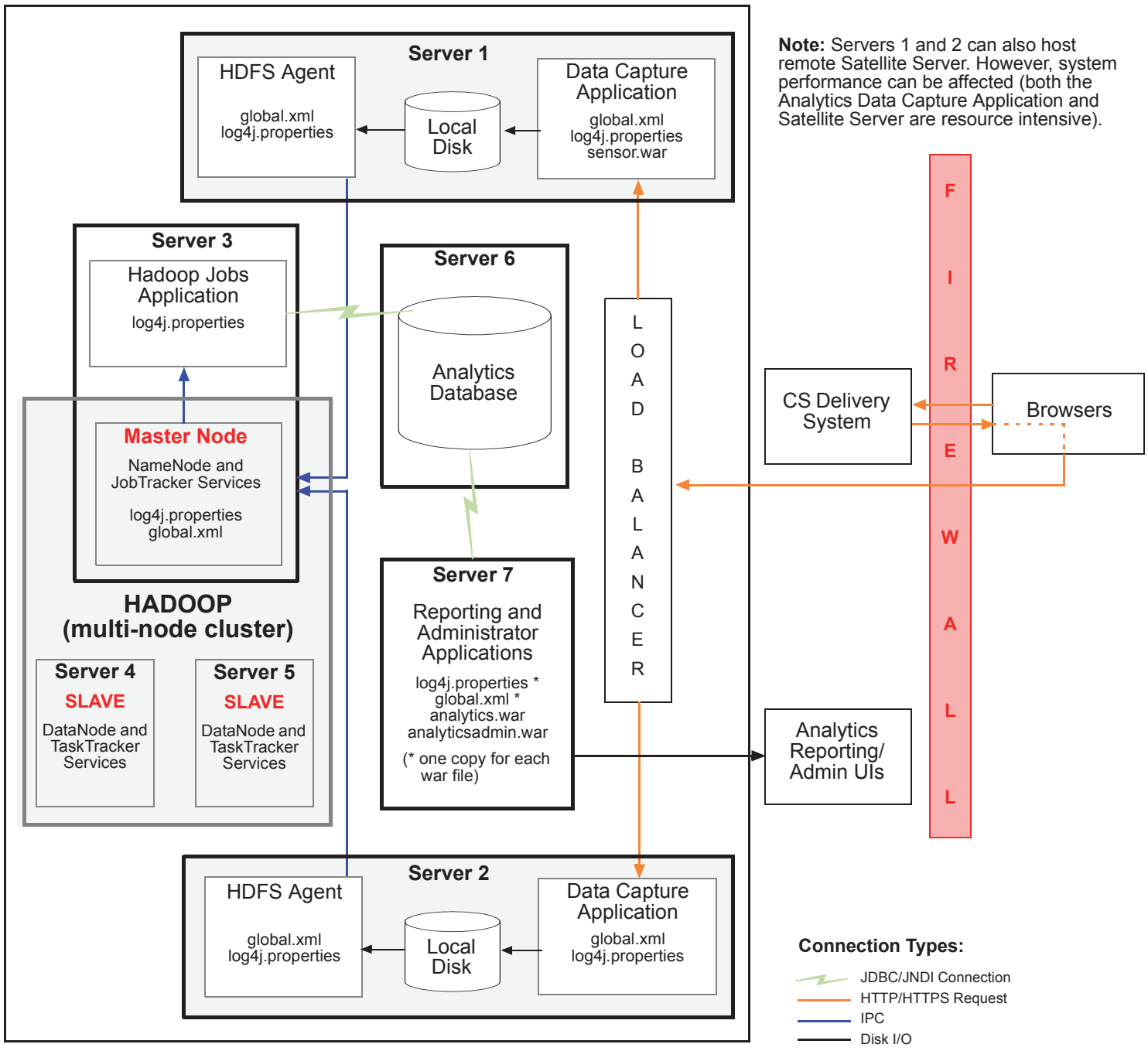
Figure 2: Dual-Server Analytics Installation



Enterprise-Level Installation: Fully Distributed

In this scenario, Analytics components run on separate computers. While more complex, this approach allows for scalability and provides better performance, as each component has dedicated processing power at its disposal. Figure 3 illustrates an enterprise-level installation and indicates where configuration files reside and services run. Arrows represent data flow. **For information about installing Analytics with remote Satellite Server, see the note in Figure 3.**

Figure 3: Enterprise-Level Analytics Installation

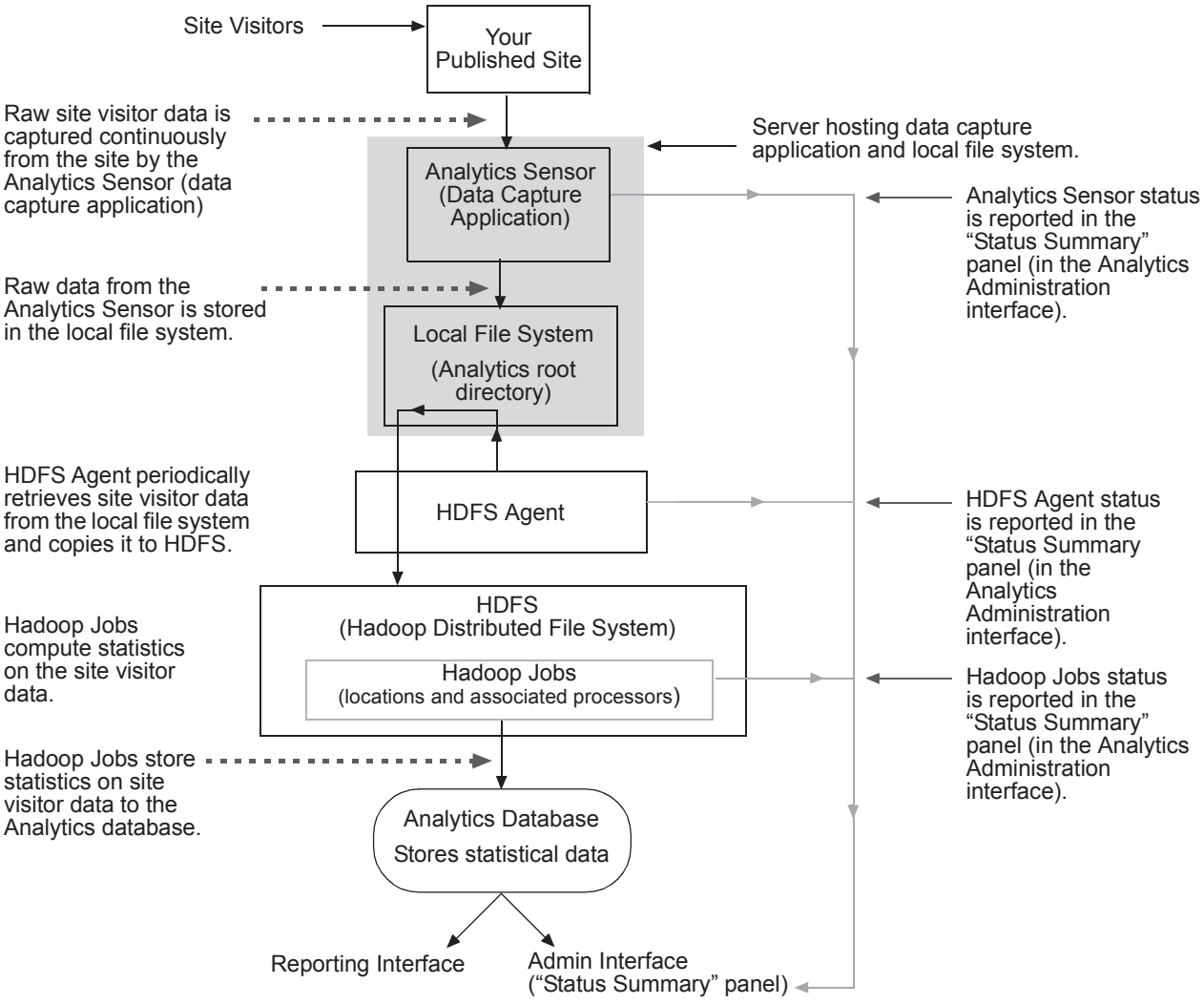


Process Flow

In a functional Analytics installation, raw site visitor data is continuously captured by the Analytics Sensor (data capture application), which then stores the data into the local file system. The raw data in the file system is called on periodically by the HDFS Agent. The HDFS Agent copies the raw data to the Hadoop Distributed File System (HDFS), where Hadoop jobs process the data.

Hadoop jobs consist of locations and FatWire-specific processors that read site visitor data in one location, statistically process that data, and write the results to another location for pickup by the next processor. When processing is complete, the results (statistics on the raw data) are injected into the Analytics database.

The status of Hadoop Jobs can be monitored from the "Status Summary" panel of the Analytics Administration interface. Detailed information about data processing and the "Status Summary" panel is available in Appendix A, "Hadoop Jobs: Processors and Locations."



Terms and Definitions

The terms listed below are used frequently throughout this guide. The glossary defines additional terms.

- The “Analytics Data Capture Application” is also referred to as the “Analytics Sensor,” or simply “sensor.”
- The term “site” in the context of installation/configuration procedures and in the interpretation of report statistics refers to the content management (CM) site that functions as the back end of your online site (or one of its sections).
- “FirstSite II” is the sample content management site, used throughout this guide to support examples of reports and to provide code snippets. FirstSite II is also the back end of the online sample site named “etravel.”

Chapter 2

Prerequisites

This chapter contains prerequisites for installing and configuring FatWire Analytics to run on the Content Server web application.

This chapter contains the following sections:

Pre-Installation Checklist

To install FatWire Analytics, you will run a silent installer (a Java-based script). Before running the silent installer, verify the availability and configuration of all components that support FatWire Analytics.

Required Experience

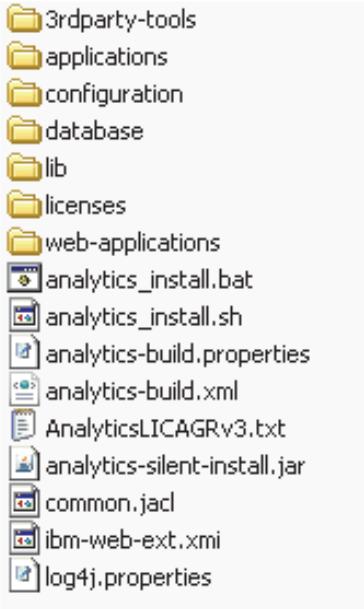
- ❑ To install FatWire Analytics, you must have experience installing and configuring enterprise-level software (such as application servers and databases), and setting system operating parameters.

System Architecture

- ❑ Read Chapter 1, "Architecture Overview" to familiarize yourself with the architecture of the Analytics product and the supported installation options.
- ❑ Read the release notes and the Analytics *Supported Platform Document* to ensure that you are using certified versions of the third-party software that supports FatWire Analytics.

FatWire Analytics Kit

- ❑ Make sure you have a licensed FatWire Analytics Kit (`analytics2.5.zip`). The kit is organized as follows:



The kit contains the FatWire Analytics silent installer files, supporting third-party software, and the Analytics suite. The Analytics suite consists of the following applications:

- Analytics Data Capture web application (also called "sensor")
- Analytics Administrator web application
- Analytics Reporting web application (reporting engine and interface)
- Hadoop Distributed File System (HDFS) Agent
- Hadoop Jobs (scheduler)

Hadoop Installation

- ❑ In the FatWire Analytics Kit, the `3rdparty-tools` folder contains Hadoop binaries. Use the Hadoop binaries to install Hadoop (not the files that are available on the Hadoop web site).
- ❑ Install and configure Hadoop in one of the following modes: local, pseudo-distributed, or fully distributed (***recommended***), whichever is best suited to meet your development, scalability, and performance requirements.
  - The local (standalone) mode is used for development and debugging. By default, Hadoop is configured to run in a non-distributed mode, as a single Java process.
  - The pseudo-distributed mode is used in single-server installations. In this mode, all the Hadoop services (for example, `NameNode`, `JobTracker`, `DataNode` and `TaskTracker`) run on a single node, and each service runs as a separate Java process.

- The fully distributed mode is used for enterprise-level installations. In this mode, Hadoop runs on multiple nodes in a parallel and distributed manner. A minimum of two nodes is required to set up Hadoop: One machine acts as the master node, while the remaining machines act as slave nodes. On the master node, the `NameNode` and `JobTracker` services will be running. On the slave nodes, the `DataNode` and `TaskTracker` services will be running.

Note

For Hadoop installation instructions, refer to the *Hadoop Quick Start* site. The URL at the time of this writing is: `http://hadoop.apache.org/core/docs/current/quickstart.html`

If you install Hadoop in either pseudo- or fully distributed mode, you must configure a property file called `hadoop-site.xml`. Recommended property values are available in Appendix B, "Hadoop Configuration Parameters."

- ❑ Once Hadoop is installed and configured, verify the Hadoop cluster:
  - To determine whether your distributed file system is running across multiple machines, open the Hadoop HDFS interface on your master node:

    `http://<hostname_MasterNode>:50070/`

    The HDFS interface provides a summary of the cluster's status, including information about total/remaining capacity, live nodes, and dead nodes. Additionally, it allows you to browse the HDFS namespace and view the content of its files in the web browser. It also provides access to the local machine's Hadoop log files.
  - View your MapReduce setup, using the MapReduce monitoring web app that comes with Hadoop and runs on your master node:

    `http://<hostname_MasterNode>:50030/`

Content Server System and Supporting Documentation

- ❑ Make sure you have a licensed version of the Content Server web application and it is powering a fully functional online site. (FatWire Analytics on Content Server portal is not supported in this release.)
- ❑ Have Content Server documentation handy. Various steps in the installation process refer you to Content Server product guides for instructions on creating and configuring third-party components. Download the following guide from the FatWire e-docs site:

  Configuring Third-Party Software – contains instructions for creating and configuring the Oracle database for the Content Server environment.
- ❑ Download, also, the following guides to help you prepare for integrating Analytics with Content Server users and CM sites:
  - *Content Server Administrator's Guide* – contains instructions for creating and assigning roles.
  - *Content Server Developer's Guide* – contains instructions for using CatalogMover.

FatWire Analytics Silent Installer

The FatWire Analytics silent installer is a Java-based script (developed on Ant) that installs Analytics. The silent installer is provided in the FatWire Analytics Kit.

- ❑ Make sure the currently supported version of Ant (required by the silent installer) is running on each server where the silent installer, itself, will be running.
- ❑ Familiarize yourself with the installation scenarios that are covered in this guide and select the scenario that is appropriate for your operations. The scenarios are:
  - Single-server installation: Figure 1, on page 11
  - Dual-server installation: Figure 2, on page 12
  - Enterprise-level installation: Figure 3, on page 13
- ❑ **Note:** The silent installer script installs FatWire Analytics ***locally*** (on the computer where it is executed) and non-interactively. A silent installation involves all the steps from preparing the installation folders and setting up the database to deploying the web applications and utility programs.

FatWire Analytics Supporting Software

- Databases
- Application Servers

Databases

- ❑ Install the Oracle database management system (DBMS) and the SQL Plus utility. Analytics schema will be installed on the Oracle database by SQL Plus. (If you need installation instructions, refer to the product vendor's documentation.)
- ❑ Create and configure an Oracle database as the Analytics database.

  If your Content Server installation runs on Oracle DBMS, you can use the same DBMS to create a database for FatWire Analytics, assuming the server has the capacity to support an additional database. Space requirements depend on the amount of site traffic data you expect to capture within a given time frame, the volume of statistics that will be computed on the captured data, and whether you plan to archive any of the raw data and statistics.

  The steps for creating and configuring an Oracle database are given below:

  1. Follow the procedures in our guide, *Configuring Third-Party Software.*

     Note

     - When setting the Global name and SID, do not create names longer than 8 characters.
     - When creating the user, create the `analytics` user.

  2. Set the encoding to Unicode (`AL32UTF8`). Change the environment variable `nls_lang` to: `NLS_LANG=AMERICAN_AMERICA.AL32UTF8`, using one of the following commands:
     - In Windows, enter the command

set NLS_LANG=AMERICAN_AMERICA.AL32UTF8

- In Linux, the command depends on the shell you are using:

  For Korn and Bourne shells:
  NLS_LANG=AMERICAN_AMERICA.AL32UTF8
  export NLS_LANG
  For C shell:
  setenv NLS_LANG AMERICAN_AMERICA.AL32UTF8

Application Servers

- ❑ Install a supported application server to host the Analytics web applications (i.e., data capture, administrator application, and reporting application). For the list of supported application servers, see the *Supported Platform Document*.

> **Note**
>
> A single-server installation requires a single application server.
>
> A multi-server installation requires up to three application servers, depending on its configuration (for example, three application servers if the data capture application, administrator application, and reporting application are installed on separate computers).

- ❑ Make sure that each application server provides a JDBC driver that works with the Analytics database. (Analytics does not ship with a JDBC driver.)

All Application Servers

- ❑ Configure each application server for UTF-8 character encoding.

> **Note**
>
> The application server's encoding setting must match the value of the `encoding` parameter in `global.xml`. The value is `UTF-8`.

- In Tomcat:

  Edit the file `$CATALINA_HOME/conf/server.xml` and set the `URIEncoding` attribute to UTF-8:

  ```
  <Connector port="8080" URIEncoding="UTF-8"/>
  ```

- In WebSphere:

  Set the value of system property `default.client.encoding` on the JVM settings of the application server to UTF-8.

- ❑ For the application server on the Analytics reporting (and administration) server, set the JVM parameter to:

  ```
  -Djava.awt.headless=true
  ```

- ❑ Enable DNS lookups on your application server. Your DNS server must perform DNS lookups in order for the "Hosts" report to display host names of the machines from

which visitors access your site. For instructions, consult your application server's documentation.

> **Note**
>
> If the application server is not configured to perform DNS lookups, the "Hosts" report will display IP addresses instead (just like the "IP Addresses" report).

JBoss Application Server

- ❑ Delete the common `jar` files from the `lib` folder used by JBoss (in order for the Analytics Administrator application to run).

WebLogic Application Server

- ❑ Add the `log4j jar` file to the `lib` folder for the WebLogic domain in order for the Analytics applications to create log files.
- ❑ Add the `antlr.jar` file to the `PRE_CLASSPATH` in the application server's startup command. For example:

```
C:/bea/wlserver_10.3/samples/domains/wl_server/bin/
   setDomainEnv.cmd
```

If You are Using WebSphere

- ❑ Configure the web application class loader for "parentLast" class loading order.

Environment Variables

- ❑ `Set JAVA_HOME` to the path of the currently supported JDK and the `PATH` variable to `$JAVA_HOME/bin`. These settings are required by Hadoop (`Hadoop-env.sh`), the HDFS Agent, and Hadoop-jobs (all of which, otherwise, will not run).

> **Note**
>
> On Windows, set `JAVA_HOME` to its canonical form:
>
> `C:\PROGRA~1\<path_to_jdk>`
>
> Otherwise, if the path contains spaces (for example, `C:\Program Files`), the path must be enclosed in double quotes (for example, `"C:\Program Files"`).

- ❑ On Solaris systems, add the following line to `hadoop-env.sh`:

```
export PATH=$PATH:/usr/ucb
```

- ❑ Set `ANT_HOME` (required by the silent installer) to the correct path.

Support for Charts

- ❑ The Swiff Chart Generator is used to render charts within Analytics reports. Install the Swiff Chart Generator either on the FatWire Analytics host (single-server installation), or on the reporting server (in multi-server installations. The reporting server hosts `analytics.war`.)

Copies of the Swiff Chart Generator can be purchased at:

```
http://www.globfx.com/
```

Evaluation copies are available at:

```
http://www.globfx.com/downloads/swfchartgen/
```

- ❑ Install Adobe Flash Player on the computers on which reports will be viewed. A free copy of Adobe Flash Player is available at:

  ```
  http://www.adobe.com/go/getflashplayer
  ```

  If you choose not to install Adobe Flash Player, you can still generate reports. However, the charts they might contain will be replaced by the **download plugin** link.

Pre-Integration Checklist

When Analytics is installed, you will integrate it with Content Server users and CM sites. You will also enable data capture for selected pages of the published site.

Experience

- ❑ Integrating Analytics with Content Server requires a knowledge of Content Server's administrative interface. Enabling data capture also requires experience with Catalog Mover and writing Java/JSP templates. If you plan to use Engage, you must have a thorough understanding of recommendations and segments.

Content Server Parameters

- ❑ In Chapter 5, you will integrate FatWire Analytics with Content Server to enable the system and its users for report generation. To complete the Analytics-CS integration steps, you will need certain information. Have the required information available (the information can be obtained from the Content Server administrator):
  - Which users must be given access to FatWire Analytics.
  - Which roles must be assigned to the users.
  - Which Content Server asset types on which sites must be registered with (identified to) FatWire Analytics in order to enable the generation of reports in the "General Information," and "Content Information" report groups.
  - If Engage is installed, which segments and recommendations must be enabled for data capture.
- ❑ In Chapter 6, you will enable data capture for published pages. Make sure you know which Content Server pages to tag for data capture. Untagged pages will be ignored by FatWire Analytics (usage data will not be captured on assets that are displayed on those pages; data on internal searches will not be captured; visitor information will not be captured).

Next Step

Install FatWire Analytics, using the silent installer. For instructions, see Chapter 3, "Installing FatWire Analytics."

Chapter 3

Installing FatWire Analytics

This chapter contains procedures for using a script to install FatWire Analytics on the Content Server web application.

This chapter contains the following sections:

Overview of the FatWire Analytics Silent Installer

The silent installer is a Java-based script, developed on Ant. The script installs FatWire Analytics ***locally*** (on the computer where it is executed) and non-interactively.

The more common installation scenarios are covered in this chapter. They are:

- Single-server: Installing Analytics and its database on a single server (Figure 1, on page 11)
- Dual server: Installing Analytics and its database on separate servers (Figure 2, on page 12)
- Enterprise-level: Analytics in fully distributed mode (Figure 3, on page 13)

A silent installation involves all the steps from preparing the installation folders and setting up the database to deploying the web applications and utility programs. The remaining sections of this chapter, starting with "Installation Steps," on page 28, guide you through the steps that you need to complete in order to run the silent installer. Below is a summary.

Installation Summary

Briefly, you will do the following to install FatWire Analytics, **after ensuring that prerequisites (in** chapter 2**) are satisfied**:

1. Unzip Analytics:
   a. Unzip Analytics on the master node. (In a distributed installation, unzip Analytics on other relevant nodes, including the server that hosts SQL Plus.)

> **Note**
>
> - The silent installer is packaged with the Analytics product.
> - The silent installer must be executed on the SQL Plus host in order to update the Oracle database with Analytics-specific schema.

   b. Unzip Analytics on Content Server's host.

> **Note**
>
> The silent installer must be executed on Content Server's host in order to initialize Content Server to Analytics. You will help to initialize by specifying the location of Content Server's `futuretense_xcel.ini` property file, so that it can be modified by the silent installer with Analytics-specific settings.
>
> Initializing Content Server enables the **analytics** link (in the upper left-hand corner of the Advanced and Dash interfaces) and the `AddAnalyticsImgTag` to capture data on site visitors. For more information about the `AddAnalyticsImgTag`, see Chapter 6, "Configuring Data Capture."

2. Customize the `analytics-build.properties` file for Analytics. Customize the file on every server where the Analytics product was unzipped) to declare information that the silent installer needs in order to correctly deploy the Analytics product in your environment, and to initialize FatWire Analytics to Content Server (as explained in the note above).
3. Prepare to run the installer by checking environment variables and classpaths to make sure they are properly set.
4. Install and deploy Analytics by running the silent installer on every server where the Analytics product is unzipped.
5. Initialize Content Server to Analytics by customizing the `analytics-build.properties` file on Content Server's host.

Silent Installer Actions

When the silent installer starts running, it performs the following steps:

1. Prepares Analytics product folders to store all the installed components, ready for your use.
2. Prepares a separate subdirectory for the Analytics/Hadoop job scheduling system.
3. Sets up and prepares a separate subdirectory for the `hdfsagent` utility, required by the `sensor.war` web application.
4. Customizes various configuration files (shown in Appendix C), using the values that you specified in the `analytics-build.properties` file:
   - Renames `global.xml-dist` to `global.xml` and sets its properties to the values that you specified in `analytics-build.properties`.
   - Renames `log4j.properties-dist` to `log4j.properties` and sets its properties to the values that you specified in `analytics-build.properties`.
   - Customizes properties in `futuretense_xcel.ini` (one of Content Server's property files).
5. Updates your Oracle database with the Analytics database schemas by using SQL Plus and running the following scripts: `create_sys.sql`, `create_normal.sql`, and `region.sql`
6. Unpacks the `reports.zip` archive and places the files in a subdirectory (on the local file system) referenced by the reporting engine.
7. Auto deploys the Analytics web applications (`sensor.war`, `analytics.war`, and `analyticsadmin.war`) to your designated application server.
8. The silent installer on Content Server's host updates the Content Server system, allowing it to determine that Analytics is installed. This update enables the **analytics** link, displayed in the upper left-hand corner of the Advanced and Dash interfaces.

Installation Steps

Note

Before starting the steps in this section, make sure that all prerequisites listed in chapter 2 are satisfied.

Steps for installing Analytics are the following:

Step 1. Unzip Analytics

The Analytics product, `analytics2.5.zip`, contains the silent installer.

To unzip Analytics

1. Complete one of the following steps, depending on the type of installation you plan to create:
   - **Single-server installation** (Figure 1, on page 11). In this scenario, you are installing Analytics on its own server (which hosts all Analytics supporting software). Unzip `analytics2.5.zip` on the server.
   - **Dual-server installation** (Figure 2, on page 12). In this scenario, you are installing Analytics and its database on their own servers.
     1) Unzip `analytics2.5.zip` on the Analytics server.
     2) If SQL Plus is installed on the database server's host, unzip `analytics2.5.zip` on the database server's host. (To enable communication between SQL Plus and the Oracle database, the silent installer must be run on the server that hosts SQL Plus.)
   - **Enterprise-level installation** (Figure 3, on page 13). In this scenario, you are installing Analytics in fully distributed mode. Unzip `analytics2.5.zip` on the master node and on each remaining server where you wish to install Analytics components.

     For example, if your load balancing scheme requires multiple Analytics Sensors (data capture applications) to be deployed on different data capture nodes, unzip `analytics2.5.zip` on all the data capture nodes.

     Note

     To enable SQL Plus to update the Oracle database with Analytics-specific schema, make sure that `analytics2.5.zip` is unzipped on the server that hosts SQL Plus.

2. Unzip `analytics2.5.zip` on Content Server's host.
3. Continue to "Step 2. Customize analytics-build.properties for Analytics."

Step 2. Customize analytics-build.properties for Analytics

The `analytics-build.properties` property file contains all the environment-specific configuration data that is required by the Analytics silent installer. In this step, you will customize `analytics-build.properties` in order to provide installation specifications to the silent installer. (The `analytics-build.properties` file is divided into sections. Each section is specific to certain information necessary to tailor the system for your use.)

To customize analytics-build.properties for Analytics

Complete the following steps on each server where you plan to install Analytics (or its components):

1. Back up `analytics-build.properties` (located in the root of the Analytics Kit).
2. Open `analytics-build.properties` in a text editor of your choice, and set the properties as indicated in the following sections:
   - General Installation Properties
   - Visitor Detection Properties
   - System Configuration and Operation Defaults
   - Web Server URL Properties
   - Application Server Deployment Properties
   - Database Connection Properties
   - Hadoop Properties

Note

Windows Only. When you specify the path to an installation directory, be sure to enclose the path in double quotes ***if it contains spaces***.

General Installation Properties

This group of properties provides information of a generalized nature for the installation process. In this section, you will specify paths to Analytics directories, the location of third-party components, and the email addresses of administrators who can reset passwords and create accounts.

Table 1: General installation properties in `analytics-build.properties`

Property	Description
`Analytics.installation.path`	Absolute path to the final Analytics installation directory.

Table 1: General installation properties in `analytics-build.properties`

Property	Description
`swchart.instdir`	Absolute path to the directory where the Swiff Chart product is installed. **Note:** The path must end with a slash (on Unix) or backslash (on Windows).
`forgotpassword.value`	Email address for users to request their password, if forgotten. The email address is the value for the link **Forgot Your Password?** in the Analytics login form. (The email address that you set here, for the Analytics application, should match the address for the link **Forgot Your Password?** in the Content Server login form. In Content Server systems, the email address is set in the `forgotpassword` property, which is located in the `uiadmin.properties` file. For more information, see the *Content Server Property Files Reference*.)
`noaccount.value`	Email address for users to request an Analytics account. The email address is the value for the link **Don't Have an Account?** in the Analytics login form. (The email address that you set here, for the Analytics application, should match the email address for the link **Don't Have an Account?** in the Content Server login form. In Content Server systems, the email address is set in the `forgotpassword` property, which is located in the `uiadmin.properties` file. For more information, see the *Content Server Property Files Reference*.)
`href.help.value`	URL at which help relating to FatWire Analytics can be obtained: `http://e-docs.fatwire.com`

Visitor Detection Properties

Visitor detection is done by the Analytics Sensor (data capture application). By default, the `analytics-build.properties` file supports the `Sessionfingerprint` method of tracking visitors across all sites. The `Sessionfingerprint` method identifies each visitor by a combination of the IP address, screen resolution, and agent string.

Using Table 2, on page 31, verify or set (as indicated) the visitor detection properties in `analytics-build.properties`. The properties specify how identifiers are generated by the system for inclusion in object impressions (which are captured and processed by the

system). Identifiers are important to providing the correct grouping of object impressions for aggregation.

Note

An **object impression** is a single invocation of the `sensor` servlet. For more information, see "Object Impressions," on page 94.

Table 2: Visitor detection properties in `analytics-build.properties`

Property	Description
`sessionIdGenerator`	Specifies the ID generator that is used to identify sessions. **Caution:** The default value is `AppServerID`. Do not modify this value. It is a reference to the object that generates the ID.
`visitorIdGenerator`	Specifies the ID generator that is used to identify a visitor to the site. **Note:** The default value is `SessionfingerprintId`. The default value generates an identifier that is a combination of IP address, screen size, and agent (browser type). If the default generator is insufficient, you have two other options by which to uniquely identify a visitor: self-organized detection and cookie method. If you wish to implement these options or refine the `Sessionfingerprint` configuration (to detect visitors on selected sites), you can do so after Analytics is installed, by modifying `global.xml` directly. Instructions are available in Appendix D, "Configuring Visitor Detection."

System Configuration and Operation Defaults

Verify that the `encoding` property in `analytics-build.properties` is set to `UTF-8` and the application server's setting is set to the same value. The silent installer will set the `encoding` parameter in `global.xml` to the value that you provide in `analytics-build.properties`.

Set the following properties related to data processing and archiving, as necessary:

Table 3: Data processing and archiving properties in `analytics-build.properties`

Property	Description
`midnight.offset`	Allows the system to derive relative midnight used for file rotation. Relative midnight and the `session.rotate.delay` determine when the daily cycle for capturing session data ends. (Information about `session.rotate.delay` can be found on page 129.) **Format:** minutes **Default value:** `0`
`cs_enabled`	Specifies whether buttons for navigating to the Content Server interface are enabled or disabled in the Analytics interface. **Default value:** `true`
`archive.enabled`	Specifies whether HDFS Agent archiving of raw data files is enabled. If `archive.enabled` is set to either `true` or `false`, the `data.txt` file will be deleted from the analytics root folder. To enable archiving, set this property to `true`. Once archiving is enabled, HDFS Agent will automatically create archives of raw Analytics data on a periodic basis by moving `data.txt` to the archiving folder. (The archive directory and start time are specified in the following properties: `archive.output.dir` and `archive.start.time`) **Default value:** `false`
`archive.output.dir`	Specifies the path to the directory for storing archived data files. Must be a valid URI. **Sample value:** • **Windows:** `archive.output.dir=file://d:/archive` • **Linux:** `archive.output.dir=/analytics/archive` **Format:** directory path
`archive.start.time`	Specifies the start time (*HH:mm*) for archiving raw data. The HDFS Agent will start the archiving task on a daily basis at the time specified in this property. For example, to start archiving at 4:00 PM every day, set: `archive.start.time=16:00` **Format:** 24-hour format, expressed as *HH:mm*, where *HH* ranges from `00-23` and *mm* ranges from `00-59`. **Default value:** `06:00`

Table 3: Data processing and archiving properties in `analytics-build.properties`

Property	Description
`purgejobs.enabled`	Determines when purge jobs will run. When this property is set to `true` the system will automatically schedule cleanup jobs to remove subfolders and files after they have been successfully processed. **Default value:** `false`
`sensor.requestqueue.maxsize`	Specifies `CRITICAL` condition for the Analytics Sensor. This property specifies a threshold value that triggers a `CRITICAL` (red) condition when the sensor cannot respond quickly enough to the amount of raw data that it needs to record. When the threshold is reached or exceeded, the **Analytics Sensor** component is displayed in red. The threshold value for this property is expressed as an object impression, i.e., a single invocation of the `sensor` servlet. (The **Analytics Sensor** component is represented in the **Components > Overview** panel of the Analytics Administration interface, shown in Figure A-1, on page 91.) **Default value:** `10000`
`sensor.requestqueue.warnsize`	Specifies `WARNING` condition for the Analytics Sensor. This property specifies a threshold value that triggers a `WARNING` (yellow) condition when the sensor cannot respond quickly enough to the amount of raw data that it needs to record. When the threshold is reached or exceeded, the **Analytics Sensor** component is displayed in yellow. The threshold value for this property is expressed as an object impression, i.e., a single invocation of the `sensor` servlet. (The **Analytics Sensor** component is represented in the **Components > Overview** panel of the Analytics Administration interface, shown in Figure A-1, on page 91.) **Default value:** `3000`

Web Server URL Properties

In this section, you will specify the Content Server URL and the basic URL (`http://<address>:<port>`) where each Analytics web application will reside after the installation is complete.

Table 4: Web server URL properties in `analytics-build.properties`

Property	Description
`analytics.sensor.web.server`	URL of sensor web application.
`analytics.report.web.server`	URL of Analytics web application.
`analytics.admin.web.server`	URL of Analytics administration web application.

Application Server Deployment Properties

In this section, you will specify which application server the installer will use to deploy the Analytics web applications. You will also disable the unused application servers.

Note

- For the application server that will be used, set its `install` property to `true`. Set all other relevant properties.
- For each unused application server:
  - Do not delete the statement pertaining to the application server.
  - Verify that the application server's `install` property is set to `false` (the default). Otherwise, the installer will try to deploy the Analytics web applications to that server.

Table 5: WebLogic deployment properties in `analytics-build.properties`

Property	Description
`install.weblogic`	Set to `true` to deploy to WebLogic.
`weblogic.userid`	WebLogic admin user id.
`weblogic.password`	Password for the WebLogic admin user specified above.
`weblogic.targets`	Server name, cluster name, or virtual host name.
`weblogic.admin.url`	URL to the admin function in WebLogic.
`weblogic.home.dir`	Path to the WebLogic home directory.

Table 6: JBoss deployment properties in `analytics-build.properties`

`install.jboss`	Set to `true` to deploy to JBoss.
`jboss.deploy.dir`	Path to the JBoss deployment directory.

Table 7: Tomcat deployment properties in `analytics-build.properties`

Property	Description
`install.tomcat`	Set to `true` to deploy to Tomcat.
`tomcat.home.dir`	Path to the Tomcat home directory.

Table 8: WebSphere deployment properties in `analytics-build.properties`

Property	Description
`install.websphere`	Set to `true` to deploy to WebSphere.
`websphere.userid`	WebSphere admin user id.
`websphere.password`	WebSphere admin password.
`websphere.home.dir`	WebSphere home directory.
`websphere.node.name`	WebSphere node name.
`websphere.base.command`	WebSphere `wsadmin` command.
`websphere.save.command`	WebSphere save command.
`websphere.engine.instdir.value`	Location of the Analytics reporting engine installation directory.
`websphere.cell.name`	Namespace that has been defined to represent a single node (machine instance) or multiple nodes where a software component is distributed and run. When installing an Analytics application on the WebSphere application server, you must specify how the application is to be distributed, by specifying the name of the cell and node where the application will be installed and run.
`websphere.server.name`	Name of the server within the WebSphere installation where the application is deployed.

Database Connection Properties

In this section, you will specify information that the installer will use to access the Oracle database to store Analytics data. You will also specify JDBC/JNDI information to be placed in the Analytics configuration files. JDBC and JNDI data are mutually exclusive. Only one of them must have its `enabled` property set to `true`.

Table 9: Database properties in `analytics-build.properties`

Property	Description
`install.database`	Set to `true` to run install schema queries.
`db.home.dir`	Database home directory.
`db.sys.user`	System user name.
`db.sys.password`	System user password.
`db.host`	Host address.
`db.port`	Host port number.
`db.sid`	Database SID.

Table 10: JDBC database writer properties in `analytics-build.properties`

Property	Description
`jdbc.enabled`	`True` | `false` Set to `true` to configure JDBC settings.
`jdbc.name.value`	Name of the database connection.
`jdbc.default.value`	There must be exactly one connection marked with `default="true"`
`jdbc.type.value`	Type of connection: `jdbc`
`jdbc.classname.value`	JDBC driver class.
`jdbc.url.value`	JDBC URL
`jdbc.user.value`	JDBC attribute. Database user name
`jdbc.password.value`	JDBC attribute; database password

Table 11: JNDI properties in `analytics-build.properties`

Property	Description
`jndi.enabled`	`True` | `false` Set to `true` to configure JNDI settings.
`jndi.name.value`	Name of the database connection.

Table 11: JNDI properties in `analytics-build.properties`

Property	Description
`jndi.default.value`	There must be exactly one connection marked with `default="true"`
`jndi.type.value`	Type of connection: `resource`
`jndi.resource.value`	JNDI attribute. JNDI name

Hadoop Properties

In this section, you will provide information about your Hadoop configuration:

- The base path to the Hadoop installation directory
- Paths to raw data. One path specifies where, on the local file system, raw data will be recorded by the Analytics Sensor (data capture application). The other path specifies where, on the Hadoop distributed file system, the raw data will be written by the HDFS agent.

Table 12: Hadoop properties in `analytics-build.properties`

Property	Description
`hadoop.installation.path`	Path to the Hadoop installation directory.
`hadoop.hdfs.defaultfs`	The default path in HDFS for writing Analytics raw data.
`hadoop.tasktracker.url`	URL of the Hadoop task tracker web application.
`hadoop.filesystem.url`	URL of the Hadoop file system web application.
`logwriter.output.path`	Local file system path where the data capture application will record raw data.

Step 3. Prepare to Run the Silent Installer

Before running the silent installer, complete the following steps on all servers where you unzipped the silent installer:

1. Make sure that `JAVA_HOME` and `ANT_HOME` in your environment are set to the correct paths.
2. Start all the Analytics application servers (they must be running for the silent installer to work).
3. If you are using Unix, issue the following command to ensure that the next command (to run the installer) can be executed:

   ```
   chmod +x analytics_install.sh
   ```

4. If you are installing on WebLogic, run one of the following commands to ensure that the classpath is properly set and the WebLogic deployment task will run correctly:
   - Unix:
     ```
     <WL_HOME>/server/bin/setWLSEnv.sh
     ```
   - Windows:
     ```
     <WL_HOME>\setWLSEnv.bat
     ```

Step 4. Install Analytics

In this step, you will run the silent installer, which will install the Analytics product against the `analytics-build.properties` file, in which you specified:

- Where Analytics will be installed on your file system.
- Which application server you wish to use and where it exists.
- The location of your database.
- The location of your FatWire Content Server.
- Where other associated products or software components are located on your system.

Single-server Analytics installation

In this scenario, you will install the entire Analytics product on the same server that runs all Analytics supporting software.

1. Run the silent installer:
   - Unix:
     ```
     ./analytics_install.sh
     ```
   - Windows:
     ```
     analytics_install.bat
     ```

   Note

   For descriptions of the components that are installed, see Table 13, on page 40.

2. When the installer has successfully completed its task, initialize Content Server to Analytics. Go to "Step 5. Initialize Analytics to Content Server," on page 42.

   Note

   For information about the events that occur when the silent installer runs, see "When the analytics_install Command Begins Executing," on page 41.

Dual-server Analytics installation

In this scenario, you will install the Analytics product on the server that hosts Analytics supporting software, except for the database. The Analytics database is installed on its own server.

1. Run the silent installer on the server where you wish to install Analytics:
   - Unix:

     ```
     ./analytics_install.sh sensor hadoopjobs analytics
        analyticsadmin cs_integration verify_install
     ```

   - Windows:

     ```
     analytics_install.bat sensor hadoopjobs analytics
        analyticsadmin cs_integration verify_install
     ```

   Note

   For information about the events that occur when the silent installer runs, see "When the analytics_install Command Begins Executing," on page 41.

2. On the database server, where SQL Plus is already installed, run the silent installer. (Running the silent installer initializes the Oracle database with Analytics schema, as explained in Table 13, on page 40,`database` target):
   - Unix:

     ```
     ./analytics_install.sh database
     ```

   - Windows:

     ```
     analytics_install.bat database
     ```

3. When the installer has successfully completed its task, initialize Content Server to Analytics. Go to "Step 5. Initialize Analytics to Content Server," on page 42.

Distributed Analytics installation

For a distributed installation, you will install different parts of the Analytics product on the servers where you unzipped `analytics2.5.zip`. You will run the **`analytics_install`** command and specify which part (i.e., Ant target) to install on the given server. If your selected target has dependencies on supporting systems, the silent installer will verify that the supporting systems are installed (and running in the case of the Analytics database). Table 13, on page 40 lists each target that is recognized by the silent installer and its dependency on other systems.

To install Analytics in distributed mode

1. Install Analytics:

   Run the installer on each Analytics server where you wish to install an Analytics component:
   - Unix:

     ```
     ./analytics_install.sh <Ant_target>
     ```

- Windows:
  `analytics_install.bat <Ant_target>`

Note

- The **`<Ant_target>`** parameter can be an individual Ant target or a space-separated list of targets, defined in Table 13. To install targets one at a time, run the silent installer for each target. If you do not specify a target, all Analytics components (defined in Table 13) will be installed on the given server.
- For information about the events that occur when the silent installer runs, see "When the analytics_install Command Begins Executing," on page 41.

2. When the installer has successfully completed its task, initialize Content Server to Analytics. Go to "Step 5. Initialize Analytics to Content Server," on page 42.

Table 13: Ant targets and their dependencies

Ant Target	Description	Server on which to Unzip Silent Installer	Dependencies
`sensor`	Installs the data capture web application and HDFS Agent (FatWire-specific).	Data Capture	
`hadoopjobs`	Installs the FatWire-specific Analytics/ Hadoop job scheduler.	Master Node	Oracle
`database`	Updates the Oracle database with Analytics schema. The `database` Ant target loads Analytics-specific database definitions into SQL Plus. SQL Plus then updates the Oracle database with the definitions, and so initializes the database with the schema required by Analytics.	Server on which SQL Plus is installed	Oracle
`analytics`	Installs the Analytics reporting engine web application.	Administration, Reporting	Oracle
`analyticsadmin`	Installs the Analytics administration web application.	Administration, Reporting	Oracle
`cs_integration`	Modifies `futuretense_xcel.ini` (located on the Content Server system. Content Server does not have to be running in order for the installer to modify `futuretense_xcel.ini`). The silent installer locates `futuretense_xcel.ini` against the `cs.local` property (which you set during the installation process, itself).	Content Server's Host	None
`verify_install`	Provides an object impression used to verify Hadoop Jobs. **Note:** You will use this target when verifying the Analytics installation in chapter 4.	Data Capture	None

When the analytics_install Command Begins Executing

1. When the `analytics_install` command begins executing, you will see the FatWire Analytics License agreement appear on your console. Be sure to read this important document and accept it by responding to the console prompt. The silent installation will not continue until you accept this license agreement. **If you do not accept the license agreement, the silent installer will immediately terminate.**
2. When the installation process begins, the script performs the steps listed in "Silent Installer Actions," on page 27. When the installation process completes successfully, the following message is displayed:

```
The silent install has finished.
-----------------------------------------------------------------
Your installation of Analytics is complete. Please review your
application server documentation and make sure that it is
configured for UTF-8 encoding.
-----------------------------------------------------------------
```

Note

If an error occurs, the installation process terminates and displays the following message:

```
"The install script ended with error code nn. Please
 consult the log and check for errors."
```

If an error is reported on the console, inspect the `analytics-install.log` to identify the problem (the `analytics-install.log` file is located in the directory where the silent installer resides). Typically, problems arise when one or more properties have been incorrectly configured. Carefully review and correct any property which is not correct, then rerun the silent installer.

If an error occurs due to failure to find a necessary component, make the appropriate adjustments to your `JAVA_HOME`, or `ANT_HOME` environment settings, and rerun the silent installer.

Step 5. Initialize Analytics to Content Server

Once Analytics is successfully installed, complete the steps below on Content Server's host.

Note

Content Server does not have to be running in order for the steps in this section to be completed successfully.

1. Specify the location of Content Server's `futuretense_xcel.ini` property file, so that it can be modified by the silent installer with Analytics-specific settings.

   Open `analytics-build.properties` and modify the following properties:

   Table 14: Properties that update `futuretense_xcel.ini`

Property	Description
`cs.local`	Specifies that Content Server is local (relative to the silent installer). **Verify that this property is set to `true`.**

`src.ini.file`	Specifies the location of Content Server's `futuretense_xcel.ini` property file.
	Sample values:
	Windows:
	`C:/JSK_060809\RunTime\ContentServer/7.5.1/futuretense_xcel.ini`
	Linux:
	`/home/fatwire/RunTime/contentServer/7.5.1/futuretense_xcel.ini`

`mod.ini.file`	Specifies the path to Content Server's `futuretense_xcel.ini` file. (The path is identical to the path in the `src.ini.file` property.) Confirms to Content Server that Analytics is installed. This property modifies `futuretense_xcel.ini` (according to the settings you specify in `analytics-build.properties`), and writes the modified `futuretense_xcel.ini` file to the specified location. The original `futuretense_xcel.ini` file (specified by `src.ini.file`) is automatically backed up. **Sample value:** Use the value that is set in `src.ini.file`.

2. Run the silent installer on Content Server's host:
   - Unix:

     ```
     ./analytics_install.sh cs_integration
     ```

   - Windows:

     ```
     analytics_install.bat cs_integration
     ```

3. When the silent installer completes its task, continue to "Next Steps," on page 43.

Next Steps

- Before verifying the Analytics installation, complete the following steps, as necessary:
  - Inspect the configuration files that have been updated by the silent installer. Refer to Appendix C, "FatWire Analytics Configuration Parameters." This appendix also contains instructions for setting up logging for the Hadoop Job scheduler.
  - Modify the visitor tracking configuration. Refer to Appendix D, "Configuring Visitor Detection" for instructions.
- To complete the Analytics installation, follow the steps in Chapter 4, "Verifying Your FatWire Analytics Installation."

Chapter 4

Verifying Your FatWire Analytics Installation

This chapter contains procedures for verifying the operation of the newly installed FatWire Analytics application and its components.

This chapter contains the following sections:

Verification Steps

In this section, you will start Hadoop, the HDFS Agent, and Hadoop Jobs and then complete the following steps:

- Verify that data capture works:
- Verify the HDFS Agent:
- Verify Hadoop Jobs:
- Verify that the Analytics reporting interface can be displayed:
- Verify that the Analytics administrator interface can be displayed:

To verify FatWire Analytics

1. Start Hadoop.
2. Start the HDFS Agent (execute `run.sh`, located in the `bin` folder):

   ```
   cd ${ANALYTICS_HOME}/hdfsagent/bin
   chmod +x run.sh
   ./run.sh
   ```

   where **`ANALYTICS_HOME`** is the Analytics installation directory.
3. Before starting Hadoop Jobs, do the following to ensure uninterrupted operation of Hadoop Jobs:
   a. Edit the file `/etc/security/limits.conf`
   b. Change the value of `soft nofile` and `hard nofile` to `65536`.
   c. Open a new shell where Hadoop will be run. Make sure `ulimit -n` is at least `1024`.
4. Start Hadoop Jobs (execute `run.sh`, located in the `bin` folder):

   Note

   The JVM must have at least 512MB of memory in order for the Hadoop Job scheduler to run when Hadoop starts. If memory is insufficient, add the memory parameter `-Xmx512m` to the `run.sh` file in Hadoop Jobs.

   As the Hadoop Job scheduler is the master of the HDFS, all HDFS Agents must be restarted when you restart the scheduler.

   ```
   cd ${ANALYTICS_HOME}/hadoop-jobs/bin
   chmod +x run.sh
   ./run.sh
   ```
5. Verify that data capture works:
   a. Make sure that the application server is running on the server where the Analytics Sensor is installed.
   b. Run the `verify_install` utility to access the URL of the Analytics Sensor. On a multi-server installation, run the `verify_install` utility on all the data capture servers. (The `verify_install` utility contains a sample object impression for testing Hadoop Jobs.)

- Unix:

  ./analytics_install.sh verify_install

- Windows:

  analytics_install.bat verify_install

The system returns a preview thumbnail (as a 1x1-pixel `gif` file). The incoming raw data is written by the Analytics Sensor to the `data.txt.tmp` file (see Figure 4, on page 47).

Figure 4: Storage of Captured Data

```
[root@localhost bin]# tree -A /analytics/data/
/analytics/data/
├── oirawdata
│   └── 2009
│       └── 01
│           └── 14
│               ├── Sensor-localhost.localdomain-10-1231924822209
│               │   └── data.txt
│               └── Sensor-localhost.localdomain-10-1231924931759
│                   └── data.txt.tmp
└── sesrawdata
    └── 2009
        └── 01
            └── 14
                └── Sensor-localhost.localdomain-10-1231924822216
                    └── data.txt.tmp
```

The `data.txt.tmp` file is stored in the local file system, in a folder. The folder is specified in the `logwriter` element (`rootpath` attribute) in `global.xml` (see Table C-3, on page 132). The sensor will rotate the `data.txt.tmp` file to `data.txt` when either the threshold interval is reached (see the `sensor.threshold` property on page 128), or the application server is restarted.

For example, if the folder is specified as `<logwriter type="LFS" name="LFS" **rootpath="C:/analytics/data"** />`, then the raw captured data will be written to the folder structure shown in Figure 4, starting with the **`/analytics/data/`** folder. The raw data is collected into `data.txt.tmp` and, after rotation, stored in `data.txt`. The folders named `oirawdata` and `sesrawdata` are system defaults.

6. Verify the HDFS Agent:

   a. Make sure that the HDFS Agent has successfully copied `data.txt` (and its directory structure) from the local file system to HDFS. The HDFS Agent handles `data.txt` as follows:

      1) Copies the `data.txt` file from the local file system to a folder in HDFS. The HDFS folder is specified by the `hadoop.hdfs.defaultfs` parameter in `global.xml` (see Table C-1, on page 125).

      2) When `data.txt` is successfully copied, it is removed from the local file system.

   b. To verify the actions of the HDFS Agent and inspect the file system, open the Hadoop HDFS web interface, running on your master node (`http://hostname_MasterNode:50070/`) and browse the folder.

7. Verify Hadoop Jobs:

   Hadoop Jobs processes the data that is copied by HDFS Agent from the local file system to HDFS. To view the status of the jobs, open the JobTracker web interface URL:

   ```
   http://<hostname_MasterNode>:50030/
   ```

   In the JobTracker web interface, the "RUNNING Jobs" section displays the `OIProcessor` jobs. After the `OIProcessor` jobs are completed, `OIInjection` job will start and will insert the data into the Analytics database.

   Note

   A series of session and visitor jobs will also run. When the jobs are completed, their status will be reported in the "Status Summary" panel of the Analytics Administrator application. You will verify the jobs in step 9, when you log in to the Administrator application.

8. Verify that the Analytics reporting interface can be displayed:

   Log in to the Analytics reporting application at the following URL, with user name `csuser` and password `csuser`:

   ```
   http://<hostname:port>/analytics
   ```

   where `<hostname>` is the host name of the server on which the reporting application is installed.

   Note

   The `csuser` (password `csuser`) is a default that ships with Analytics. This user has administrative rights in Analytics. Log in as `csuser/csuser` when you need to work in the Analytics administration interface (for example, in chapter 5 you will integrate Analytics with Content Server).

   If the database connection fails, the system will display a message denoting that fact.

9. Verify that the Analytics administrator interface can be displayed:

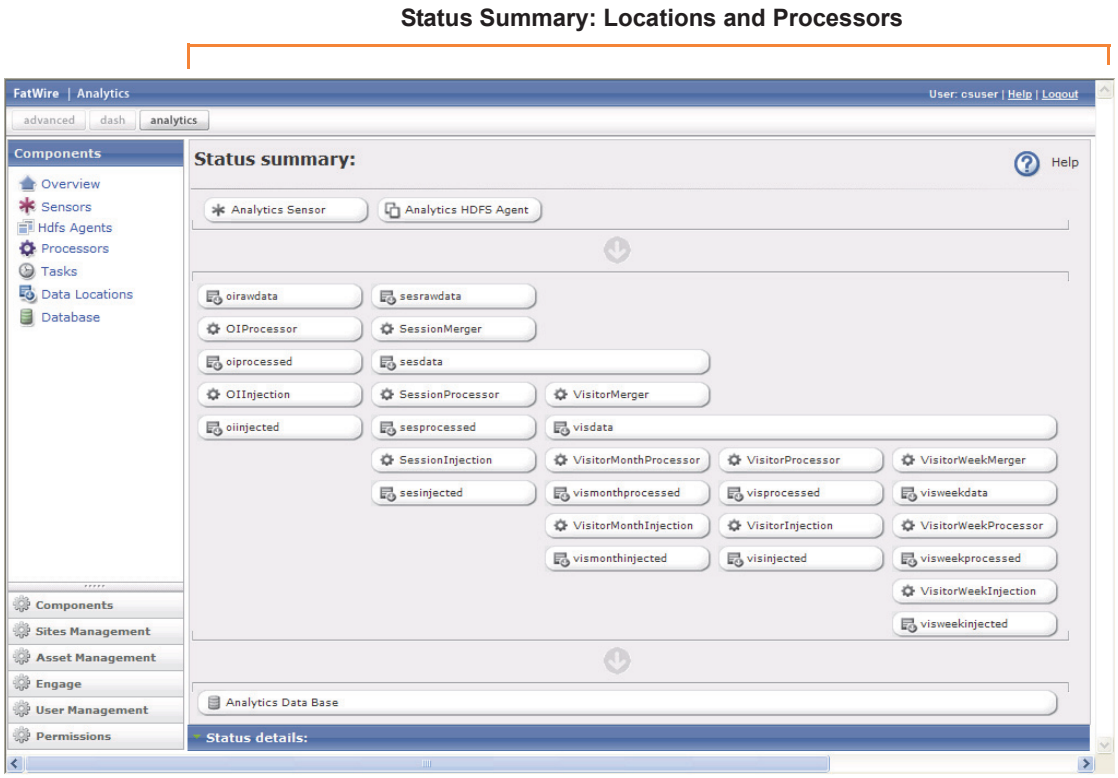
   Log in to the Analytics Administrator application at the following URL, with user name `csuser` and password `csuser`:

   ```
   http://<hostname:port>/analyticsadmin
   ```

   where `<hostname>` is the host name of the server on which the administrator application is installed.

   The opening screen displays the "Status Summary" panel of locations and processors:

Figure 5: Analytics Administrator Interface (**Components > Overview**)



Here, you can monitor the status of various components. For example:

- You can monitor the Analytics Sensor to ensure that it is functioning and responding properly to site traffic. (For more information about monitoring the Sensor and responding to alerts, see "Sensor Overload Alerts," on page 50.)
- You can stop the Analytics Sensor.
- You can verify Hadoop Jobs by clicking on a location or processor to view its status. (For more information about Hadoop Jobs and the "Status Summary" panel, see Appendix A, "Hadoop Jobs: Processors and Locations.")

Recommended Configurations

Installer Files

When you have verified your Analytics installation, keep the installer files on their respective hosts. Should you need to modify operating parameters at a later time, you can rerun the installer. **Modifying operating parameters manually is not recommended.**

Installer files are the following:

- `lib` is the folder that contains all the additional `jar` files needed to support the silent installer when it runs.
- `analytics-build.properties` contains all the properties that define how the Analytics product should be installed on your system.
- `analytics-build.xml` is the Ant build script to perform the silent installation.
- `analytics_install.bat` is the batch file that runs the silent installer in a Windows command window.
- `analytics_install.sh` is the Unix/Linux shell script that runs the silent installer.
- `AnalyticsLICAGRv3.txt` is the text of the FatWire Analytics License agreement, which you must accept before the silent installer will run.
- `analytics-silent-install.jar` contains all the custom Java classes that are required by the silent installer.
- `log4j.properties` is used to configure logging behavior such as output target, type and level of message, and the format of messages at runtime.

Sensor Overload Alerts

During heavy site traffic, the Analytics Sensor can become overloaded with incoming data and stop responding normally. The Analytics Sensor will stop writing to the file system and will instead store incoming data in memory, until an out-of-memory condition is reached.

The Analytics Administrator interface (Figure 5, on page 49) alerts you to an “overload” condition by displaying the **Analytics Sensor** button in either yellow or red. Yellow indicates a `WARNING` condition. Red indicates a `CRITICAL` condition (assuming the sensor is running).

Note

A stopped or non-functional Analytics Sensor is also displayed in red.

Setting an ‘Overload Alert’ Threshold

Properties in `global.xml` determine the threshold that triggers a `WARNING` or `CRITICAL` condition. The properties are sensor. requestqueue. warnsize and sensor. requestqueue. maxsize. Set these properties to a threshold that is compatible with the configuration of

your Analytics installation and the volume of site traffic. For more information about these properties, see page 130.

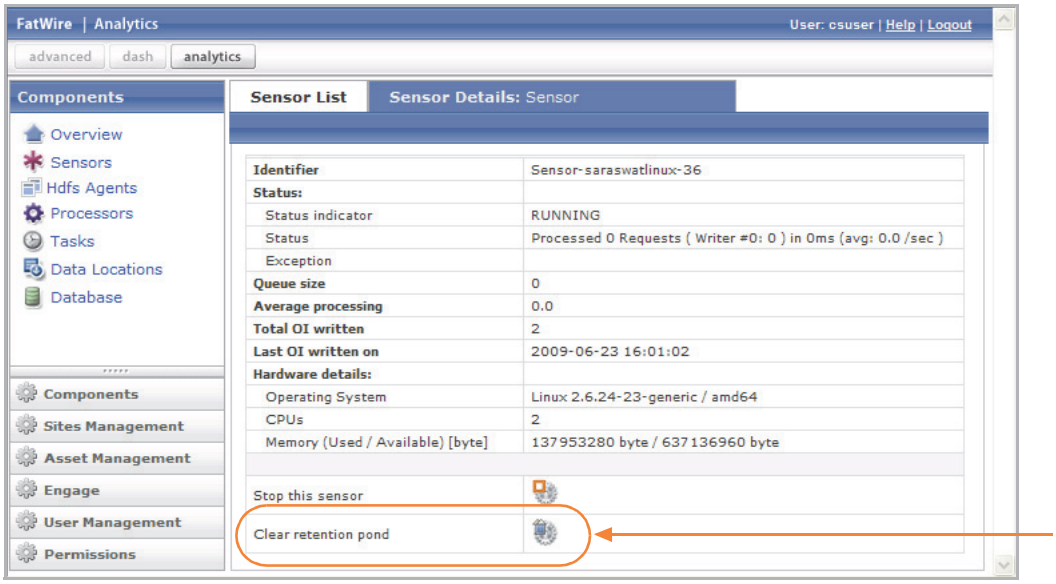
Responding to a “red” Condition

If you are monitoring the “Status Summary” panel (in the Administration interface, Figure 5, on page 49) and you notice that the **Analytics Sensor** button is displayed in red, you need to determine whether the Sensor has stopped, has failed, or is overloaded. In case of overload, you will need to clear the memory in order to reset the system and resume normal functioning. **Data cleared from memory cannot be retrieved and will be lost.**

To respond to a “red” condition

Click the **Analytics Sensor** button and note the main panel.

- If you see `"No data available"`, the Analytics Sensor has either stopped or failed.
- If you see the “Sensor Details” panel, the Analytics Sensor is running, but it is overloaded. Click the icon labeled “Clear retention pond” to clear the memory.



Geolocation Database

GeoLite City is a highly optimized geolocation database provided by MaxMind. GeoLite City is in binary format for performing fast lookups. It is used by FatWire Analytics for the information it contains: country, region, area code, metro code, city, and postal code. The GeoLite City database is updated monthly, at the beginning of each month, by MaxMind.

To install the latest GeoLite City database

1. Download the GeoLite City database in binary format from the following location:

   ```
   http://geolite.maxmind.com/download/geoip/database/
      GeoLiteCity.dat.gz
   ```

2. Uncompress the file:
   - On Windows: Use the winzip or zip program to unzip the file.

- On Unix: Use the `tar` command to uncompress the file:

  ```
  tar xvzf GeoLiteCity.dat.gz
  ```

3. Copy the `GeoLiteCity.dat` file to the `CACHE` folder under the analytics installation folder on the server where the Hadoop Jobs application is installed:

   On Unix:

   ```
   cp GeoLiteCity.dat <ANALYTICS_INSTALL_DIR>/CACHE
   ```

Next Steps

If you have successfully completed all the steps in this chapter, then Analytics is ready to be integrated with Content Server. When integrating, you will configure Analytics users and their permissions to reports. You will also enable reporting options. Continue to Chapter 5, "Integrating Analytics with Content Server."

Chapter 5

Integrating Analytics with Content Server

This chapter shows you how to integrate FatWire Analytics with your Content Server system and how to test and maintain your Analytics integration.

This chapter contains the following sections:

Overview

This section describes the following tasks you will perform as an Analytics administrator:

- First-Time Setup
- Maintenance

First-Time Setup

In order to integrate Analytics with your CS system and its users, you must perform the following steps:

1. Create the `Analytics` role on your CS system and grant it to the appropriate CS users and yourself. For instructions, see "Granting Users Access to Analytics," on page 56.
2. Register CM sites with Analytics. For instructions, see "Registering Sites," on page 57.
3. Configure the "Page Views" report. This report configures the `Pageview` object on which reports in the "General Information" and "Content Information" groups are based. Reports in these groups will not function until you configure the "Page Views" report.
   - If you wish to track the searches that visitors perform on your site, you can configure an internal searches report.
   - You can also configure a custom report.

   For instructions on configuring reports, see "Managing Asset Reports," on page 59
4. (Optional) Enable the performance indicator for assets of the types you selected for tracking by Analytics. For instructions, see "Enabling the Performance Indicator," on page 63.
5. Register CM site users with Analytics. For instructions, see "Registering Users," on page 70.
6. Create user groups. For instructions, see "Creating Groups," on page 72.
7. Grant permissions to user groups. For instructions, see "Managing Permissions," on page 75.
8. Assign users to user groups. For instructions, see "Assigning Users to User Groups," on page 73.
9. If you are using FatWire Engage and want to track recommendations and segments, register these asset types with Analytics. For instructions see the following sections:
   - For recommendations, see "Registering Recommendations," on page 65.
   - For segments, see "Registering Segments," on page 67.

Note

At this point, you can generate the reports you have configured, but the reports will be empty. Your developers must enable data capture for assets of the types you selected for tracking. If you configured a custom report, you must also programmatically develop (1) an Analytics job to statistically process the captured data, and (2) the report that will display the statistics in the desired layout.

Maintenance

Once you have integrated Analytics with your CS system, you can use the procedures in the following sections to maintain your Analytics installation:

- "Granting Users Access to Analytics," on page 56
- "Managing Asset Reports," on page 59
- "Managing the Performance Indicator," on page 62
- "Managing Engage Asset Reports," on page 64
- "Managing Users and Groups," on page 70
- "Managing Permissions," on page 75

Granting Users Access to Analytics

In this step you will create the `Analytics` role, which grants users access to the Analytics installation, and the permissions to access and generate reports. You will then assign the role to users on your Content Server system.

To grant users access to Analytics

1. Log in to Content Server's Advanced interface as the administrator. For a new installation, the default administrator user name and password are as follows:
   - User name: `fwadmin`
   - Password: `xceladmin`

2. Create a role named `Analytics`. For instructions, see the *Content Server Administrator's Guide*.

> **Note**
>
> If the FirstSite II sample site is installed, it is unnecessary to create the `Analytics` role.

3. Assign the `Analytics` role to the CS users who need to have access to Analytics. For instructions, see the *Content Server Administrator's Guide.*
4. Assign the `Analytics` role to yourself (the logged in administrator). You will need this role in order to complete the steps necessary to integrate Analytics with your CS system.

Managing Sites

The first step towards enabling visitor activity tracking on your site is registering the site with Analytics. Once you have registered one or more sites, you can change their registration data, or unregister them (that is, delete them from Analytics).

This section contains the following procedures:

- Registering Sites
- Editing Sites
- Deleting Sites

Registering Sites

This procedure shows you how to register a CM site with Analytics.

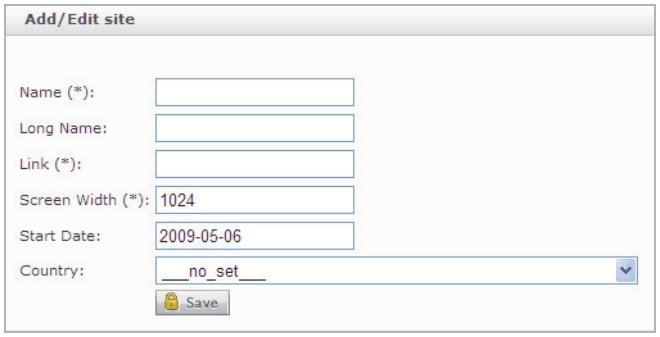
To register a CM site with Analytics

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the "Sites Management" pane on the left, click **Register**.

   Analytics displays the "Add/Edit Site" form:



3. Fill in the form as follows:
   a. **Name** – enter the name of the site, as it appears in the CS admin interface.
   b. **Long name** – enter a short, informative description of the site.
   c. **Link** – enter the URL of the site.
   d. **Screen Width** – enter the target pixel width for which the site was designed.
   e. **Start Date** – specify when Analytics should start capturing data on this site.
   f. **Country** – select the country targeted by the site. The selection you make here is used to calculate the **PV/MIO** statistic in the "Overview" report.
4. Click **Save**.

Editing Sites

This procedure shows you how to edit an existing CM site in Analytics.

To edit a registered CM site

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

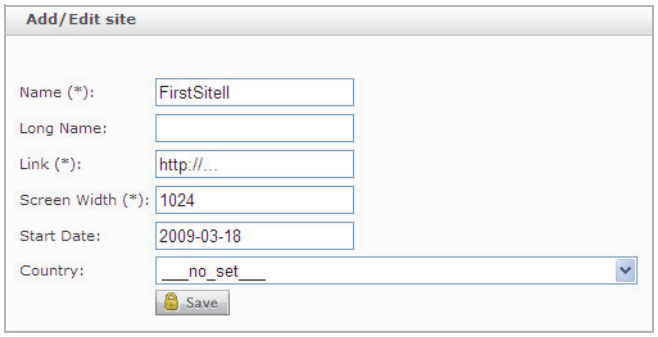
   `http://<host>:<port>/analyticsadmin/Admin`

2. In the "Sites Management" pane on the left, click **View/Edit**.

   Analytics displays a list of currently registered sites.

3. In the list of sites, click the desired site.

   Analytics displays an "Add/Edit Site" form populated with values for the selected site.



4. In the form, make your changes, then click **Save**.

Deleting Sites

This procedure shows you how to delete a CM site from Analytics.

To delete a CM site from Analytics

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) interface via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the "Sites Management" pane on the left, click **View/Edit**.

   Analytics displays a list of currently registered sites.

3. In the list of sites, click the **Delete** (trash can) icon next to the site you want to delete.
4. In the pop-up warning dialog that appears, click **OK**.

Managing Asset Reports

Before users can generate reports on Content Server assets, you must first configure the reports. Configuring an asset report enables Analytics to:

- Recognize Content Server asset types that you register for the report
- Configure report menu options in the "Content Information" report groups
- Generate reports on assets of the registered types

Requirements

After you install Analytics, you must configure the "Page Views" report. When you configure this report, you configure the `Pageview` object, a system default that specifies the type (or types) of assets Analytics will track. Configuring the `Pageview` object enables default reports that are based on the `Pageview` object.

The following reports can function only when the `Pageview` object is configured:

- Page Views
- Site Information
- Clickstream

For more information about the `Pageview` object, see the glossary at the end of this guide.

Note

Once you have configured the "Page Views" report, users will be able to generate the reports. The report, however, will be empty until you enable data capture for assets of the types that are specified in the `Pageview` object.

For instructions on configuring the "Page Views" report, follow the steps in "Configuring an Asset Report," on page 60.

Options

Internal Search Report

If you want Analytics to capture data on the keywords and phrases visitors enter into the site's internal search engine, configure an internal searches report. (This report will be listed in the "Content Information" group.) For instructions on configuring reports on internal searches, follow the steps in "Configuring an Asset Report," on page 60.

Custom Reports

You have the option to configure asset reports of your own. You will then need to complete the following steps (otherwise, the reports will remain empty):

1. Enable data capture for assets of the types you have selected for the report.
2. Develop an Analytics job to statistically process the captured data.
3. Create the report programmatically to display the required data in the desired layout.

For instructions on creating a custom asset report, follow the steps in "Configuring an Asset Report," on page 60.

Configuring an Asset Report

In this procedure, you will name the asset report, and specify the types of assets that Analytics will track for the report.

Note

- Reports that you configure will appear in the "Content Information" report group.
- Configure reports judiciously:
  - Deleting reports from the system (if necessary) is not trivial. A report deleted from the registration screen is not deleted as a report option. It remains as a menu item in the user's report generation interface and must be removed programmatically.
  - Deleting a report from the registration screen disables Analytics from analyzing data for that report. Previously analyzed report data is not deleted.

  If you need to delete a report from the Administration interface, follow the steps in "Deleting an Asset Report," on page 62.

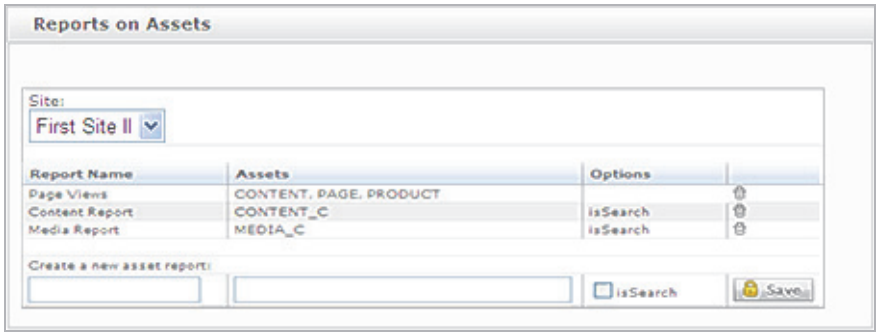
To configure a report (register one or more asset types)

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the "Reports on Assets" section of the "Asset Management" pane on the left, click **Configure**.

   Analytics displays the "Reports on Assets" form.



3. In the form, do the following:

   a. In the "Site" drop-down list, select the target site.

b. In the "Report Name" column, go to the "Create a new asset report" field, and enter a short, descriptive name for the report.

> **Note**
>
> - If you are configuring a "Page Views" report, enter **Page Views**. Analytics will not recognize the report under any other name.
> - If you are configuring an internal searches report, you must enter a name for your report and select the **isSearch** check box (in the "Options" column).

c. In the **Assets** field, enter the name(s) of the asset type(s) whose assets will be tracked by this report.

- If you are configuring the "Page Views" report, use the following convention:

  AssetType1, AssetType2,..., AssetTypeN

> **Note**
>
> - Enter asset type names exactly as they appear in the CS database.
> - Instead of commas, you can also use semicolons (;) to separate asset types in the list.

- If you are configuring an internal searches report, enter **Search** in the "Assets" field.

4. Click **Save**.

5. Next steps:

a. (Optional) Enable the performance indicator for assets of the types you selected for tracking by Analytics. For instructions, see "Enabling the Performance Indicator," on page 63.

b. Grant permissions to users, allowing them to generate the report you configured:

1) Register CM site users with Analytics. For instructions, see "Registering Users," on page 70.

2) Create user groups. For instructions, see "Creating Groups," on page 72.

3) Grant permissions to user groups. For instructions, see "Managing Permissions," on page 75.

4) Assign users to user groups. For instructions, see "Assigning Users to User Groups," on page 73.

c. An asset report remains empty until data capture is enabled for assets of the specified types. Data capture can be enabled now, or after the integration process is completed. Instructions for enabling data capture can be found in Chapter 6.

d. If you have created a custom report, develop an Analytics job to statistically process the captured data. Also, create the report programmatically to display the required information in the desired layout. Instructions are available in the *Analytics Developer's Tutorial*.

Deleting an Asset Report

This procedure shows you how to delete an asset report from the registration screen (in the Analytics Administration interface).

Note

- Deleting a report disables Analytics from analyzing data for that report. Previously analyzed report data is not deleted.
- A report deleted from the registration screen (in the Analytics Administration interface) is not deleted as a report option. It remains as a menu item in the user's report generation interface and must be removed programmatically.

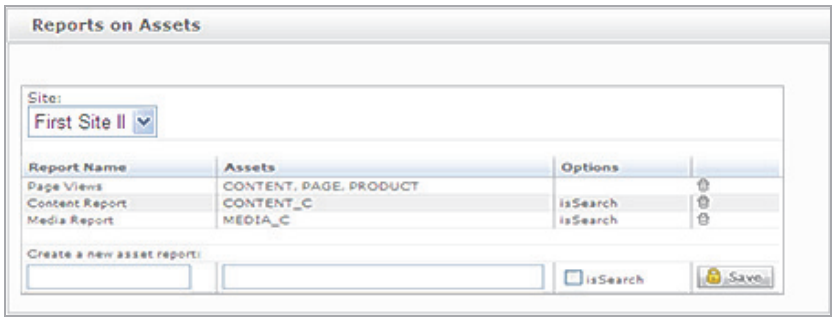
To delete an asset report

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the "Reports on Assets" section of the "Asset Management" pane on the left, click **Configure**.

   Analytics displays the "Reports on Assets" form.



3. In the form, do the following:
   a. In the "Site" drop-down list, select the target site.
   b. Click the **Delete** (trash can) icon next to the report you want to delete.

      The report you selected is removed from the "Reports on Assets" list.

Managing the Performance Indicator

This section shows you how to enable and disable the performance indicator for assets on your CS system.

The performance indicator appears in the "Inspect" form of every asset being tracked by Analytics. For example:

The indicator displays usage statistics for the asset on the chosen site.

This section contains the following procedures:

- Enabling the Performance Indicator
- Disabling the Performance Indicator

Enabling the Performance Indicator

This procedure shows you how to enable the performance indicator for one or more asset types on your CS system.

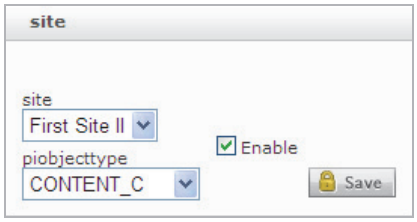
To enable the performance indicator

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the "Performance Indicator" section of the "Asset Management" pane on the left, click **Configure**.

   Analytics displays the "Site" form.



3. In the form, do the following:
   a. In the "Site" drop-down list, select the target site.
   b. In the "piobjecttype" drop-down list, select the desired asset type.
   c. Select the **Enable** check box.
4. Click **Save**.

Disabling the Performance Indicator

This procedure shows you how to disable the performance indicator for an asset type on your CS system.

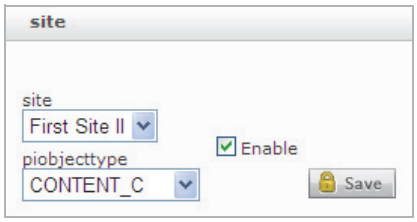
To disable the performance indicator

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the "Performance Indicator" section of the "Asset Management" pane on the left, click **Configure**.

Analytics displays the “Site” form.



3. In the form, do the following:
   a. In the “Site” drop-down list, select the target site.
   b. In the “piobjecttype” drop-down list, select the desired asset type.
   c. Deselect the **Enable** check box.
4. Click **Save**.

Managing Engage Asset Reports

Note

Skip this section if Engage is not installed on your Content Server system.

This section shows you how to configure Analytics to support Engage assets (recommendations and segments). This allows Analytics to:

- Track usage statistics for recommendations and the assets they reference
- Track segment membership for each visitor

This section covers the following topics:

Managing Recommendations

This section shows you how to manage recommendations in Analytics. It contains the following procedures:

Registering Recommendations

This procedure shows you how to register recommendations with Analytics. You must register each recommendation you want Analytics to track.

Note

If you do not have FatWire Engage installed on your CS system, stop here. You cannot register recommendations.

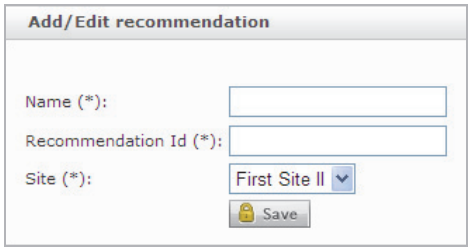
To register a recommendation with Analytics

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the "Recommendations" section of the "Engage" pane on the left, click **Register**.

   Analytics displays an empty "Add/Edit Recommendation" form.



3. Fill in the form as follows:

   Note

   An asterisk (*) next to a field name indicates that the field is required. You cannot leave required fields blank.

   a. In the **Name** field, enter the name of the recommendation asset. The name can, but is not required to, be the same as the name in the CS database.

   b. In the **Recommendation ID** field, enter the asset ID of the recommendation asset, exactly as it appears in the CS database.

   c. In the "Site" drop-down list, select the CM site for which you are registering the recommendation.

4. Click **Save**.

Editing Recommendations

This procedure shows you how to edit recommendations registered with Analytics.

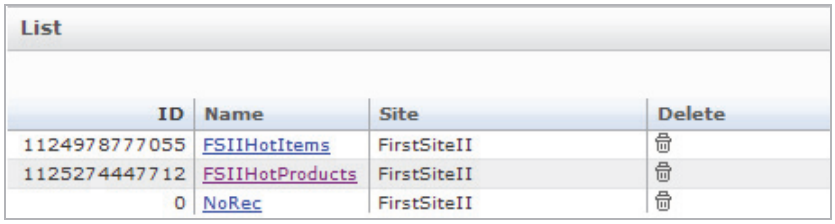
To edit a recommendation registered with Analytics

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

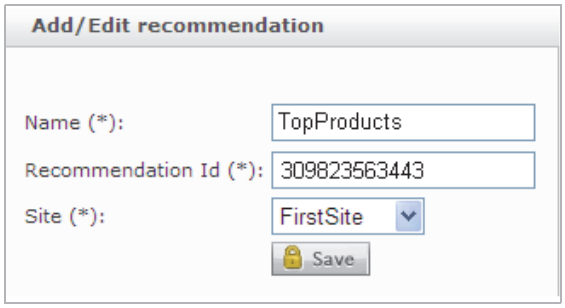
2. In the "Recommendations" section of the "Engage" pane on the left, click **View/Edit**.

   Analytics displays a list of currently registered recommendations.



3. In the list, click the desired recommendation.

   Analytics displays an "Add/Edit Recommendation" form populated with values for the selected recommendation.



4. In the form, make your changes, then click **Save**.

> **Note**
>
> An asterisk (*) next to a field name indicates that the field is required. You cannot leave required fields blank.

Deleting Recommendations

This procedure shows you how to delete recommendations from Analytics.

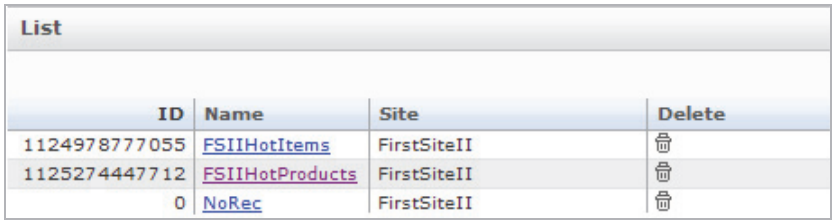
To delete a recommendation from Analytics

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the "Recommendations" section of the "Engage" pane on the left, click **View/Edit**.

   Analytics displays a list of currently registered recommendations.



3. In the list, click the **Delete** (trash can) icon next to the desired recommendation.
4. In the pop-up warning dialog that appears, click **OK**.

Managing Segments

This section shows you how to manage segments in Analytics. It contains the following procedures:

- Registering Segments
- Editing Segments
- Deleting Segments

Note

Before you register segments from your site, you must configure a segment named "No segment" with segment ID of 0 (zero), so that visitors that do not belong to any segments can be tracked. You must do this even if all visitors on your site have been assigned to segments.

Registering Segments

This procedure shows you how to register segments with Analytics. You must register each segment for which you want to track visitor membership.

Note

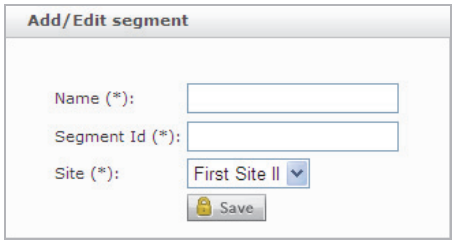
If you do not have FatWire Engage installed on your CS system, stop here. You cannot register segments.

To register a segment with Analytics

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) interface via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`
2. In the "Segments" section of the "Engage" pane on the left, click **Register**.

   Analytics displays an empty "Add/Edit Segment" form.



3. Fill in the form as follows:

> **Note**
>
> An asterisk (*) next to a field name indicates that the field is required. You cannot leave a required field blank.

   a. In the **Name** field, enter the name of the segment asset. The name can, but is not required to, be the same as in the CS database.
   b. In the **Segment ID** field, enter the asset ID of the segment asset.
   c. In the "Site" drop-down list, select the site for which you are registering the segment.

4. Click **Save**.

Editing Segments

This procedure shows you how to edit segments registered with Analytics.

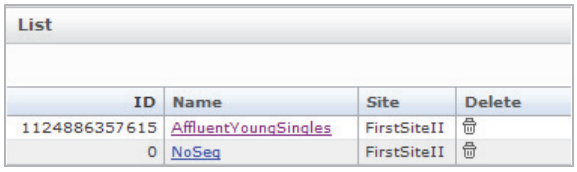
To edit a segment registered with Analytics

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

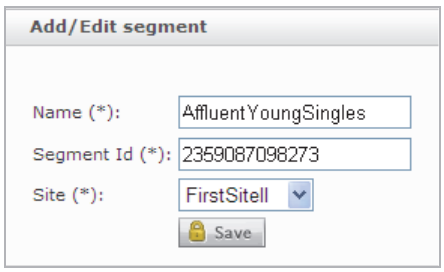
2. In the "Segments" section of the "Engage" pane on the left, click **View/Edit**.

   Analytics displays a list of currently registered segments.



3. In the list, click the desired segment.

   Analytics displays an "Add/Edit segment" form populated with values for the selected segment.



4. In the form, make your changes, then click **Save**.

> **Note**
>
> An asterisk (*) next to a field name indicates that the field is required. You cannot leave a required field blank.

Deleting Segments

This procedure shows you how to delete segments registered with Analytics.

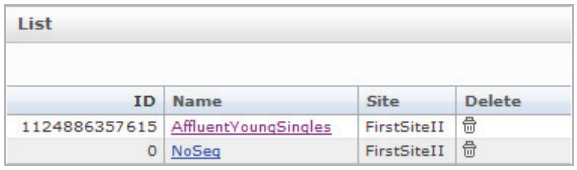
To delete a segment from Analytics

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the "Segments" section of the "Engage" pane on the left, click **View/Edit**.

   Analytics displays a list of currently registered segments.



3. In the list, click the **Delete** (trash can) icon next to the desired segment.
4. In the pop-up warning dialog that appears, click **OK**.

Managing Users and Groups

In FatWire Analytics, access to reports and Analytics interfaces (User and Admin) is granted to CS users via membership in user groups, to which you assign the desired permissions. When you integrate Analytics with your CS system, you will register the CS users individually (in the Analytics Admin interface), and then add them to a user group.

This section includes the following topics:

Managing Users

This section shows you how to manage CS users in Analytics. It contains the following procedures:

Registering Users

This procedure shows you how to register CS users with Analytics.

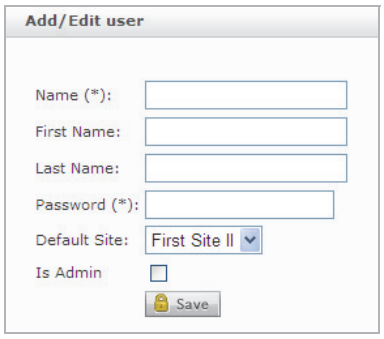
To register a CS user with Analytics

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the "User" section of the "User Management" pane on the left, click **Register**.

   Analytics displays an empty "Add/Edit User" form.



3. Fill in the form as follows:

   a. **Name (required)** – enter the user name of the user as it appears in the CS interface.

   b. **First name** – enter the user's first name.

   c. **Last name** – enter the user's last name.

   d. **Password** – enter the password you want to assign to the user.

e. **Default Site** – select the which the user will see when he/she logs in to the Analytics interface.

f. **Is Admin** – grants the user access to the Analytics administration interface.

4. Click **Save**.

Editing Users

This procedure shows you how to edit CS users that have been registered with Analytics.

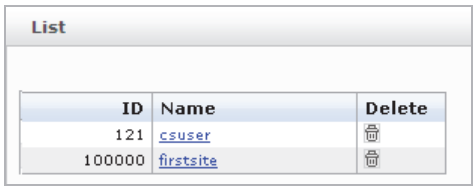
To edit a registered CS user

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

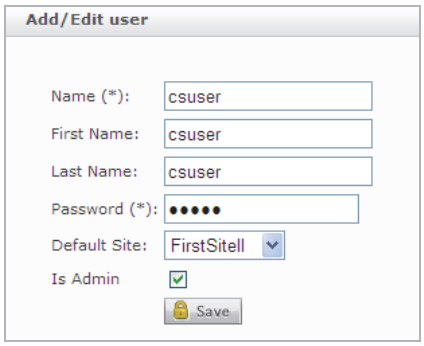
2. In the "User" section of the "User Management" pane on the left, click **View/Edit**.

   Analytics displays a list of currently registered users.



3. In the list, click the desired user.

   Analytics displays an "Add/Edit User" form populated with values for the selected user.



4. In the form, make your changes, then click **Save**.

Deleting Users

This procedure shows you how to delete CS users from Analytics.

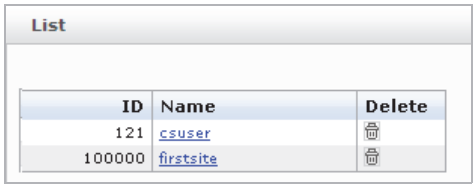
To delete a CS user from Analytics

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the "User" section of the "User Management" pane on the left, click **View/Edit**.

Analytics displays a list of currently registered users.



3. In the list, click the **Delete** (trash can) icon next to the desired user.
4. In the pop-up warning dialog that appears, click **OK**.

Managing User Groups

The purpose of user groups is to grant permissions to group members. This section shows you how to manage user groups in Analytics. It contains the following sections:

- Creating Groups
- Editing Groups
- Deleting Groups
- Assigning Users to User Groups

Creating Groups

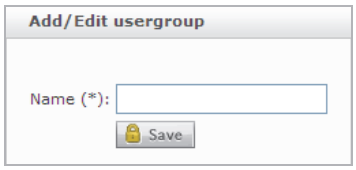
This procedure shows you how to add a user group in Analytics.

To add a user group

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`
2. In the "Usergroups" section of the "User Management" pane on the left, click **Register**.

   Analytics displays an empty "Add/Edit Usergroup" form.



3. In the form, enter a name for the group and click **Save**.

Editing Groups

This procedure shows you how to edit a user group in Analytics.

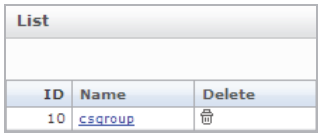
To edit a user group

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

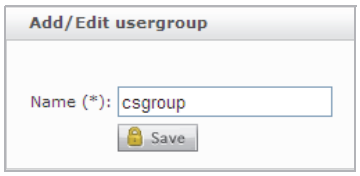
2. In the “Usergroups” section of the “User Management” pane on the left, click **View/Edit**.

   Analytics displays a list of existing user groups.



3. In the list, click the desired user group.

   Analytics displays the name of the user group you wish to edit in the “Add/Edit Usergroup” form.



4. In the form, make your changes and click **Save**.

Deleting Groups

This procedure shows you how to delete a user group in Analytics.

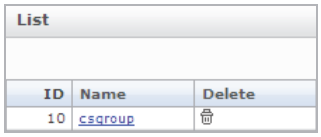
To delete a user group

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the “Usergroups” section of the “User Management” pane on the left, click **View/Edit**.

   Analytics displays a list of existing user groups.



3. In the list, click the **Delete** (trash can) next to the desired user group.
4. In the pop-up warning dialog that appears, click **OK**.

Assigning Users to User Groups

This procedure shows you how to assign users to user groups in Analytics. Analytics users can only access the reports for which their group has the appropriate permissions.

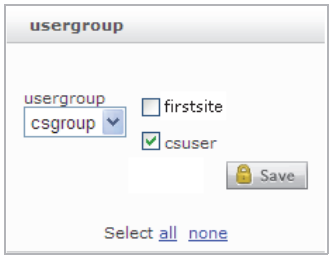
To assign users to a user group

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the “User to Usergroup” section of the “User Management” pane, click **Register**.

Analytics displays the “Usergroup” form.



3. In the form, do the following:
   a. In the “usergroup” drop-down list, select the target group.
   b. Select the users which you want to add to the user group. Deselect the users you want to remove from the group.
4. Click **Save**.

Managing Permissions

This procedure shows you how to grant and deny permissions to users in a user group. At the very least, each user must have access to the "Page Views" report in their default site (the site they see when they log in to the Analytics interface) or they will not be able to log in to the Analytics interface.

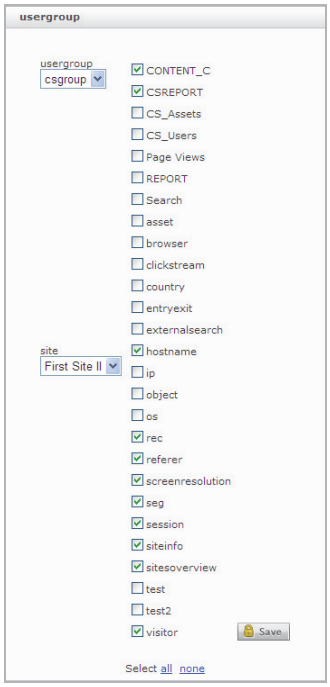
To grant or deny permissions to users in a user group

1. Log in to the Analytics administration interface as the `csuser` user (password: `csuser`) via the following URL:

   `http://<host>:<port>/analyticsadmin/Admin`

2. In the "Permissions" pane on the left, click **Grant Rights**.

   Analytics displays the "Grant Rights" form.



3. In the form, do the following:
   a. In the "Usergroup" drop-down list, select the desired group.
   b. In the "Site" drop-down list, select the target site.
   c. Select the permissions you want to grant to the users in the group. Deselect the permissions you want to deny the users in the group.
4. Click **Save**.

Next Step

Your next step is to configure data capture for all FatWire Analytics reports. Your final step will be to test your installation. For instructions on both steps, see Chapter 6, "Configuring Data Capture."

Chapter 6

Configuring Data Capture

In this chapter, you will enable data capture for published pages by implementing the `AddAnalyticsImgTag` and the associated code. Captured data can then be processed and made available to all FatWire Analytics reports.

This chapter contains the following sections:

Enabling Data Capture

Note

The steps in this section require you to have knowledge of Content Server elements, Java, and JSP. If you plan to enable data capture on Engage assets, you must also have a thorough understanding of recommendations and segments.

Before starting the steps in this chapter, make sure you have completed the procedures in Chapter 5, "Integrating Analytics with Content Server."

FatWire Analytics can capture data on the usage of Content Server assets and on their visitors, and the visitor's searches only if published pages are tagged for data capture. In the case of Engage assets and internal searches, the assets themselves must be tagged for data capture.

This section shows you how to:

A. Enable Data Capture for General, Content, and User Information Reports

B. Enable Data Capture for the "Internal Searches" Report

C. Enable Data Capture for Engage Reports

Completing the steps enables data capture for all reports in all report groups.

A. Enable Data Capture for General, Content, and User Information Reports

In this section, you will enable data capture for all report groups except "Internal Searches" and "Engage."

To enable data capture

1. Configure data capture on CS assets and their visitors by tagging the pages on which the assets are displayed. Use the following code:

   Data Capture Code:

```
<%-- Analytics Image tag
     Data capture code for asset uses and user information
--%>
<ics:if
   condition='<%="true".equalsIgnoreCase(ics.GetProperty("analy
   tics.enabled", "futuretense_xcel.ini", true))%>'>
<ics:then>
   <ics:if condition='<%=ics.GetVar("packedargs") != null%>'>
   <ics:then>
      <render:unpackarg unpack="recid" remove="true"
         packed='<%=ics.GetVar("packedargs")%>'
         outvar="packedargs"/>
   </ics:then>
   </ics:if>
<render:callelement elementname="Analytics/AddAnalyticsImgTag">
<render:argument name="c" value='<%=ics.GetVar("c")%>'/>
<render:argument name="cid" value='<%=ics.GetVar("cid")%>'/>
```

```
<render:argument name="site" value='<%=ics.GetVar("site")%>'/>
<render:argument name="pagename"
   value='<%=ics.GetVar("childpagename")%>'/>
<render:argument name="recid"
   value='<%=ics.GetVar("recid")%>'/>
</render:callelement>
</ics:then>
</ics:if>
```

Parameters

`c`: Type of asset being visited

`cid`: ID of an asset of a given type

`site`: Name of the content management site hosting the page

`pagename`: Name of the page being visited

`pageurl`: URL of the page being visited

Note

- Be sure to tag pages that display the assets whose asset types you have registered. If the pages remain untagged, empty reports will be generated. (For registration procedures, see section "Configuring an Asset Report," on page 60.)
- If you are using the FirstSite II sample site, note that it uses a wrapper page that is called by all other pages. Add the data capture code (see page 78) to the wrapper element (for example, `ElementCatalog/FSIIWrapper`). Do the same for any other site that uses a wrapper page. For any site that does not use a wrapper page, make sure that the `c`, `cid`, `site` and `pagename` parameters are properly passed from the page to the data capture code.
- To enable generation of a site's "Entry - Exit Pages" report (which identifies a session's first and last pages), you can code either all pages on that site, or a wrapper page that is called by all other pages.

For each page request, the `FatWire/Analytics/AddAnalyticsImgTag` invokes the `sensor` servlet in FatWire Analytics to capture data on visitors browsing the page. The data includes visitors' geographic locations, IP addresses, browsing technologies, and session parameters. The remaining code captures usage information on Content Server assets displayed on the same page. The captured data is later processed statistically. The statistics are then available for "General Information" reports, "Content Information" reports, and "User Information" reports.

B. Enable Data Capture for the “Internal Searches” Report

Enable data capture for “Internal Searches” report by calling the HTML `img` tag on every page where search data must be captured. Pass the following parameters:

```
<img
 style="display:none;"
 height="1"
 width="1"
 border="0"
 src="<%=statisticsUrl%>
 ?siteName=<%= ics.GetVar("site")%>
 &objType=SEARCH
 &query=<SQL_query>
 &objID=1
 &directhits=<number_of_results>
 &maxscore=<relevance_score>
 &sessionID=<%=sessionId%>"
 alt="pixel"
/>
```

Parameters

`src`
URL of the data capture servlet for FatWire Analytics. The URL is configured in the `analytics.datacaptureurl` property, in `futuretense.ini`. Get the URL as follows:

```
<property:get param="analytics.datacaptureurl"
  inifile="futuretense.ini" varname="datacaptureurl"/>
<% String statisticsUrl=
    ics.GetVar("datacaptureurl");%>
```

`siteName`
Name of the content management site where the query is entered.

`objType`
Object name for the internal SQL query. The value must be `SEARCH`.

`query`
The query that is entered in the search field.

`objID`
A random value (pass a value of “1”).

`directhits`
Number of results returned by the search query.

`maxscore`
(Optional) the relevance score of the most relevant result.
For example, `88.0` for a result that is 88% relevant to the keywords entered.

`sessionID`
Browser session ID.

Example:

The following example shows how the `img` tag could look after the variables are replaced:

```
<img
 style="display:none;"
 height="1"
 width="1"
 border="0"
 src="http://10.120.12.68:7001/analytics/statistics
 ?siteName=FirstSiteII
 &objType=SEARCH
 &query=Select * from Activities
 &objID=1
 &directhits=10
 &maxscore=88.0
 &sessionID=5E7B9B22F84F0FAB0E68108089AA1609"
 alt="pixel"
/>
```

If your pages contain Engage assets and you wish to enable data capture, continue with the next section "Enable Data Capture for Engage Reports." Otherwise, you have completed configuring FatWire Analytics. Test the system by following the steps in section "Test FatWire Analytics," on page 88.

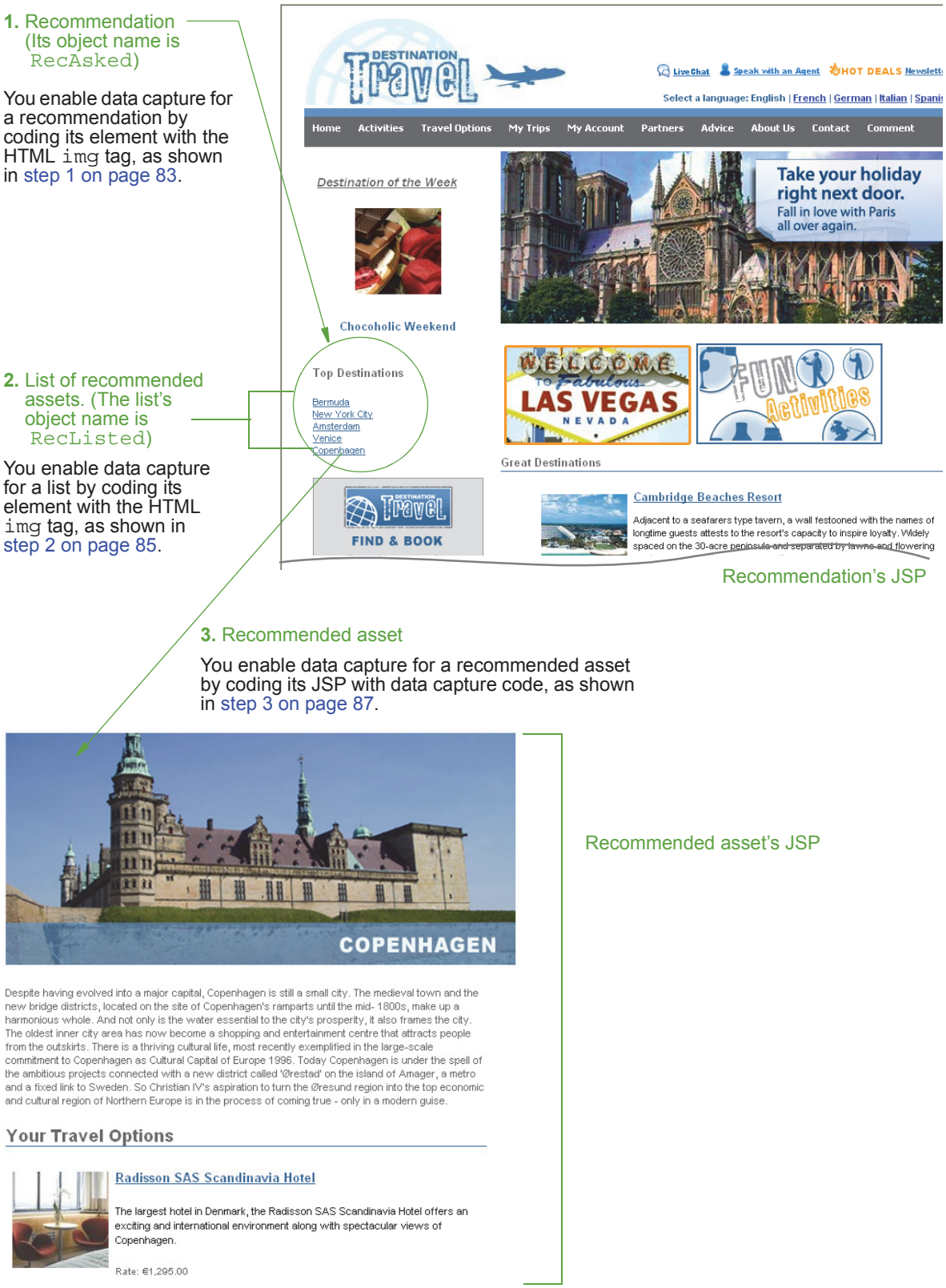
C. Enable Data Capture for Engage Reports

Engage assets include recommendations, recommended assets (advertised in the recommendations), and segments to which the recommendations are made. Before starting the steps in this section, familiarize yourself with their general flow by referring to the steps in Figure 6, on page 82. The same figure provides examples of Engage assets and their counterpart objects `RecAsked` and `RecListed`.

Note

Data capture on segments is implicit via the `segID` parameter in the `img` tag, used to enable data capture.

Figure 6: Definitions of Engage Assets



To enable data capture on Engage assets

Note

To complete the steps below, you must have a thorough understanding of recommendations and segments, as well as experience with Java and JSP.

1. Enable data capture on each recommendation (for an example of a recommendation, see Figure 6, on page 82):
   a. Add the lines below to the recommendation's element:

```
<%@ taglib prefix="property" uri="futuretense_cs/
   property.tld"%>
<%@ page import="java.util.*, java.text.*, java.io.*"%>
<%@ page import="at.onetoone.esa.tools.*"%>
```

Note

The `at.onetoone.esa.tools` files are located in `analyticscs.jar`.

   b. Insert the HTML `img` tag below into the recommendation's element. The `img` tag will capture the ID and name of the recommendation as soon as the page is displayed to the visitor.

Note

If you are using the FirstSite II sample site, add the `img` tag to `ElementCatalog/AdvCols/FSIIDetail`.

```
<script type="text/javascript">
      pixelreccalled = new Image();
      pixelreccalled.src = "<%=statisticsUrl%>?siteName
                          =<%= ics.GetVar("site")%>

&objType=RecAsked&objID=<%=ics.GetVar("cid")%>

&objName=<%=ics.GetVar("reconame")%>
           &segID=<%=segmentidlist%>
           &sessionID=<%=sessionId%>
           &Referer=<%= referer %>
           &size=<%=screenResolution%>";
</script>
<noscript>
<img style="display:none;"
height="1"
width="1"
src="<%=statisticsUrl%>?siteName=<%= ics.GetVar("site")%>
&objType=RecAsked
&objID=<%=ics.GetVar("cid")%>
&objName=<%=ics.GetVar("reconame")%>
&segID=<%=segmentidlist%>
```

```
&sessionID=<%=sessionId%>
&Referer=<%= referer %>
&size=<%=screenResolution%>
alt="pixel" />
</noscript>
```

Parameters

`src`
URL of the data capture servlet for FatWire Analytics. The URL is configured in the `analytics:datacaptureurl` property, in `futuretense.ini`. Get the URL as follows:

```
<property:get param="analytics.datacaptureurl"
  inifile="futuretense.ini" varname="datacaptureurl"/>

<% String statisticsUrl=
    ics.GetVar("datacaptureurl");%>
```

`siteName`
Name of the content management site where the recommendation is displayed.

`objType`
Object name for the displayed recommendation. The value must be `RecAsked`. (For an example of a recommendation, see step 1 in Figure 6, on page 82.) The `objType` is used to identify Engage assets (recommendations and segments), capture data on the assets, and create reports in the "Engage" report group.

`objID`
Asset ID of the recommendation.

`objName`
Name of the recommendation.

`segID`
Either a comma-separated list of IDs of the segments to which the visitor belongs, or `0` if the visitor does not belong to a segment.

`sessionID`
Browser session ID.

`size`
Browser screen resolution.

Example

The following example shows how the `img` tag could look after the variables are replaced:

```
<img
   style="display:none;"
   height="1"
   width="1"
   src="http://etravel.fatwire.com:80/analytics/statistics
   ?siteName =FirstSiteII
```

```
      &objType=RecAsked&objID=1120593564193
      &objName=FSIIItemOfTheWeek
      &segID=0
      &sessionID=B28822803C3843B8CC
      &size=1280x1024"
      alt="pixel"
   />
```

2. Enable data capture on each recommendation's list of recommended assets (for an example of such a list, see Figure 6, on page 82).

   This step requires you to first encode three parameters, then insert the HTML `img` tag into the same recommendation element as in the previous step, but pass parameters which are specific to the list of recommended assets. Do the following:

   a. Encode the parameters `objListName`, `objListID`, and `objListType` as follows, using the `Base64` file located in `analyticscs.jar`:

   ```
   String objListNameValue = Base64.encodeBytes(<comma-
    separated list of asset names in the
    recommendation>.getBytes(), Base64.DONT_BREAK_LINES);

   String objListIDValue = Base64.encodeBytes(<comma-separated
    list of asset IDs in the recommendation>.getBytes(),
    Base64.DONT_BREAK_LINES);

   String objListTypeValue = Base64.encodeBytes(<comma-
    separated list of asset types in the
    recommendation>.getBytes(), Base64.DONT_BREAK_LINES);
   ```

   b. Insert the HTML `img` tag with the following parameters into the JSP:

   Note

   If you are using the FirstSite II sample site, add the `img` tag to `ElementCatalog\AdvCols\FSIIDetail`.

   For example code, see `FSIIDetail.jsp` (in `FatWire Analytics\examples\FirstSiteII\ElementCatalog\AdvCols`).

```
<script type="text/javascript">
      imagereclisted = new Image();
      imagereclisted.src = "<%=statisticsUrl%>?siteName=<%
                         = ics.GetVar("site")%>
                         &objType=RecListed
                         &objID=<%=ics.GetVar("cid")%>
                         &objName=<%=ics.GetVar("reconame")%>
                         &objListID=<%=objListIDValue%>
                         &objListName=<%=objListNameValue%>
                         &objListType=<%=objListTypeValue%>
                         &segID=<%=segmentidlist%>
                         &sessionID=<%=sessionId%>
                         &size=<%=screenResolution%>";
</script>
```

```
<noscript>

<img style="display:none;"
height="1"
width="1"
src="<%=statisticsUrl%>
?siteName=<%= ics.GetVar("site")%>
&objType=RecListed
&objID=<%=ics.GetVar("cid")%>
&objName=<%=ics.GetVar("reconame")%>
&objListID=<%=objListIDValue%>
&objListName=<%=objListNameValue%>
&objListType=<%=objListTypeValue%>
&segID=<%=segmentidlist%>
&sessionID=<%=sessionId%>
&size=<%=screenResolution%>"
alt="pixel"
/>

</noscript>
```

Parameters

`src`
URL of the data capture servlet for FatWire Analytics. The URL is configured in the `analytics.datacaptureurl` property, in `futuretense.ini`. Get the URL as follows:

```
<property:get param="analytics.datacaptureurl"
 inifile="futuretense.ini" varname="datacaptureurl"/>

<% String statisticsUrl=
    ics.GetVar("datacaptureurl");%>
```

`siteName`
Name of the content management site where the recommendation is listed.

`objType`
Object name for the list of recommended assets. The value must be `RecListed`. (For an example of a list of recommended assets, see step 2 in Figure 6, on page 82.) The `objType` is used to identify Engage assets (recommendations and segments), capture data on the assets, and create reports in the "Engage" report group.

`objID`
Asset ID of the recommendation.

`objName`
Name of the recommendation.

`objListName`
Comma-separated list of asset names in the recommendation. The value for this parameter is the value that was encoded in the previous step.

`objListType`
Comma-separated list of asset types in the recommendation. The value for this parameter is the value that was encoded in the previous step.

`segID`
Either a comma-separated list of IDs of the segments to which the visitor belongs, or `0` if the visitor does not belong to a segment.

`sessionID`
Browser session ID.

`size`
The browser screen resolution.

Example:

The following example shows how the `img` tag could look after the variables are replaced:

```
<img
   style="display:none;"
   height="1"
   width="1"
   src="http://etravel.fatwire.com:80/analytics/statistics
   ?siteName=FirstSiteII
   &objType=RecListed
   &objID=1120593564193
   &objName=FSIIItemOfTheWeek
   &objListID=MTEzODY2MDI1NjExMQ===
   &objListName=Q2hvY29ob2xpYyBXZWVrZW5k=
   &objListType=UHJvZHVjdF9D=
   &segID=0&sessionID=B28822803C3843B8CC496743D5B132B0
   &size=1280x1024"
   alt="pixel"
/>
```

c. Add the `recid` parameter to the element that generates the links which make up the list of recommended assets (each link points from the recommendation to a recommended asset). Also, add the `recid` parameter to the `pagecriteria` variable for the element's `SiteCatalog` entry.

Note

`recid` is the ID of the recommendation to which an asset belongs. For example code, see `FSIISummary.jsp` for `Product_C` (in `FatWire Analytics\examples\FirstSiteII\ElementCatalog\Product_C`).

3. Enable data capture on each recommended asset (for an example of a recommended asset, see Figure 6, on page 82):

a. If the data capture code from page 78 does not exist in the recommended asset's JSP, add the code.

Note

Data capture code can exist on a page displaying an Engage asset if that page was already coded for data capture on Content Server assets.

b. Pass the `recid` parameter to the data capture element `"FatWire/Analytics/AddAnalyticsImgTag"` by inserting the following line into the data capture code on page 78:

```
<render:argument name="recid" value='<%=ics.GetVar
  ("recid")%>'/
```

This enables FatWire Analytics to record which Engage asset has been clicked.

4. At this point, you have enabled data capture for each recommendation, its list of recommended assets, and its individual recommended assets. You have also implicitly enabled data capture for segments, via the `segID` parameter in the `img` tags.

To complete the FatWire Analytics installation, continue to the next step, to test your installation.

Test FatWire Analytics

You can test FatWire Analytics either on your own or with the help of the Content Server administrator and other users.

To test FatWire Analytics

1. Go to the web site and visit the pages you have tagged for data capture.

2. Allow the data analysis jobs to finish processing the captured data, then generate all the reports that users will be generating. The jobs process data captured within the past 24 hours. Bear in mind that the data analysis job is a resource-intensive process that can take a significant amount of time to complete, depending on the amount of raw data that has been captured.

For descriptions of the reports and instructions on generating them, see the *FatWire Analytics User's Guide*.

Assuming a successful outcome, you are now ready to use FatWire Analytics.

Appendix A

Hadoop Jobs: Processors and Locations

Hadoop Jobs consists of locations that store site visitor data and processors that selectively handle the portion of the data they are programmed to handle. A given processor reads one location, processes the data, and writes the results to the next location for pickup by the next processor.

This appendix contains a summary of Hadoop Jobs, guidelines for monitoring Hadoop Jobs, and descriptions of the processors and locations. This appendix contains the following sections:

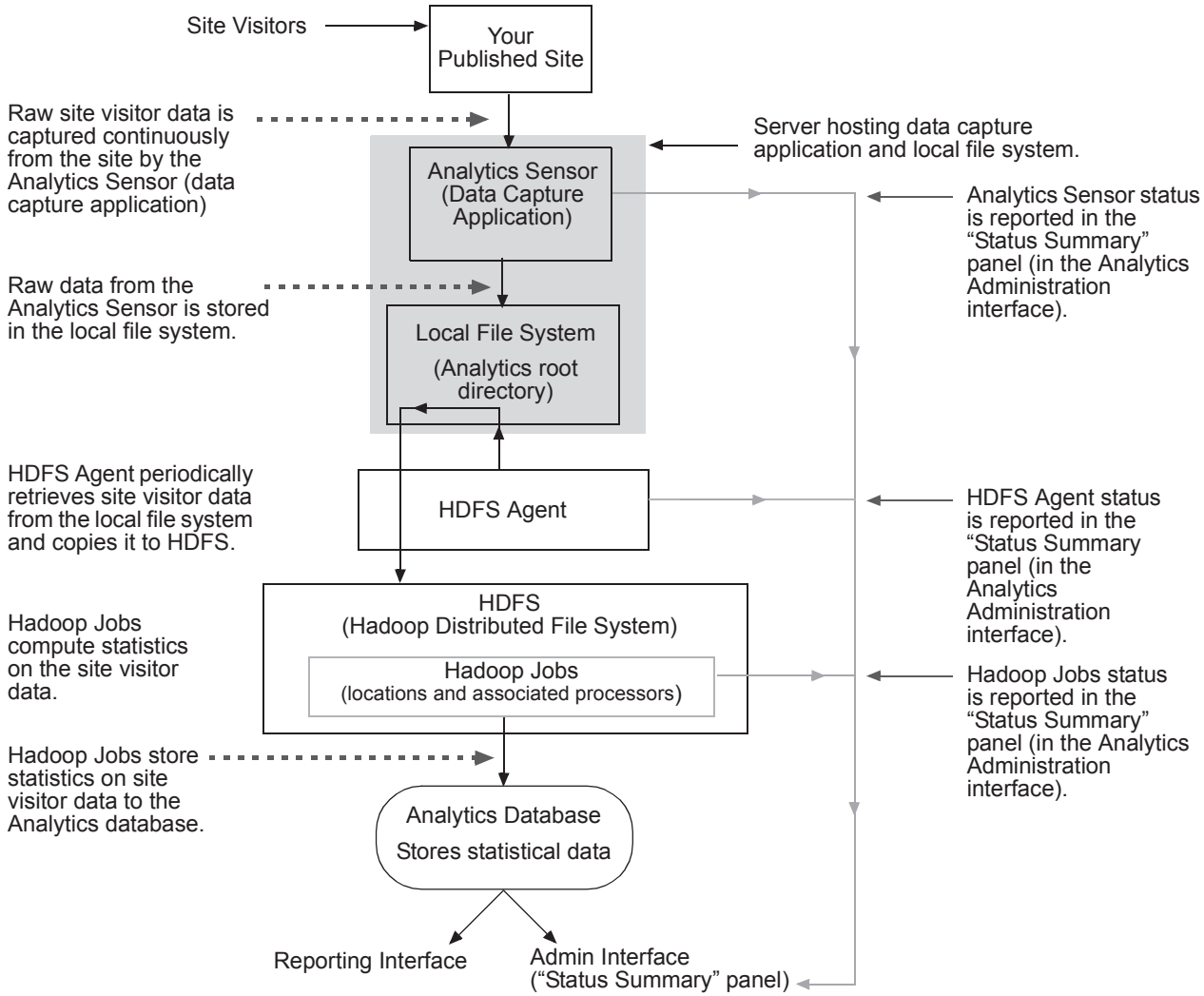
Hadoop Jobs Process Flow

Hadoop Jobs is a FatWire application that statistically processes Analytics data and stores the results in the Analytics database.

In a functional Analytics installation, raw site visitor data is continuously captured by the Analytics Sensor (Data Capture Application), which then stores the data into the local file system. The raw data in the file system is called on periodically by the HDFS Agent, which then copies the raw data to the Hadoop Distributed File System (HDFS), where Hadoop jobs process the data.

Hadoop jobs consist of locations and FatWire-specific processors that read site visitor data in one location, statistically process that data, and write the results to another location for pickup by the next processor. When processing is complete, the results (statistics on the raw data) are injected into the Analytics database.

Hadoop jobs can be monitored from the "Status Summary" panel of the Analytics Administration interface (see "Monitoring Hadoop Jobs," on page 91).



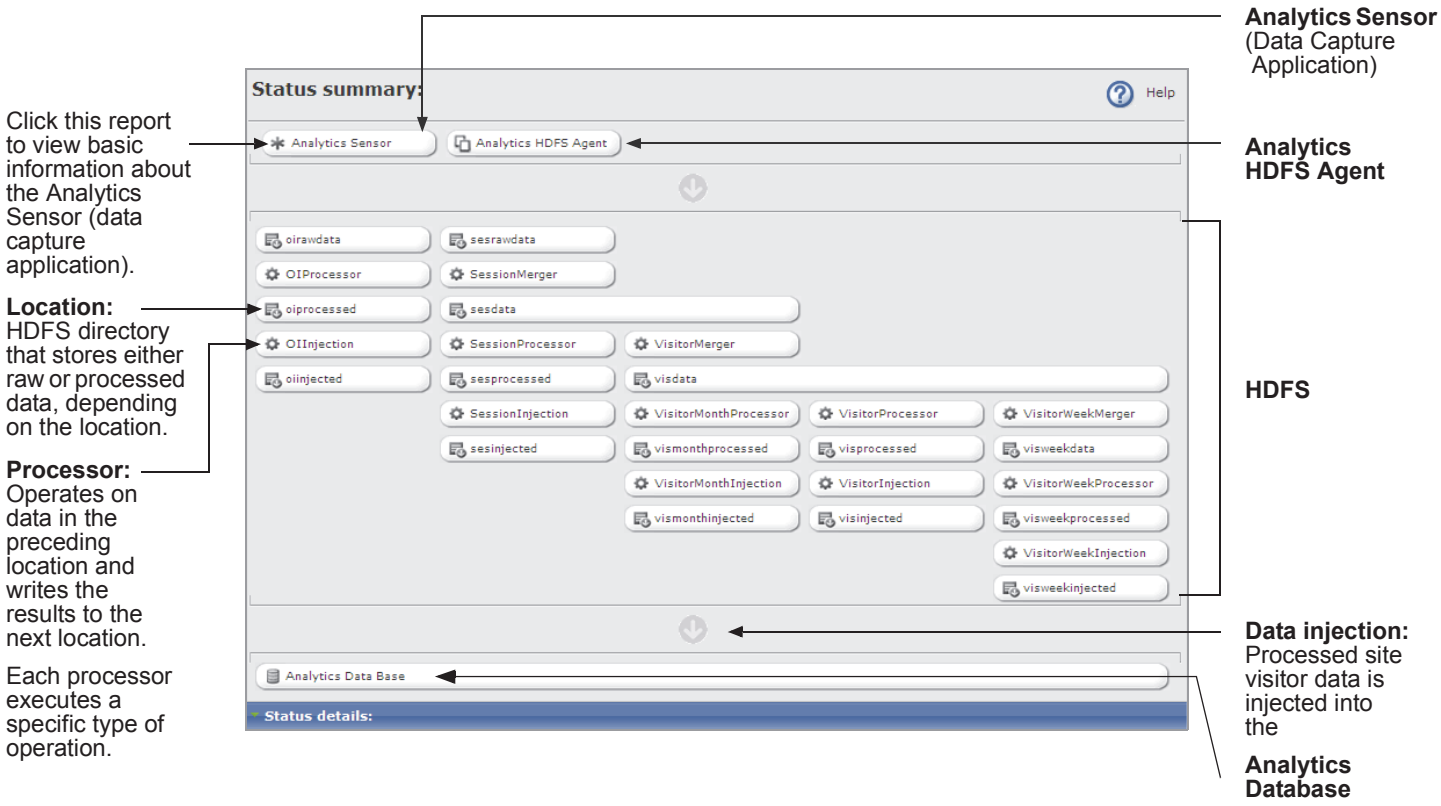
Monitoring Hadoop Jobs

Hadoop jobs can be monitored from the "Status Summary" panel of the Analytics Administration interface. The "Status Summary" panel renders an interactive flow chart which displays Hadoop job components—locations that store site visitor data captured by the Analytics Sensor (data capture application) and processors that calculate daily/weekly/monthly sums for the stored site visitor data.

To access the "Status Summary" panel

When working in the Analytics Administration interface, you can access the "Status Summary" panel by clicking the **Components** tab and selecting the **Overview** option.

Figure A-1: "Status Summary" Panel (**Components > Overview**)



Each location stores different types of site visitor data. The type of site visitor data that is stored in a given location is determined by how that data is aggregated by the location's associated processor. For example, the `oiprocessed` location is associated with the `OIProcessor` (it stores the results of the `OIProcessor`'s computation) and therefore stores data such as the number of times specific assets have been rendered during a given time interval on a given date.

- Clicking on a location enables you to view the status of the location and its data.
- Clicking on a processor enables you to view the status of the data processing job.

- Clicking on the **Analytics Sensor** and **HDFS Agent** buttons provides a status summary of those components. For more information about monitoring the Analytics Sensor, see "Sensor Overload Alerts," on page 50.

Processors and Locations

This section describes the different locations that are involved in storing site visitor data, and the processors that read the data from their locations, map/reduce the data, and write the results to another location.

- **Processors** – HDFS includes several processors developed by FatWire to process Analytics data. A processor consists of two parts: a mapper and a reducer. The mapper starts with a set of object impressions (collection of raw data) and creates intermediate data (*n* Java beans). The intermediate data is processed by the reducer in a way that aggregates the *n* Java beans into one Java bean containing *x* occurrences of a given data type, a second Java bean containing *y* occurrences of a different data type, and so on. As the reducer runs, it writes the aggregated data to the next location. The output of a processor is called a "work package" (for more information, see "Work Packages," on page 94).

  Each execution of a processor is called a **job**, and each job is a map/reduce job. As each job is scheduled, it is assigned a unique job-identifier.

- **Locations** – HDFS stores the site visitor data (both raw and processed) in different folders known as locations. A **location** is a specific folder in HDFS, which can be monitored through the "Status Summary" panel in the Administration interface (Figure A-1, on page 91).

  A location has sub-folders which represent year, month, day, and time, and these sub-folders are arranged in a hierarchical manner, as show in the sample below.

```
<hadoop.hdfs.defaultfs>
```

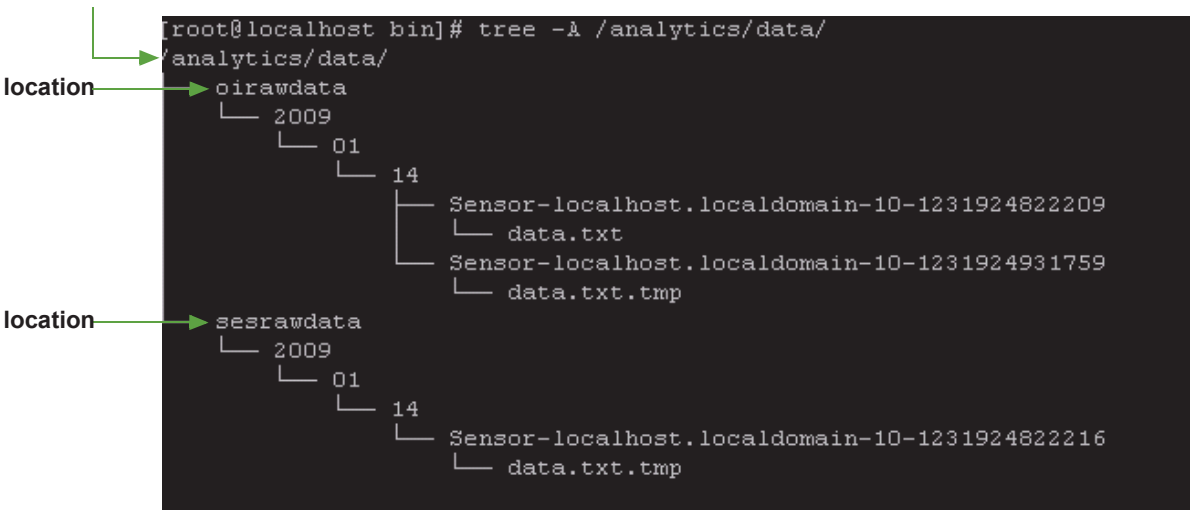


Table A-1, on page 93 lists the different locations and processors that can be monitored from the Analytics "Status Summary" panel. See also "Directory Structure for Raw Data and oiprocessed Data," on page 95," for example.

Each location stores site visitor data in one of the following formats, depending on the location:

- **Raw data** – Site visitor data in every object impression that is captured by the Analytics Sensor (data capture application).
- **Processed data** – Site visitor data that is processed from its raw form. Fully processed site visitor data is injected into the Analytics database for reporting purposes.

Note

Locations and processors are listed alphabetically in Table A-1. A processor and a location listed in the same row are not necessarily associated with each other.

Table A-1: Processors and locations

Processor	Location
OIInjection (page 102)	oiinjected (page 102)
OIProcessor (page 99)	oiprocessed (page 101)
SessionInjection (page 106)	oirawdata (page 100)
SessionMerger (page 103)	sesdata (page 104)
SessionProcessor (page 105)	sesinjected (page 106)
VisitorInjection (page 111)	sesprocessed (page 105)
VisitorMerger (page 107)	sesrawdata (page 103)
VisitorMonthInjection (page 109)	visdata: (page 107)
VisitorMonthProcessor (page 108)	visinjected (page 111)
VisitorProcessor (page 110)	vismonthinjected (page 109)
VisitorWeekInjection (page 114)	vismonthprocessed (page 108)
VisitorWeekMerger (page 112)	visprocessed (page 110)
VisitorWeekProcessor (page 113)	visweekdata (page 112)
	visweekinjected (page 114)
	visweekprocessed (page 113)

Object Impressions and Work Packages

Object impressions and work packages are the main constructs of Hadoop Jobs. Object impressions are raw site visitor data that is captured as visitors browse and then processed by Hadoop Jobs in units called "work packages." Results of the processing are stored to the Analytics database, where they are available on demand for the reports users generate.

Object Impressions

An **object impression** is a single invocation of the `sensor` servlet. An object impression can also be thought of as a "snapshot" of raw site visitor data that is captured for analysis.

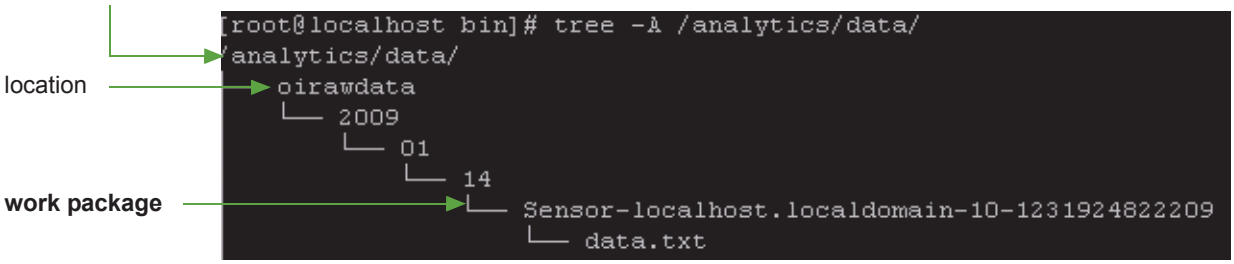
An object impression contains many types of raw data on the site visitor at the moment of capture. It contains session data and visitor data including object types, object IDs, sessions, session IDs, IP addresses, operating systems used, browsers used, referrers, and so on. If FatWire Engage is installed, the raw data also includes segments and recommendations.

When site visitors browse, object impressions are collected. They are collected during a 24-hour period as work packages in the `oirawdata` and `sesrawdata` locations.

Work Packages

A work package is a directory within a location, as shown below.

<hadoop.hdfs.defaultfs>



A work package stores:

- A data file containing either object impressions (raw data) or intermediate data (Java beans). The contents of the data file are statistically analyzed by a series of processors. When analysis is complete, the final processor injects results into the Analytics database for report generation.
- A metadata file. The metadata file reports the data processing status.

At least one work package exists in each location in the Hadoop Distributed File System (the number of work packages depends on the location. For an example, see "Data Collection"). Each work package is positioned hierarchically in the location's directory structure, according to a calendar type of structure.

The data file in a given work package is input for the processor that is associated with the location containing the work package. When the processor completes its analysis of the

data file, it writes the results, as a work package, to the next location for pickup by the next processor.

Note

The initial work package, containing newly captured object impressions, is created by the Analytics Sensor. All other work packages are created by the processors.

During data processing, neither work packages nor their contents are moved from one location to another. Instead, each work package's data file is read by the appropriate processor and analyzed by the processor. Results are written (by the processor) as a work package to the next location.

Data Collection

Object impressions are collected as work packages for a 24-hour period into two locations, simultaneously—`oirawdata` and `sesrawdata`. All work packages in the two locations contain a data file named `data.txt`. The locations (and their work packages) differ as follows:

- The `oirawdata` location collects objects impressions at fixed intervals during a 24-hour period; each interval has its own work package. The interval is specified by the `sensor.thresholdtime` property in the sensor's `global.xml` file. For example, if `sensor.thresholdtime` is set to `4` hours, then six work packages will have been collected in the `oirawdata` location at the end of 24 hours. All six packages are stamped with the creation time, and they all contain a `data.txt` file.
- The `sesrawdata` location collects object impressions continuously as a single work package during a 24-hour period. The work package is stamped with its creation time and contains a `data.txt` file.

Any one of the work packages in the `oirawdata` location contains only a portion of the day's raw data. A work package in the `sesrawdata` location contains the complete set of the day's raw data. In both locations, each work package is analyzed as soon as it is complete and computational resources are available.

Processed Data

All work packages are collected for a 24-hour period (see "Data Collection"). They are processed on a daily basis. For visitor data, additional work packages are created to represent weekly and monthly statistics. The work package directory structure for weekly and monthly processing differs from the directory structures for daily processing and data collection.

Note

Analytics administrators can obtain the directory structures of locations and paths to work packages from the HDFS file browser:

```
http://<hostname_MasterNode>:50070/
```

Directory Structure for Raw Data and oiprocessed Data

Work packages that contain raw data and `oiprocessed` data are stored in directories with a structure that identifies the day and time that the work packages were created. The

following locations use a day-time directory structure: **oirawdata**, **sesrawdata**, and **oiprocessed** *(an exception, as this location contains processed data)*.

The path to a raw data or oiprocessed work package is the following:

```
/<hadoop.hdfs.defaultfs>/<location>/<yyyy>/<mm>/<dd>/
   <workpackageDir>-<n>-<time>/data.txt <or part-<xxxxx>
```

For example:

```
/analytics/data/oirawdata/2009/01/14/Sensor-localhost.localdomain-
10-1231924822209/data.txt
```

The variables are defined as follows:

- <hadoop.hdfs.defaultfs> is the location of the root directory under which raw data, output, and cache files are stored on the Hadoop file system. For more information about the hadoop.hdfs.defaultfs property, see Table C-1, "Analytics parameters in global.xml" (page 125).
- <location> is the name of the location that stores the raw data work package(s). Valid values for <location> are the following:
  - oirawdata, collects data into multiple work packages during a 24-hour period
  - sesrawdata, collects data into a single work packages during a 24-hour period
  - oiprocessed (although this location contains processed data)

  (For a list of locations and descriptions, see Table A-1, on page 93.)
- <yyyy> is the year in which the work package was created.
- <mm> is the month in which the work package was created.
- <dd> is the day on which the work package was created. The day is determined from the site's time zone.
- <workpackageDir> is the sensor name (the IP address or host name of the data capture server)
- <n> is a system-generated number
- <time> is the creation time of the work package, computed in milliseconds elapsed since January 1, 1970.
- data.txt is the file that contains object impression. Raw data in the object impressions will be statistically analyzed by the processor that reads the file. Data.txt files are stored in the oirawdata location and in the sesrawdata location, as explained in "Data Collection," on page 95.

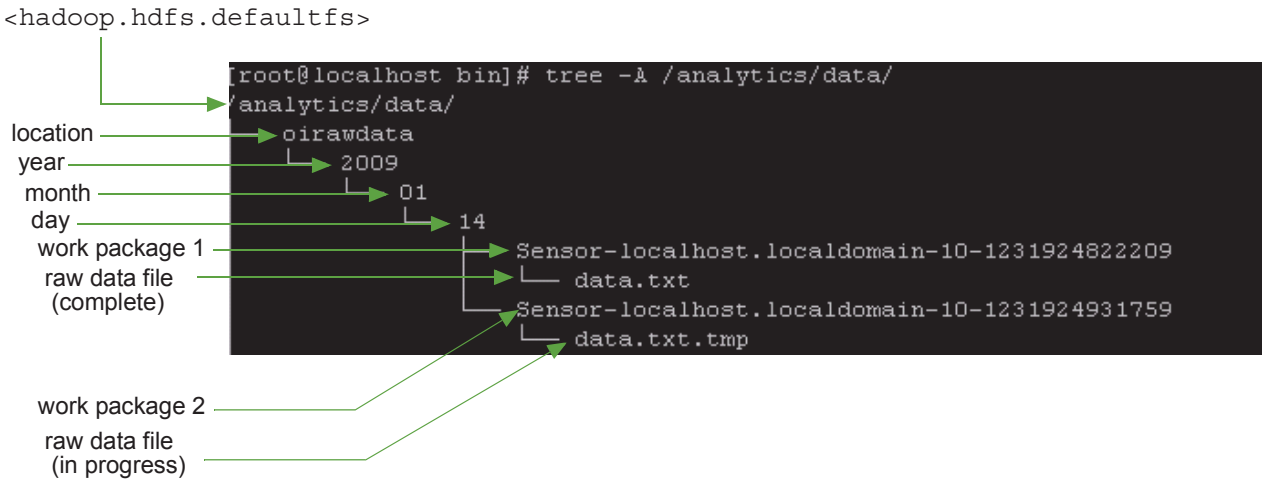
  Note that all raw data files are named data.txt. The <time> stamp in the work package directory containing the data file uniquely identifies the data file.
- <part-xxxxx> is the name of the work package in the oiprocessed location.

Figure A-2 illustrates the directory structure of the oirawdata location. The first data.txt file that was created in the oirawdata location on January 14, 2009 is stored as shown in Figure A-2:

```
/analytics/data/oirawdata/2009/01/14/Sensor-
   localhost.localdomain-10-1231924822209/data.txt
```

The second data file `data.txt.temp` is in progress. (At the end of the collection interval, the file will be complete and will take the name `data.txt`.) The file is stored as shown in Figure A-2.

Figure A-2: `oirawdata` directory structure



Directory Structure for Daily Work Packages

When a raw data work package is complete, the associated processor statistically analyzes the work package's `data.txt` file and writes the results to the work package in the next location for pickup by the next processor.

Work packages that contain daily statistics are stored in directories with a structure that identifies the day on which the work package was created. The following locations use a day-based directory structure: **`oiinjected`**, **`sesdata`**, **`sesprocessed`**, **`sesinjected`**, **`visdata`**, **`visprocessed`**, and **`visinjected`**

The path to a daily work package is the following:

```
/<hadoop.hdfs.defaultfs>/<location>/<yyyy>/<mm>/<dd>/
   <workpackageID>/part-<xxxxx>
```

For example:

```
/analytics/sesprocessed/2009/06/25/181bd6cd-c040-46a2-abb4/
   part-00000
```

The variables are defined as follows:

- `<location>` is the name of the location that stores the daily work package(s). Valid values for `<location>` are the following:
- `oiinjected`, `sesdata`, `sesprocessed`, `sesinjected`, `visdata`, `visprocessed`, and `visinjected`
- `<workpackageID>` is a system-generated number used to identify the work package.
- The remaining variables are defined in "Directory Structure for Raw Data and oiprocessed Data," on page 95.

Directory Structure for Weekly Work Packages

Work packages that are processed for weekly statistics are stored in directories with a structure that identifies the ISO week in which the work package was stored. The following locations use a week-based directory structure: **visweekdata**, **visweekprocessed**, **visweekinjected**

The path to a weekly work package is the following:

- visweekdata

  /<hadoop.hdfs.defaultfs>/**visweekdata**/<yyyy>/W<no.>/<yyyy>/<mm>/<dd>/<workpackageID>/part-<xxxxx>

  For example:

  /analytics/**visweekdata**/2009/W26/2009/06/25/1db1039-0b10-417d-9895/part-00000

  The variables are defined as follows:

  - W<no.> represents the number of the week in the given year.
  - <workpackageID> is a system generated number.
  - The remaining variables are defined in "Directory Structure for Raw Data and oiprocessed Data," on page 95.

- visweekprocessed and visweekinjected

  /<hadoop.hdfs.defaultfs>/<location>/<yyyy>/W<no.>/<workpackageID>/part-<xxxxx>

  For example:

  /analytics/**visweekprocessed**/2009/W26/9fe7607b-31b1-417d-9895/part-00000

  The variables are defined as for visweekdata.

Directory Structure for Monthly Work Packages

Work packages that are processed for monthly statistics are stored in directories with a structure that identifies the month in which the work package was stored. The following locations use a month-based directory structure: **vismonthprocessed** and **vismonthinjected**

The path to a monthly work package is the following:

/<hadoop.hdfs.defaultfs>/**<**location**>**/<yyyy>/<mm>/<workpackageID>/part-<xxxxx>

For example:

/analytics/**vismonthprocessed**/2009/06/c3b9ex84-0417-4b6f-9e38/part-00000

The variables are defined as follows:

- <workpackageID> is a system-generated number used to identify the work package
- The remaining variables are defined in "Directory Structure for Raw Data and oiprocessed Data," on page 95.

Processor Descriptions

Analytics supports three types of processors. They analyze the same object impressions, collected within a 24-hour period, but they perform their computations differently.

- Object Impression Processors analyze object impressions directly, by computing the frequency of occurrence of each type of data within the object impressions.
- Session Data Processors analyze session objects derived from the object impressions.
- Visitor Data Processors analyze visitor data derived from the session objects (which are created by the session data processors).

Object Impression Processors

- OIProcessor
- OIInjection

OIProcessor

Output: Intermediate daily sums. This processor reads each work package that is created in the `oirawdata` location and computes an intermediate daily sum (i.e., frequency of occurrence) for all types of data within the object impression.

Daily sums are called *intermediate* when they are computed for a work package containing less than 24 hours of data. Work packages are collected into the `oirawdata` location throughout the day, at the interval specified by the `sensor.thresholdtime` property in the sensor's `global.xml` file (for example, every 4 hours). Each work package then holds data that was collected for the specified interval—4 hours, in our example. At the end of 24 hours, six work packages will have been collected in the `oirawdata` location.

Details of the computation process are described below:

1. When a work package collected in the `oirawdata` location is complete, `OIProcessor` reads the data file in the work package and counts (i.e., sums, aggregates) the number of occurrences of each selected type of raw data in the work package. Thus, a work package has a set of intermediate daily sums, one for each selected type of raw data in the package. (If six work packages are collected and processed over a 24 hour period, then each work package has its own set of intermediate daily sums.)
2. `OIProcessor` writes the intermediate daily sums for each work package to the `oiprocessed` location for pickup by the `OIInjection` processor.

   Intermediate daily sums written to the `oiprocessed` location are counted (i.e., summed, aggregated) by the `OIInjection` processor and injected into the Analytics database. The sum of intermediate daily sums for a given type of raw data is the grand total for the day *for that type of raw data*; it is called the **complete daily sum**, or **aggregated daily sum**.

Note

The Content Server database and Analytics database are not synchronized. Therefore, Analytics creates an `L2ObjectBean` object for each unique object impression. The `L2ObjectBean` saves the object impression's name (title) and object (asset) id in the `L2_Object` table of the Analytics database.

Input Location for `OIProcessor`

`oirawdata:` Stores the current day's `data.txt` file (and a metadata file). More information about `data.txt` can be found in "Data Collection," on page 95.

`oirawdata`	
Directory structure	See "Directory Structure for Raw Data and oiprocessed Data," on page 95.
Work package	`<workpackageDir>-<n>-<time>/data.txt`
Work package data file	Contains beans of type `TransferObject`.
Data source	`oirawdata` folder on the local file system. The `oirawdata` location in the Hadoop File System is a duplicate of the `oirawdata` directory on the server where the Analytics Sensor is installed. Every 10 minutes (or the time interval that is explicitly set for the `sensor.thresholdtime` property), the Analytics Sensor creates a new work package (which is the input for the `OIProcessor`). Every object impression captured by the Analytics Sensor results in one line of data in the work package. The `/<dd>` folder (see "Directory Structure for Raw Data and oiprocessed Data," on page 95) represents the day of the object impressions (the day the work package was created).
Work package used by	This processor.

Output Location for `OIProcessor`

`oiprocessed:` Stores the work packages of this processor. Each work package contains a data file with intermediate daily sums for the work package (i.e., the frequency of occurrence of each type of data that was collected into the work package). (Each work package also contains a metadata file.)

`oiprocessed`	
Directory structure	See "Directory Structure for Raw Data and oiprocessed Data," on page 95.
Work package	`<workpackageDir>-<n>-<time>/part-<xxxxx>`
Work package data file	Contains beans of type `L2ObjectBean`, `L3ObjecttypeBean`, `L3DownloadBean`, `L3InternalSearchBean`, and `L3ObjectBean` (and custom pre-aggregated object impression data).
Data source	This processor.
Work package used by	OIInjection processor.

OIInjection

Output: Complete daily sums for specific types of data (i.e., the frequency of occurrence of each type of data that was collected during the last 24 hours). Injection status report.

1. This processor reads the intermediate daily sums in the data files of work packages in the `oiprocessed` location, and counts (i.e., sums, aggregates) the intermediate daily sums. The result is a grand total — a complete daily sum for each type of data that was collected during the last 24-hour period.
2. This processor injects the complete daily sums into various tables in the Analytics database, and creates a status report in the `oiinjected` location. (For information about intermediate daily sums, see "OIProcessor," on page 99.)

Data injected into the database is retrieved into the reports that Analytics users generate.

Input Location for `OIInjection` Processor

`oiprocessed`
See page 101.

Output Locations for `OIInjection` Processor

Analytics database: Stores the output of this processor. The output is *complete* daily sums (i.e., the frequency of occurrence of each type of data that was collected during the last 24- hours).

`oiinjected`: Stores the work package created by this processor. Note that the work package does not contain a data file. It contains only the metadata file indicating the status of the injection process.

`oiinjected`	
Directory structure	See "Directory Structure for Daily Work Packages," on page 97.
Work package	`<workpackageID>/part-<xxxxx>`
Work package data file	No data file is created in this work package. The metadata file `(.txt)` reports the status of the injection process.
Data source	This processor.
Work package used by	None of the processors. Administrators can open the HDFS file browser to view the metadata file (injection status report).

Session Data Processors

- SessionMerger
- SessionProcessor
- SessionInjection

SessionMerger

Output: Session objects for the last 24 hours of session data (i.e., aggregated object impressions grouped by their respective sessions and stored in the `sesdata` location).

This processor reads the object impressions in the data file of the work package in the `sesrawdata` location. It takes session data from the object impressions and combines the data to create a session object for each entire session. The session object contains all the information that relates to the specific session. In this manner, `SessionMerger` aggregates all object impressions collected during a 24-hour period into their respective sessions. This processor writes the aggregated data (as a work package) to the `sesdata` location (for pickup by the SessionProcessor).

Input Location for `SessionMerger` Processor

`sesrawdata`: Stores the current day's `data.txt` file (and a metadata file). More information about `data.txt` can be found in "Data Collection," on page 95.

`sesrawdata`	
Directory structure	See "Directory Structure for Raw Data and oiprocessed Data," on page 95. (Contains one work package for each calendar day.)
Work package	`<workpackageDir>-<n>-<time>/data.txt`
Work package data file	Contains beans of type `RawSensorCallBean`.
Data source	Analytics Sensor. The Analytics Sensor creates a new work package every 24 hours. (A **work package** contains the 24-hour interval of collected raw data, which is the input for the `SessionMerger` processor.) The work package folder in the directory structure represents the day of all the collected session information (the day is determined by the site's time zone).
Work package used by	`SessionMerger` processor.

Output Location for `SessionMerger` Processor

`sesdata`: Stores the work package of the `SessionMerger` processor. The work package's data file contains session objects for the last 24 hours of session data (i.e., aggregated object impressions grouped by their respective sessions). (The work package's metadata file contains the data processing status report.)

`sesdata`	
Directory structure	See "Directory Structure for Daily Work Packages," on page 97. (Contains one work package for each calendar day.)
Work package	`<workpackageID>/part-<xxxxx>`
Work package data file	Contains beans of type `SessionBean`.
Data source	This processor.
Work package used by	SessionProcessor.

SessionProcessor

Output: Complete daily sums of session data (i.e., frequency of occurrence of each type of data across sessions that ran during the last 24 hours).

This processor reads the session objects in the data file of the work package in the `sesdata` location, computes complete daily sums, and writes the results (as a work package) to the `sesprocessed` location for pickup by the SessionInjection processor.

Input Location for `SessionProcessor`

`sesdata`
See page 104.

Output Location for `SessionProcessor`

`sesprocessed`: Stores the work package created by this processor. The work package's data file contains complete daily sums of session data (i.e., the frequency of occurrence of each type of data across sessions that ran in the last 24-hours). (The work package's metadata file contains the data processing status report.)

`sesprocessed`	
Directory structure	See "Directory Structure for Daily Work Packages," on page 97. (Contains one work package for each calendar day.)
Work package	`<workpackageID>/part-<xxxxx>`
Work package data file	Contains beans of all `L3*` types, such as `L3BrowserBean`, `L3ClickStreamBean`, and `L3SessionEntryBean` (and custom pre-aggregated data).
Data source	This processor.
Work package used by	SessionInjection processor.

SessionInjection

Output: Injection status report.

This processor reads the complete daily sums in the data file of the work package in the `sesprocessed` location, injects the complete daily sums into various tables in the Analytics database, and creates a status report in the `sesinjected` location. Data injected into the database is retrieved into the reports that Analytics users generate.

Input Location for `SessionInjection` Processor

`sesprocessed`
See page 105.

Output Locations for `SessionInjection` Processor

Analytics database: Stores complete daily sums of session data (i.e., frequency of occurrence of each type of data across sessions that ran in the last 24 hours).

`sesinjected`: Stores the work package created by this processor. Note that the work package does not contain a data file. It contains only the metadata file indicating the status of the injection process.

`sesinjected`	
Directory structure	See "Directory Structure for Daily Work Packages," on page 97.(Contains one work package for each calendar day.)
Work package	`<workpackageID>/part-<xxxxx>`
Work package data file	No data file is created in this work package. The metadata file `(.txt)` reports the status of the injection process.
Data source	This processor.
Work package used by	None of the processors. Administrators can open the HDFS file browser to view the metadata file (injection status report).

Visitor Data Processors

Visitor data identifies site visitors by their IP addresses, for example. If FatWire Engage is installed, visitor data includes the segments visitors belong to and the recommendations associated with those segments.

The following processors analyze visitor data:

- VisitorMerger
- VisitorProcessor
- VisitorWeekMerger
- VisitorMonthProcessor
- VisitorInjection
- VisitorWeekProcessor
- VisitorMonthInjection
- VisitorWeekInjection

VisitorMerger

Output: Raw site visitor data

This processor reads the visitor-specific data (such as segments and recommendations) from the data file in the work package of the `sesdata` location. It writes that visitor data (as a work package) to the `visdata` location, in raw format (not aggregated) in order to save all visitor IDs. The visitor data is not aggregated by this processor because it must be used in its raw form by other visitor data processors to compute daily, weekly, and monthly sums.

Input Location for `VisitorMerger` Processor

`sesdata`
See page 105.

Output Location for `VisitorMerger` Processor

`visdata:` Stores the work package created by this processor. The work package contains a data file with site visitor data in raw format. (The work package's metadata file contains the data processing status report.)

`visdata`	
Directory structure	See "Directory Structure for Daily Work Packages," on page 97.
Work package	`<workpackageID>/part-<xxxxx>`
Work package data file	Contains beans of type `VisitorLocationBean`, `VisitorBean` (and custom visitor-related data).
Data source	This processor.
Work package used by	VisitorProcessor VisitorWeekMerger processor VisitorMonthProcessor

VisitorMonthProcessor

Output: Complete monthly sums for visitor data (i.e., the frequency of occurrence of each type of visitor data that was collected during the last month).

This processor reads the raw visitor data in the data file of the work package in the `visdata` location and computes monthly sums. It writes the monthly sums (as a work package) to the `vismonthprocessed` location for pickup by the VisitorMonthInjection processor.

Input Location for `VisitorMonthProcessor`

`visdata`
See page 107.

Output Location for `VisitorMonthProcessor`

`vismonthprocessed`: Stores the work package created by this processor. The work package's data file contains complete monthly sums for visitor data. (The work package's metadata file contains the data processing status report.)

`vismonthprocessed`	
Directory structure	See "Directory Structure for Monthly Work Packages," on page 98.
Work package	`<workpackageID>/part-<xxxxx>`
Work package data file	Contains beans of type `L3CityBean`, `L3CountryBean`, `L3RegionBean`, `L3VisitorBean` (and custom visitor-related data).
Data source	This processor.
Work package used by	VisitorMonthInjection processor.

VisitorMonthInjection

Output: Injection status report.

This processor reads the complete monthly sums in the data file of the work package in the `vismonthprocessed` location, injects the complete monthly sums into the Analytics database, and creates a status report in the `vismonthinjected` location. Data injected into the database is retrieved into the reports that Analytics users generate.

Input Location for `VisitorMonthInjection` Processor

`vismonthprocessed`
See page 108.

Output Locations for `VisitorMonthInjection` Processor

Analytics database: Stores the data from this processor's input location.

`vismonthinjected`: Stores the work package created by this processor. Note that the work package does not contain a data file. It contains only the metadata file indicating the status of the injection process.

`vismonthinjected`	
Directory structure	"Directory Structure for Monthly Work Packages," on page 98.
Work package	`<workpackageID>/part-<xxxxx>`
Work package data file	No data file is created in this work package. The metadata file `(.txt)` reports the status of the injection process.
Data source	This processor.
Work package used by	None of the processors. Administrators can open the HDFS file browser to view the metadata file (injection status report).

VisitorProcessor

Output: Complete daily sums for visitor data (i.e., the frequency of occurrence of each type of visitor data that was collected during the last 24 hours).

This processor reads the raw visitor data in the data file of the work package in the `visdata` location. It then computes complete daily sums and writes the complete daily sums (as a work package) to the `visprocessed` location for pickup by the VisitorInjection processor.

Input Location for `VisitorProcessor`

`visdata`
See page 107.

Output Location for `VisitorProcessor`

`visprocessed`: Stores the work package created by this processor. The work package's data file contains complete daily sums for visitor data. (The work package's metadata file contains the data processing status report.)

`visprocessed`	
Directory structure	See "Directory Structure for Daily Work Packages," on page 97.
Work package	`<workpackageID>/part-<xxxxx>`
Work package data file	Contains beans of type `L3CityBean`, `L3CountryBean`, `L3RegionBean`, `L3VisitorBean` (and custom visitor-related data).
Data source	This processor.
Work package used by	VisitorInjection processor.

VisitorInjection

Output: Injection status report.

This processor reads the complete daily sums in the data file of the work package in the `visprocessed` location, injects the complete daily sums into various tables in the Analytics database, and creates a status report in the `visinjected` location. Data injected into the database is retrieved into the reports that Analytics users generate.

Input Location for `VisitorInjection` Processor

`visprocessed`
See page 110.

Output Locations for `VisitorInjection` Processor

Analytics database: Stores the data from this processor's input location.

`visinjected`: Stores the work package created by this processor. Note that the work package does not contain a data file. It contains only the metadata file indicating the status of the injection process.

`visinjected`	
Directory structure	See "Directory Structure for Daily Work Packages," on page 97.
Work package	`<workpackageID>/part-<xxxxx>`
Work package data file	No data file is created in this work package. The metadata file `(.txt)` reports the status of the injection process.
Data source	This processor.
Work package used by	None of the processors. Administrators can open the HDFS file browser to view the metadata file (injection status report).

VisitorWeekMerger

Output: Raw site visitor data from the `visdata` location merged into a weekly folder.

This processor reads the raw visitor data in the data file of the work package in the `visdata` location. It merges the raw site visitor data into the appropriate ISO-week directory (in the processor's work package). **This processor does not modify the data.** This processor then writes its work package to the `visweekdata` location.

Input Location for `VisitorWeekMerger` Processor

`visdata`
See page 107.

Output Location for `VisitorWeekMerger` Processor

`visweekdata`: Stores the work package created by this processor. The work package's data file contains raw site visitor data (from the `visdata` location) merged into a weekly directory. (The work package's metadata file contains the data processing status report.)

`visweekdata`	
Directory structure	See "Directory Structure for Weekly Work Packages," on page 98.
Work package	`<workpackageID>/part-<xxxxx>`
Work package data file	Contains beans of type `VisitorLocationBean` and `VisitorBean` (and custom visitor-related data).
Data source	This processor.
Work package used by	VisitorWeekProcessor VisitorMonthProcessor

VisitorWeekProcessor

Output: Complete weekly sums for site visitor data (i.e., the frequency of occurrence of each type of visitor data that was captured in the last week.)

This processor reads the weekly raw data in the data file of the work package in the `visweekdata` location. It computes weekly sums, and writes the weekly sums (as a work package) to the `visweekprocessed` location.

Input Location for `VisitorWeekProcessor`

`visweekdata`
See page 112.

Output Location for `VisitorWeekProcessor`

`visweekprocessed`: Stores the work package created by this processor. The work package's data file contains weekly sums for site visitor data (i.e., the frequency of occurrence of each type of site visitor data that was captured during the last week.) (The work package's metadata file contains the data processing status report.)

`visweekprocessed`	
Directory structure	See "Directory Structure for Weekly Work Packages," on page 98.
Work package	`<workpackageID>/part-<xxxxx>` **Note:** If the work package contains visitor data for the last week of the year, it will also contain visitor data for the new year ***if the week runs over to the new year***.
Work package data file	Contains beans of type `L3CityBean`, `L3CountryBean`, `L3RegionBean`, `L3VisitorBean` (and custom visitor-related data).
Data source	This processor.
Work package used by	VisitorWeekInjection processor.

VisitorWeekInjection

Output: Injection status report.

This processor reads the weekly sums in the data file of the work package in the `visweekprocessed` location, injects the weekly sums into the Analytics database, and creates a status report in the `visweekinjected` location. Data injected into the database is retrieved into the reports that Analytics users generate.

Input Location for `VisitorWeekInjection` Processor

`visweekprocessed`
See page 113.

Output Locations for `VisitorWeekInjection` Processor

Analytics database: Stores the data from this processor's input location.

`visweekinjected`: Stores the work package created by this processor. Note that the work package does not contain a data file. It contains only the metadata file indicating the status of the injection process.

`visweekinjected`	
Directory structure	See "Directory Structure for Weekly Work Packages," on page 98.
Work package	`<workpackageID>/part-<xxxxx>`
Work package data file	No data file is created in this work package. The metadata file (`.txt`) reports the status of the injection process.
Data source	This processor.
Work package used by	None of the processors. Administrators can open the HDFS file browser to view the metadata file (injection status report).

Appendix B

Hadoop Configuration Parameters

If you are installing Hadoop in pseudo- or fully distributed mode, you must configure the `hadoop-site.xml` file with the recommended settings, listed in this appendix.

This appendix contains the following sections:

hadoop-site.xml Configuration Properties

If you are installing Hadoop in pseudo- or fully distributed mode, you must configure the `hadoop-site.xml` on all master and slave computers as shown in this section.

To configure Hadoop

1. Configure the `hadoop-site.xml` file as shown in Table B-1. Your configured file should look similar to the sample file on page 119.
2. If you are installing in fully distributed mode, copy the configured `hadoop-site.xml` to all master and slave computers.

Table B-1: Properties in `hadoop-site.xml`

Property	Description	Sample Value

`fs.default.name`	Name of the default file system. A URI whose scheme and authority determine the `FileSystem` implementation.
	The URI's scheme determines the configuration property (`fs.SCHEME.impl`) that names the `FileSystem` implementation class.
	The URI's authority is used to determine the host, port (and so on) for a file system.
		`hdfs://<ipaddress>:<port1>`
		where
		`<ipaddress>` is the IP address of the master node, and
		`<port1>` is the port on which NameNode will listen for incoming connections.
		E.g.,
		`hdfs://10.112.69.247:9090`

`mapred.job.tracker`	Host and port on which the `MapReduce` job tracker runs.
	If this property is set to `local`, then jobs are run in-process, as a single map and reduce task.
		`<ipaddress>:<port2>`
		`local`
		E.g.,
		`10.112.69.247:7070.`
		Note: In fully distributed mode, enter the IP address of the master node.

`dfs.replication`	Default block replication. The number of replications for any file that is created in HDFS. The value should be equal to the number of `DataNodes` in the cluster. The default is used if `dfs.replication` is not set.	`<equal to the number of data nodes>`

Table B-1: Properties in `hadoop-site.xml`

Property	Description	Sample Value
`dfs.permissions`	Enables/disables permission checking in HDFS. • `true` enables permission checking in HDFS. • `false` disables permission checking, but leaves all other behavior unchanged. Switching from one value to the other does not change the mode, owner or group of files, or directories.	`true | false`
`hadoop.tmp.dir`	Hadoop file system location on the local file system.	`/work/hadoop/hadoop-0.18.2/tmp/hadoop-${user.name}`
`mapred.child.java.opts`	Java options for the `TaskTracker` child processes. The following parameter, if present, will be interpolated: `@taskid@` will be replaced by the current `TaskID`. Any other occurrences of `@` will be unchanged. For example: To enable verbose gc logging to a file named for the `taskid` in `/tmp` and to set the heap maximum to a gigabyte, pass a value of: `-Xmx1024m -verbose:gc -Xloggc:/tmp/@taskid@.gc` The configuration variable `mapred.child.ulimit` can be used to control the maximum virtual memory of child processes.	`-Xmx1024m`
`mapred.tasktracker.expiry.interval`	Time interval, in milliseconds, after which a TaskTracker is declared 'lost' if it does not send heartbeats.	`60000000`

Table B-1: Properties in `hadoop-site.xml`

Property	Description	Sample Value
`mapred.task.timeout`	Number of milliseconds before a task is terminated if it neither reads an input, writes an output, nor updates its status string.	`60000000`
`mapred.map.tasks`	Default number of map tasks per job. Typically set to a prime number, several times greater than the number of available hosts. Ignored when `mapred.job.tracker` specifies the local IP address.	`11`
`mapred.reduce.tasks`	Default number of reduce tasks per job. Typically set to a prime number, close to the number of available hosts.Ignored when `mapred.job.tracker` specifies the local IP address.	`7`
`mapred.tasktracker.map.tasks.maximum`	Maximum number of `map` tasks that will be run simultaneously by a TaskTracker. Specify a number that exceeds the value of `mapred.map.tasks`.	Integer that exceeds the value of `mapred.map.tasks`
`mapred.tasktracker.reduce.tasks.maximum`	Maximum number of `reduce` tasks that will be run simultaneously by a TaskTracker. Specify a number that exceeds the value of `mapred.reduce.tasks`.	Integer that exceeds the value of `mapred.reduce.tasks`

Sample hadoop-site.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>

<property>
  <name>fs.default.name</name>
  <value>hdfs://10.112.69.247:9090</value>
  <description>The name of the default file system.  A URI
whose
  scheme and authority determine the FileSystem implementation.
The
  uri's scheme determines the config property (fs.SCHEME.impl)
naming
  the FileSystem implementation class.  The uri's authority is
used to
  determine the host, port, etc. for a filesystem.</
description>
</property>

 <property>
  <name>mapred.job.tracker</name>
  <value>10.112.76.120:7090</value>
  <description>The host and port that the MapReduce job tracker
runs
  at.  If "local", then jobs are run in-process as a single map
  and reduce task.
  </description>
</property>

<property>
  <name>dfs.replication</name>
  <value>1</value>
  <description>Default block replication.
  The actual number of replications can be specified when the
file is created.
  The default is used if replication is not specified in create
time.
  </description>
</property>

<property>
  <name>dfs.permissions</name>
  <value>false</value>
  <description>
    If "true", enable permission checking in HDFS.
    If "false", permission checking is turned off,
    but all other behavior is unchanged.
```

```
    Switching from one parameter value to the other does not
change the
    mode,owner or group of files or directories.
  </description>
</property>

<property>
  <name>hadoop.tmp.dir</name>
  <value/work/hadoop/hadoop-0.18.2/tmp/hadoop-${user.name}</
value>
  <description>A base for other temporary directories.</
description>
</property>

 <property>
  <name>mapred.child.java.opts</name>
  <value>-Xmx200m</value>
  <description>Java opts for the task tracker child processes.
  The following symbol, if present, will be interpolated:
@taskid@ is
  replaced  by current TaskID. Any other occurrences of '@'
will go
  unchanged.

   For example, to enable verbose gc logging to a file named
for the
  taskid in  /tmp and to set the heap maximum to be a gigabyte,
pass a
  'value' of: -Xmx1024m -verbose:gc -Xloggc:/tmp/@taskid@.gc
  The configuration variable mapred.child.ulimit can be used to
control
  the maximum virtual memory of the child processes.
  </description>
</property>

<property>
  <name>mapred.tasktracker.expiry.interval</name>
  <value>600000</value>
  <description>Expert: The time-interval, in miliseconds, after
which
  a tasktracker is declared 'lost' if it doesn't send
heartbeats.
  </description>
</property>

<property>
  <name>mapred.task.timeout</name>
  <value>600000</value>
  <description>The number of milliseconds before a task will be
  terminated if it neither reads an input, writes an output,
nor
  updates its status string.
```

```
  </description>
</property>

<property>
  <name>mapred.map.tasks</name>
  <value>2</value>
  <description>The default number of map tasks per job.
Typically set
  to a prime several times greater than number of available
hosts.
  Ignored when mapred.job.tracker is "local".
  </description>
</property>

<property>
  <name>mapred.reduce.tasks</name>
  <value>1</value>
  <description>The default number of reduce tasks per job.
Typically
    set to a prime close to the number of available hosts.
Ignored
     when mapred.job.tracker is "local".
  </description>
</property>

<property>
  <name>mapred.tasktracker.map.tasks.maximum</name>
  <value>2</value>
  <description>The maximum number of map tasks that will be run
  simultaneously by a task tracker.
  </description>
</property>

<property>
  <name>mapred.tasktracker.reduce.tasks.maximum</name>
  <value>2</value>
  <description>The maximum number of reduce tasks that will be
run
  simultaneously by a task tracker.
  </description>
</property>

</configuration>
```

Appendix C

FatWire Analytics Configuration Parameters

This appendix describes the configuration files that are updated by the silent installer when it installs FatWire Analytics. This appendix also provides guidelines for setting properties in the configuration files. The files are:

- global.xml
- log4j.properties
- futuretense_xcel.ini

Overview

When installing FatWire Analytics, the silent installer sets the values of various properties to match the values that you specified in `analytics-build.properties` (see "Step 2. Customize analytics-build.properties for Analytics," on page 29). The properties are stored in the following files:

- global.xml
- log4j.properties
- futuretense_xcel.ini

Once Analytics is installed, you can tune various properties directly in their respective files, as necessary. Guidelines are available in the rest of this appendix.

Caution!

Installation directories (and related configurations) created by the silent installer must not be moved, renamed, or modified in any other way.

Properties that must not be reset are shaded in the tables where they are listed.

global.xml

The Analytics silent installer modifies the `global.xml` file on each server where the file is stored and where the installer is executed. Once Analytics is installed, you can customize `global.xml` directly on its respective hosts. The customizable sections are:

- Parameters Within <params> </params>
- Database Connection Parameters
- LFS Logwriter Implementation Parameters

Note

Properties that must not be reset are shaded in the tables below. Each application has its own `global.xml` file, located in the application's classpath.

Parameters Within <params> </params>

This section contains properties that define Analytics installation directories, system administrators' contact information, the handling of raw data, and data processing conditions.

Table C-1: Analytics parameters in `global.xml`

Parameter	Description	Sample Value / Format	Host
`swchart_instdir`	Absolute path to the installation directory of the Swiff Chart Generator.	**Unix:** `/usr1/software/SwiffChart` **Windows:** `C:/Program Files/GlobFX/Swiff Chart Generator 3`	
`engine_instdir`	Absolute path to the directory into which the Analytics application was deployed.	**Unix:** `/usr/share/tomcat/analytics` **Windows:** `C:/CS/tomcat5/webapps/analytics`	
`href_cs`	Context URL of the Content Server interface.	`http://analytics.yourcompany.com/analytics`	
`report_instdir`	Absolute path to the deployed report configuration XML files.	**Unix:** `/data/analytics/reports` **Windows:** `C:/CS/reports`	

Table C-1: Analytics parameters in `global.xml` *(continued)*

Parameter	Description	Sample Value / Format	Host
`forgotpassword`	E-mail address of the analytics administrator who is responsible for password recovery. This address is the value for the **Forgot Your Password?** link in the Analytics login form (and should match the address for the **Forgot Your Password?** link in the Content Server login form). On Content Server systems, the `forgotpassword` property is located in the `uiadmin.properties` file. For more information, see the *Property Files Reference*.	`admin@yourcompany.com`	Reporting Node
`noaccount`	E-mail address of the analytics administrator who is responsible for creating accounts. This address is the value for the **Don't Have an Account?** link in the Analytics login form (and should match the address for the **Don't Have an Account?** link in the Content Server login form). On Content Server systems, the `forgotpassword` property is located in the `uiadmin.properties` file. For more information, see the *Property Files Reference*.	`admin@yourcompany.com`	Reporting Node
`href_reporting`	URL of the Analytics Reporting application. This URL is required to link the Analytics Reporting application to the Analytics Administrator application.	`../analytics/`	
`href_admin`	URL of the Analytics Administrator application. This URL is required to link the Analytics Administrator application to the Analytics Reporting application.	`../analyticsadmin/`	
`href_help`	URL at which help relating to FatWire Analytics can be obtained.	`http://e-docs.fatwire.com`	
`encoding`	Character encoding to be used for decoding request parameters and encoding response parameters. This encoding should match the encoding of the application server	`utf8`	

Table C-1: Analytics parameters in `global.xml` *(continued)*

Parameter	Description	Sample Value / Format	Host

`hadoop.hdfs.defaultfs`	Location of the root directory under which raw data, output, and cache files are stored on the Hadoop file system.	`hdfs://<hostname>:<port>/analytics`
		where:
		`<hostname>` is the name of the master node
		`<port>` is the `NameNode` port specified in the `fs.default.name` configuration parameter in `hadoop-site.xml`	

`hadoop.local.cachedir`	Local path of the folder that stores the `fileEnvObjects` at job startup.	`/usr/local/cache/`	
`analytics.filtercurrentdata`	Flag that specifies whether to skip processing data for the current day, week, month, and year. Set this property to `true` if wish to enable any of the properties listed in the row below. (The properties are: `analytics.filtercurrent`*XXX*)	**Default value**: `false`	Master Node

`analytics.filtercurrent`*XXX*
(where *XXX* stands for `day`, `week`, `month`, or `year`)
	Flag that specifies whether to skip processing data for the current day/week/month/year. Processing for each day/week/month/year can be set individually by adding the following parameters and setting them to `false`:
	`analytics.filtercurrentday`
	`analytics.filtercurrentweek`
	`analytics.filtercurrentmonth`
	`analytics.filtercurrentyear`
	Note: If you add one or more of the above parameters, make sure that `analytics.filtercurrentdata` (listed above) is set to `true`.
		`analytics.filtercurrentday=false`	Master Node

`admin.context`	URL of the Analytics Administrator application. Add a trailing slash (/).	`http://<hostname>:<port>/analyticsadmin/` where: `<hostname>` is the name of the Admin Server `<port>` is the port of the application server	

Table C-1: Analytics parameters in `global.xml` *(continued)*

Parameter	Description	Sample Value / Format	Host
`sensor.context`	URL of the sensor application. Add a trailing slash.	`http://<hostname>:<port>/sensor/` where: `<hostname>` is the name of the Data Capture server `<port>` is the port of the application server	
`monitoring.registry.port`	Port on which to start the RMI service.	`11199`	
`href.hadoop.tasktracker`	URL to the Hadoop TaskTracker admin interface.	`http://<hostname>:50030/` where: `<hostname>` is the name of the master node	
`href.hadoop.filesystem.browser`	URL to the `hdfs` file system browser interface.	`http://<hostname>:50070/` where: `<hostname>` is the name of the master node	
`importer.sleeptime`	Time interval (in minutes) after which `hdfsagent` will look for raw data to be copied from the local file system to `hdfs`.	`10`	Data Capture
`NumberOfProcessorThreads`	Number of jobs that run simultaneously. Each job is divided into tasks by Hadoop. For high-volume systems, `NumberOfProcessorThreads` should be set to `1`, so that only a single job can run at a given time. For low-volume systems or in demonstration scenarios, the value can be greater (`3`, `4`, or `5`).	`1`	Master Node
`sensor.thresholdtime`	Time interval (in minutes) after which the sensor will rotate the `data.txt.tmp` file (where incoming raw data is first written) to `data.txt`. Set a time interval such that no more than 5 to 10GB of data will be processed during the interval. If this parameter is omitted, the default threshold time (10 min) is used.	`240`	Data Capture

Table C-1: Analytics parameters in `global.xml` *(continued)*

Parameter	Description	Sample Value / Format	Host
`session.rotate.delay`	Interval of time (in minutes) after midnight that raw session data is kept open. The default is 360 minutes. This means that session data will be moved to HDFS 6 hours after midnight; session processing will then start.	`360`	Data Capture
`scheduler.checkinterval`	Specifies the frequency at which the scheduler will create new Hadoop jobs for fresh data. The default value is 15 minutes.	`15`	Master Node
`midnight.offset`	Allows the system to derive relative midnight used for file rotation. Relative midnight and `session.rotate.delay` determine when the daily cycle for capturing session data ends.	**Default value:** 0 **Format:** minutes	Data Capture
`cs_enabled`	Specifies whether buttons for navigating to the Content Server interface are enabled or disabled in the Analytics interface.	**Default value:** `true`	Reporting Application
`archive.enabled`	Specifies whether HDFS Agent archiving of raw data files is enabled. When this property is set to `true`, HDFS Agent will automatically create archives of raw analytics data on a periodic basis. The archive directory and start time are specified in the following properties: `archive.output.dir` and `archive.start.time`	**Default value:** `false`	Data Capture
`archive.output.dir`	Path to the directory for storing archived data files. Must be a valid URI.	**Format:** directory path	Data Capture
`archive.start.time`	Start time for archiving. The archiving task will start at *`HH:mm`* on a daily basis.	**Default value:** `06:00` **Format:** *`HH:mm`* in 24-hour time format. *`HH`* ranges from `00–23`; *`mm`* ranges from `00–59`.	.Data Capture
`purgejobs.enabled`	When this property is set to `true` the system will automatically schedule cleanup jobs to remove subfolders and files after they have been successfully processed.	**Default value:** `false`	Master Node

Table C-1: Analytics parameters in `global.xml` *(continued)*

Parameter	Description	Sample Value / Format	Host
`notification.enabled`	Indicates whether email notifications are enabled. Email notifications are sent when the availability of Analytics services changes.	**Default value:** `false`	Admin Node
`mail.from`	Email address from which notifications are sent.	**Format:** Email address	Admin Node
`sensor.requestqueue.maxsize`	Specifies `CRITICAL` condition for the Analytics Sensor. This property specifies a threshold value that triggers a `CRITICAL` (red) condition when the sensor cannot respond quickly enough to the amount of raw data that it needs to record. When the threshold is reached or exceeded, the **Analytics Sensor** component is displayed in red. The threshold value for this property is expressed as an object impression, i.e., a single invocation of the `sensor` servlet. (The **Analytics Sensor** component is represented in the **Components > Overview** panel of the Analytics Administration interface, shown in Figure A-1, on page 91.)	**Default value:** `10000`	Admin Node
`sensor.requestqueue.warnsize`	Specifies `WARNING` condition for the Analytics Sensor. This property specifies a threshold value that triggers a `WARNING` (yellow) condition when the sensor cannot respond quickly enough to the amount of raw data that it needs to record. When the threshold is reached or exceeded, the **Analytics Sensor** component is displayed in yellow. The threshold value for this property is expressed as an object impression, i.e., a single invocation of the `sensor` servlet. (The **Analytics Sensor** component is represented in the **Components > Overview** panel of the Analytics Administration interface, shown in Figure A-1, on page 91.)	**Default value:** `3000`	Admin Node

Database Connection Parameters

Typically, users require only one database connection. Custom reporting may require multiple connections. If you need to define your own JDBC resources or reference the existing JDBC connections via JNDI, use the following tag:

```
<connection
   name="<connection_name>"
   default="true"
   type="<jdbc_or_resource>"
   classname="<database_driver_classname>"
   url="<database_url>"
   user="<database user name>"
   password="<database password>" />
```

Table C-2: Database Connection Parameters

Parameter	Description
`name`	Name of the connection. **Example:** `localhostDB`
`default`	There must be exactly one connection marked with `default="true"`
`type`	Type of connection: `jdbc` (JDBC) or `resource` (JNDI)
`resourcename`	JNDI attribute; JNDI name **Note:** Used only if `type` is set to `resource`
`classname`	JDBC driver class. **Example:** `oracle.jdbc.driver.OracleDriver`
`url`	JDBC URL
`user`	JDBC attribute; database user name
`password`	JDBC attribute; database password

Example JDBC:

```
<connection
   name="jdbcsample"
   default="true"
   type="jdbc"
   classname="oracle.jdbc.driver.OracleDriver"
   url="jdbc:oracle:thin:@dbserver:1521:sid"
   user="analytics"
   password="analytics"/>
```

Example JNDI:

```
<connection
   name="conn1"
   default="false"
   type="resource"
   resourcename="java:comp/env/jdbc/tadev"/>
```

LFS Logwriter Implementation Parameters

The LFS logwriter implementation writes incoming raw data to the local file system. If you wish to change the root path (the location to which raw data will be written), use the following tag:

```
<logwriters>
   <logwriter type="LFS" name="LFS" rootpath="C:/analytics/
    sensorlocal" />
</logwriters>
```

Table C-3: Logwriter Parameters

Parameter	Description
`type`	Type of logwriter. **Legal value:** LFS
`name`	Alias name of the logwriter. **Legal value:** LFS

`rootpath`	Location, on the local file system, to which raw data will be written.
	Examples:
	Unix:
	`rootpath="/analytics/sensor"`
	Windows:
	`rootpath="c:/analytics/sensor"`

log4j.properties

The silent installer modifies the `log4j.properties` file on each server where the file is stored and the installer is executed. Once the Analytics installation is complete, you can customize properties directly in all or selected `log4j.properties` files, as shown in Table C-4. (The `log4j.properties` file is located in `<HADOOP_HOME>/conf`.)

Table C-4: Parameters in `log4j.properties`

Property	Description	Example / Format

`log4j.rootLogger`	Specify the log level and the appender of the root logger. Multiple appenders can be specified, separated by commas.	`log4j.rootLogger=INFO, DaRoFiAppender`
		- or -
		`log4j.rootLogger=INFO, DaRoFiAppender, ConsoleAppender`

`log4j.category. com.fatwire. analytics`	Specify the log level. The following explains the log levels in decreasing order of severity: • `FATAL` – Severe errors that cause premature termination. • `ERROR` – Runtime errors, or unexpected conditions. • `WARN` – Other runtime situations that are undesirable or unexpected, but not necessarily "wrong." • `INFO` – Provides informative messages about the workflow and status of the application. • `DEBUG` – Various kinds of debug information. • `TRACE` – All logging information. **Note:** In production mode, this property should be set to `WARN`.	`INFO`
`log4j.appender. DaRoFiAppender`	Specify the appenders to be used for logging.	`org.apache.log4j.Daily RollingFileAppender`
`log4j.appender. DaRoFiAppender. datePattern`	Specify the date pattern in the following format:	`'.'yyyy-MM-dd`
`log4j.appender. DaRoFiAppender. file`	Specify the location of the log file, along with the name of the log file.	`../logs/xxx.log`

Table C-4: Parameters in `log4j.properties`

Property	Description	Example / Format
`log4j.appender.DaRoFiAppender.layout`	Specify the layout.	`org.apache.log4j.PatternLayout`
`log4j.appender.DaRoFiAppender.layout.ConversionPattern`	Specify the layout pattern.	`%d{ISO8601}%-5p[%t]%c:%m%n`

Setting Up Logging for the Hadoop Job Scheduler

Edit `log4j.properties` file by adding the following parameters:

```
hadoop.root.logger=WARN,console, DRFA
hadoop.log.file=hadoop.log
log4j.rootLogger=${hadoop.root.logger}, DRFA, EventCounter

#
# Daily Rolling File Appender
#
log4j.appender.DRFA=org.apache.log4j.DailyRollingFileAppender
log4j.appender.DRFA.File=${hadoop.log.dir}/${hadoop.log.file}

# Rollver at midnight
log4j.appender.DRFA.DatePattern=.yyyy-MM-dd

# 30-day backup
log4j.appender.DRFA.MaxBackupIndex=30
log4j.appender.DRFA.layout=org.apache.log4j.PatternLayout

# Pattern format: Date LogLevel LoggerName LogMessage
log4j.appender.DRFA.layout.ConversionPattern=%d{ISO8601} %p %c:
%m%n
```

futuretense_xcel.ini

The silent installer modifies the `futuretense_xcel.ini` file to specify the location of the Analytic application and the authorized user. (The `futuretense_xcel.ini` file resides on the Content Server delivery system.)

Table C-5: FatWire Analytics Properties in `futuretense_xcel.ini`

Property	Description	Example
`analytics.datacaptureurl`	URL where the Analytics data capture servlet (`sensor` servlet) is running.	`http://<ipaddress>:<port>/sensor/statistic`
`analytics.enabled`	Indicates whether FatWire Analytics is available. Setting this property to `true` enables the **analytics** button in the Dash and Advanced interfaces. **Note:** If set to `false`, this property disables data capture.	`true`
`analytics.piurl`	URL where the Analytics performance indicator servlet is running. For information about the performance indicator, see "Managing the Performance Indicator," on page 62.	`http://<ipaddress>:<port>/analytics/PI`
`analytics.reporturl`	URL where the generated report is displayed.	`http://<ipaddress>:<port>/analytics/Report.do`
`analytics.user`	Pre-configured Analytics user who logs in to FatWire Analytics from Content Server.	`csuser` Default in FatWire Analytics. Changing the name is not recommended.

Appendix D

Configuring Visitor Detection

This appendix contains the following sections:

Overview

Visitor tracking is done by the data capture application (sensor). Visitors can be tracked in the following ways:

- Self-organized method, which supports tracking of new visitors
- Sessionfingerprint method
- Cookie method

Note

By default, the `Sessionfingerprint` method is enabled on all sites.

Methods

Methods for tracking visitors are implemented by adding visitor tracking code to `global.xml` (and to the `img` tag, depending on the tracking method).

Note

When changing the visitor tracking method, you must also update the `global.xml` configuration file for each sensor instance. The location of `global.xml` depends on the application server you are using.

When modifying parameters in the `global.xml file(s)`, stop the Analytics Sensor web application, modify its `global.xml` file, then start the sensor to effect the change.

Self-Organized Visitor Tracking

The self-organized visitor tracking method uses the value of the `cookieid` parameter appended to the image request.

To track the visitor

1. Add the following parameters to the `img` tag:

   `cookieid` – the visitor ID

   `newvisitor` – determines whether the user has already visited the site

2. If you need to change the name of the `cookieid` parameter (to `visitorid` for example), you must also do the following:

   a. Change the following line in `global.xml` as shown below:

   From:

   ```
   <alias name="cookieid" for="cookieid" />
   ```

   To:

   ```
   <alias name="visitorid" for="cookieid" />
   ```

   b. Pass the `visitorid` and `newvisitor` parameters to the `img` tag.

Sessionfingerprint Method

The `Sessionfingerprint` method identifies each visitor by a combination of the IP address, screen resolution, and agent string to assign sessions to visitors. You can enable `Sessionfingerprint` for all sites or a selected site. `Sessionfingerprint` is enabled for all sites by default when Analytics is first installed.

Note

The `Sessionfingerprint` method does not support *new* visitor tracking.

To enable the sessionfingerprint method

- To enable visitor tracking on all sites, add the lines in bold type below to `global.xml` (nest them in the default `<params> ... </params>` tag, as shown):

```
<params host="default">
<param type="string" name="sessionIdGenerator"
   value="AppServerId"/>
<param type="string" name="visitorIdGenerator"
   value="SessionfingerprintId"/>
...
</params>
```

- To enable visitor tracking on a selected site, add the lines in bold type below to `global.xml` (nest them in the `<root>` tag, as shown):

```
<root>
<params site="sitename">
<param type="string" name="sessionIdGenerator"
   value="AppServerId"/>
<param type="string" name="visitorIdGenerator"
   value="SessionfingerprintId"/>
</params>

<params host="default"> ...
```

Table D-1: Parameters for `Sessionfingerprint` visitor tracking

Parameter	Description
`sessionIdGenerator`	Specifies the ID generator that is used to identify sessions. **Caution:** The default value is `AppServerID`. Do not modify this value. It is a reference to the object that generates the ID.
`visitorIdGenerator`	Specifies the ID generator that is used to identify a visitor to the site. **Required value for this method:** `SessionfingerprintId`
`site`	Name of the site that is passed via the `img` tag

Cookie Method

Using the cookie method, you can enable visitor tracking across all sites or within a selected site.

To enable the cookie method

- **To enable visitor tracking on all sites**, add the lines in bold type below to `global.xml` (nest them in the default `<params> ... </params>` tag, as shown):

> **Note**
>
> When using the code below, replace the sample values with values of your own. For parameter definitions, see Table D-2, on page 141.

```
<params host="default">
<param type="string" name="visitorIdGenerator"
   value="CookieId"/>
<param type="string" name="visitorIdCookieName" value="visid"/>
<param type="string" name="visitorIdCookieMaxAge"
   value="31536000"/>
<param type="string" name="visitorIdCookieDomain"
   value="firstsiteii"/>
<param type="string" name="visitorIdCookiePath" value="/"/>
<param type="string" name="visitorIdCookieSecure"
   value="false"/>
...
</params>
```

- **To enable visitor tracking on a selected site**, add the lines in bold type below to `global.xml` (nest them in the `<root>` tag, as shown):

> **Note**
>
> When using the code below, replace the sample values with values of your own. For parameter definitions, see Table D-2, on page 141.

```
<root>
<params site="FirstSiteII">
   <param type="string" name="sessionIdGenerator"
value="AppServerId"/>
   <param type="string" name="visitorIdGenerator"
value="CookieId"/>
   <param type="string" name="visitorIdCookieName"
value="visid"/>
   <param type="string" name="visitorIdCookieMaxAge"
value="31536000"/>
   <param type="string" name="visitorIdCookieDomain"
value=".firstsiteii.at"/>
   <param type="string" name="visitorIdCookiePath" value="/"/>
   <param type="string" name="visitorIdCookieSecure"
value="false"/>
</params>
<params host="default">
...
```

Table D-2: Parameters for cookie-based visitor tracking

Parameter	Description
`sessionIdGenerator`	Specifies the ID generator that is used to identify sessions. **Caution:** The default value is `AppServerID`. Do not modify this value. It is a reference to the object that generates the ID.
`visitorIdGenerator`	Specifies the ID generator that is used to identify a visitor to the site. **Required value for this method:** `CookieId`
`visitorIdCookieName`	Name of the cookie set on the client side. **Example:** `visid`
`visitorIdCookieMaxAge`	Age of the cookie in milliseconds. After this time, the cookie is invalid. (The value that you set is added to the current time.) **Example:** `31536000`

`visitorIdCookieDomain`	Domain on which the cookie should be set.
	Example: `firstsiteii`
	Note: If you wish to set `visitorIdCookieDomain` on your local host, do not use the `localhost` value. It is not accepted by most browsers. Instead, add the following type of entry to your hosts file:
	For example,
	`127.0.0.1 firstsiteii.at`
	and change the `analytics.datacaptureurl` in `futuretense_xcel.ini` to:
	`http\://firstsiteii.at\:8081/sensor/statistic`

`visitorIdCookiePath`	Sub-path on which the cookie should be set. **Example:** `"/"`
`visitorIdCookieSecure`	Specifies that the cookie will be sent via a secure channel, such as an SSL connection. **Do not change the default value (`false`)**.
`site`	Name of the site that is passed via the `img` tag. **Example:** `FirstSite II`

Glossary

This glossary explains the terms used throughout this guide that are specific to FatWire Analytics.

Analytics Data Capture Application

Also referred to as the "sensor."

Asset Registration

Enabling report generation for assets. Because Content Server assets are specific to a CS installation, you must register their asset types with Analytics by assigning them to reports through the Analytics Administration interface. This enables Analytics to:

- Recognize Content Server asset types
- Configure report menu options in the "General Information" and "Content Information" report groups
- Generate reports on assets of the registered asset types

For instructions on assigning asset types to reports, see "Configuring an Asset Report," on page 60.)

Data Capture

The process of recording each visitor's clicks and the associated information—the date and time of each click, the assets that are clicked, the IP address from which the clicks are issued, the site being visited, and so on. The information is captured in real time by the `sensor` servlet and recorded in a `data.txt.tmp` file on the local file system (local to the Analytics data capture application). The `data.txt.tmp` file will be rotated by the sensor to `data.txt` when either the threshold interval is reached (see the `sensor.threshold` property on page 128), or the application server is restarted.

FatWire Analytics can capture data on the usage of Content Server assets and on their visitors only if published pages are tagged for data capture. In the case of Engage assets, the assets themselves must be tagged for data capture. For information on how to enable data capture, see Chapter 6, "Configuring Data Capture."

Hadoop Jobs

Runs jobs in a parallel and distributed fashion in order to efficiently compute statistics on the raw data that is stored in the Hadoop Distributed File System.

Hadoop implements a computational paradigm named `Map/Reduce`, which divides a large computation into smaller fragments of work, each of which may be executed on any node in the cluster. `Map/Reduce` requires a combination of `jar` files and classes, all of which are collected into a single `jar` file that is usually referred to as a "job" file. To execute a job, you submit it to a `JobTracker`. Hadoop Jobs then responds with the following actions:

- Schedules and submits the jobs to `JobTracker`.
- Processes raw data captured by the data capture application into statistical data and injects the statistics into the Analytics database.

(Hadoop provides a web interface to browse HDFS and to determine the status of the jobs.)

Hadoop jobs pre-calculate commonly requested site usage statistics (such as average number of requests for a piece of content per unit time) in order to shorten report generation time. Statistical computation is typically resource-intensive and time-consuming. Therefore, it is performed not on-the-fly, each time a report is generated, but in advance so that it can be available by the time it is needed. Thus, precalculated statistics are immediately available for retrieval into reports. Statistics include, for example:

- Current information, such as today's total hits to each site, visiting countries, total number of visits from a given country, types of browsers, and average session duration.
- Historical results, such as:
  - Daily, weekly, and monthly statistics—for example, the total number of requests for a given asset on a given site during a certain month in the reporting period.
  - Yearly statistics—a histogram in the performance indicator indicating the frequency with which certain assets were accessed during each week of the past year.

How long a Hadoop job runs depends on a number of factors, including site activity within the latest data capture time frame, the cumulative volume of captured data, and the configuration of the Analytics application. When data analysis is complete, the resulting statistics are available, at any time, for report generation.

Integration

Integrating Analytics with your Content Server system means enabling report generation for asset types and users on your online site. Integration involves registering CM sites, Content Server users, and asset types with Analytics, configuring the Pageview Object (through the "Page Views" Report), and granting users the appropriate permissions through membership in the appropriate user groups. The steps necessary to accomplish these tasks are described in Chapter 5, "Integrating Analytics with Content Server."

Internal Search

A search performed by a visitor using the site's built in search engine. This search returns results from within the site's contents.

Object

A FatWire Analytics construct. The subject of a report.

When storing and processing information, Analytics uses objects, whereas Content Server uses assets and asset types. To allow Analytics to recognize a Content Server asset type and track assets of that type, administrators define an Analytics object in terms of a

Content Server asset type. They do so by configuring an Analytics report for the object and assigning the desired asset type to that object. The process of configuring a report defines the underlying asset.

Note

A special instance of an object is the Pageview Object, which administrators must configure (by configuring the "Page Views" Report) in order for reports in the "General Information" group to work.

The "Page Views" report supports multiple asset types.

Object Impression

A single invocation of the `sensor` servlet. For more information, see "Object Impressions," on page 94.

Page View

A FatWire Analytics construct. A group of one or more assets, whose asset types are enabled for tracking by the Analytics data capture application.

Asset types are enabled for tracking when they are defined in the Pageview Object and when published pages displaying those asset types are tagged with `AddAnalyticsImgTag` (data capture tag). For more information about tracking, see "Data Capture," on page 143.

Pageview Object

A default Analytics object which you configure through the "Page Views" report. The `Pageview` object is the basis for the "Page Views," "Site Information" and "Clickstream" reports, and thus it should be assigned asset types whose assets make the most sense (from the marketing standpoint) to be included in these reports.

A `Pageview` object can be assigned multiple asset types. The "Page Views" report will contain statistics on the usage of those asset types.

"Page Views" Report

A report, based on the Pageview Object. The "Page Views" report displays statistics on Page View activity on your site.

Processed Data

Visitor activity data that has been processed by Hadoop Jobs into statistical data. When processing is complete, the data is injected into the Analytics database, where it is immediately available for the reports that users request from the Analytics reporting interface. *For information about data processing, see* Hadoop Jobs.

Raw Data

Unprocessed data describing visitor activity on the site, recorded during the Data Capture process and stored in the local file system for future processing. This is the data on which statistics are calculated by the Hadoop Jobs for display in reports. (This data cannot be directly used for report generation.)

Sensor

Also referred to as the "Analytics data capture application."

Site Registration

Identifying a Content Server CM site to Analytics in order to enable Analytics to track visitor activity on that site.

Statistical Data

See "Processed Data," on page 145.

Work Package

A collection of object impressions. For more information, see "Object Impressions and Work Packages," on page 94.