Developer's Guide

Genesys Web Engagement 8.1.1

12/30/2021

Table of Contents

Genesys Web Engagement Developer's Guide

This Developer's Guide includes the following introductory material to help you to understand and use Genesys Web Engagement:

- Product Overview — Start here for an overview of Genesys Web Engagement's features, components, and related components.
- Architecture — A look at the high-level architecture and event workflow for Genesys Web Engagement.
- Visitor Identification — Identify visitors and capture their activities on your website.
- Engagement Models — Define categories or business events with the Simple and Advanced Web Engagement models.
- Application Development — Contains information about the entire application development process, including how to create, build, and deploy your application; creating business information, catetories, and rules; customizing the widgets and engagement strategies; testing with the Web Engagement Proxy; and enabling monitoring.
- Genesyslab Sample — A full Genesys Web Engagement sample application you can deploy and test.
- Check Your Application Version — Details how you can check the version of your application and the Genesys Web Engagement Servers.
- Using the Plug-in for GAX — Provides information about the Plug-in for Genesys Administrator Extension and how you can use it to create your Web Engagement application.
- Using the Plug-in for IW — Provides information about how the Plug-in for Interaction Workspace extends the IW interface to support Genesys Web Engagement.

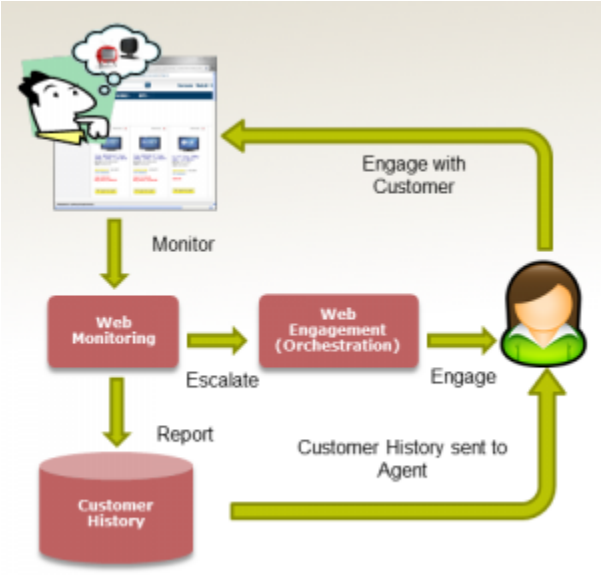
Product Overview

What is Genesys Web Engagement?

Genesys Web Engagement provides the ability to monitor, identify, and proactively engage web visitors in conversations that match business objectives. Customers are identified using robust business rules that provide a simple and comprehensive means for identifying key customers based on their behavior on your website and their value to your business. Key customers are then evaluated, leveraging the full power of Genesys Orchestration, and the best candidates are matched with the best agents, allowing you to better achieve your business objectives, including new customer acquisition, product sales, or customer support.

Genesys Web Engagement integrates the browsing activity of your web visitors into the overall Genesys customer service process. It records the customer web-browsing history, gathers accurate information, and converts it into Genesys interactions.

In addition to its monitoring features, Genesys Web Engagement enables you to engage the online customer by chat or voice callback.



From the customer standpoint, there is no visible change in the web experience.

Get to Know your Web Visitors

Genesys Web Engagement develops an understanding approach of the customer's interaction with your website. Its web monitoring features connects the web content as a self-service channel into the Context Services.

It translates the raw web activity into a form suitable to customer service.

- **Basic Usage Information.** Get to know if a customer has ever used the web channel, and if so, how recently (and/or how frequently).
- **Browsing History.** Each time that the customer browses your website, a session is created to store the visited pages in the customer history. This provides information on what the customer may have been looking for or may be interested in.
- **Activities and Outcomes.** Genesys Web Engagement allows you to tag web pages and define associations between URIs and outcomes to build a higher-level model of the customer browsing activity. This model is then usable to drive further interactions on other channels (chat and web callback, for now).

In addition, Genesys Web Engagement includes several scenarios for identified and unknown web visitors, detailed in Visitor Identification Scenario.

- If the user is not authenticated, the engagement decision should be taken by an agent or configured through custom Web Engagement rules.
- When engaging an unauthenticated user, Genesys Web Engagement asks for the user's registration.

Pro-active Follow-up

Genesys Web Engagement enables real-time or offline post processing of customers' web-browsing activity to identify potential for proactive follow-up. You can define service assistance to notify agents when some specific use cases should lead to a proactive follow-up. In addition, the flexibility of Genesys Web Engagement allows to submit this follow-up for validation to agents, in order to make sure that the follow-up is appropriate.

For example, some use cases could be:

- When a shopping cart is abandoned, it can be caused by a lack of information. A proactive follow-up by an agent - via any number of channels - can help to close the sale.
- If a customer bought a product several weeks ago but abandoned a transaction recently, an agent could call to ask about satisfaction with the earlier purchase and afterwards follow-up with a question about the abandonment.
- If a customer submits a bad rating or comments about one of your products, an agent could follow-up by e-mail with a survey about his dissatisfaction.

In addition, if the contact center decides to make a proactive offer, the Browser-Tier Component checks that the visitor is still present, then pops up a widget in the browser window ("Click here to chat"). If the visitor accepts the offer, the chat connection is made in the standard way using existing components.

Components

Genesys Web Engagement interfaces with the standard Genesys Call Center Solution and requires minimal changes to your website: to interface your website with this product, you simply add a JavaScript tracking code to your web pages. In addition, Standard Genesys interfaces, such as Composer, Genesys Rules Development Tool, and Rules Authoring Tool, enable you to develop and

deploy custom rules and business attributes to fine-tune your web engagement scenarios. Genesys Web Engagement is composed of three components, detailed in the Architecture page:

- **Web Engagement Browser-Tier Agents** are loaded in the JavaScript tracking code, which submits system and custom events based on the customer's browsing activity.
- **Web Engagement Frontend Server** manages the event flow submitted by the browser-tier agents and is responsible for the complex event processing and submitting actionable events.
- **Web Engagement Backend Server** interfaces with the Web Engagement Frontend Server and the Genesys Contact Solution to store web engagement information and implement web engagement action.

In addition to rules templates, deliverables include the following plug-ins:

- **Web Engagement Plug-in for Administrator Extension**, implementing:
  - **Script Generator**, to generate the standard JavaScript tracking code that you must add to your web pages.
  - **Categories**, to create custom business data.
- **Web Engagement Plug-in for Interaction Workspace**, to get all the web-based contexts routed to agents. This plug-in is mandatory to enable chat and web callback engagement features in Interaction Workspace.

Browser Support

Genesys Web Engagement supports the following web browsers:

- Google Chrome
- Mozilla Firefox 9.0+
- Microsoft Internet Explorer 7, 8, 9, 10
- Apple Safari 5.0+

Genesys Web Engagement supports the following mobile browsers:

- iOS Safari
- Android Chrome

Features

Genesys Web Engagement includes the following features:

- Integrated Proactive Chat and Proactive Web Callback
  Optimization / Pacing of Proactive Engagement Invitations

  Included Web Engagement applications for proactive Genesys Chat and Genesys WebCallback

- Behavior Rules Authoring for simplified tooling of Web Pages
  - Categorization - Key word and regular expressions for out-of-the box web page identification
  - Out-of-the box business events for capturing searches and timeout as part of behavior rules
  - Out-of-the box rule templates and business rules interface for defining engagement rules based on customer behavior
  - Support for advanced business events to capture events not covered by categories
  - Business User friendly UIs for creating both Categories and Business rules
- Monitoring and data storage of customer web activity
  - Storage of web history for authenticated customers
  - RESTful API for full access to web history
- Integrated Agent Interface for Interaction Workspace
  - Out-of-the box Agent Desktop support
  - Live monitoring of customer during engagement
  - Integrated view of Web History
- Reporting – Historical and Real Time
  - Templates for out-of-the box real-time interaction reporting of Web Engagement
- Integrated with core Genesys product suite

Related Components

Genesys Web Engagement interacts with the following Genesys Products:

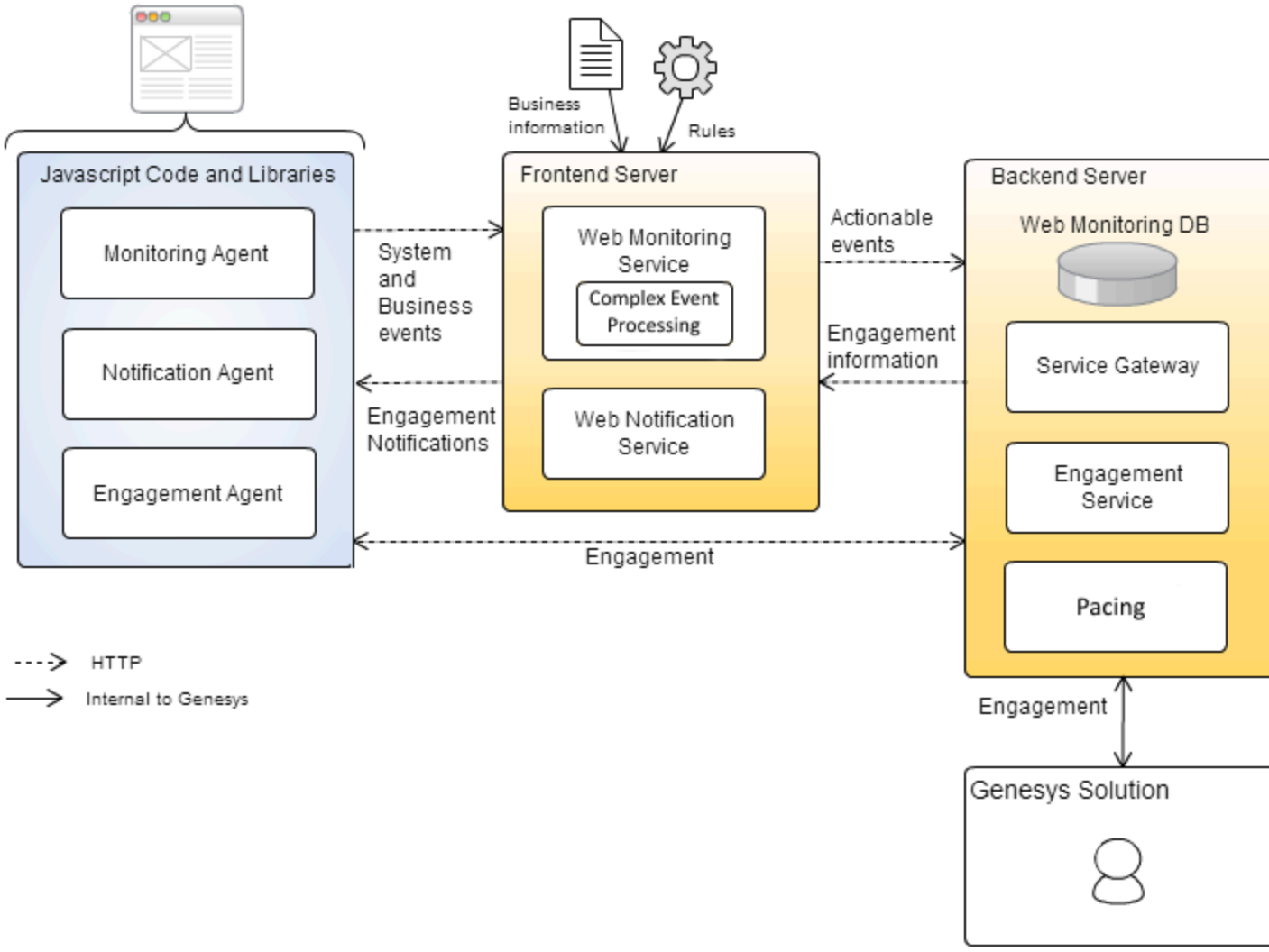
- Interaction Workspace—The Genesys Web Engagement Plug-in for Interaction Workspace is required to interface Genesys Interaction Workspace with Web Engagement. This plug-in enables you to get all the web-based contexts routed to agents. This plug-in is mandatory to enable chat and voice engagement.
- Context Services
- Orchestration Server
- Stat Server
- SIP Server
- Interaction Server
- Chat Server

Architecture

High Level Overview

Genesys Web Engagement provides web services to interface your website with the Genesys Contact Center Solution:

- The Browser Tier widgets or **Agents** are embedded in javascript code snippets inserted into your webpages; they run in the customers' browser and track their browsing activity;
- The **Frontend Server** includes the Web Monitoring Service and the Web Notification Service, responsible for managing the data and event flow, based on a set of configurable rules and customer's defined business events;
- The **Backend Server** stores data, submits information to the Genesys Solution, and manages engagement requests to the Genesys Contact Center Solution.



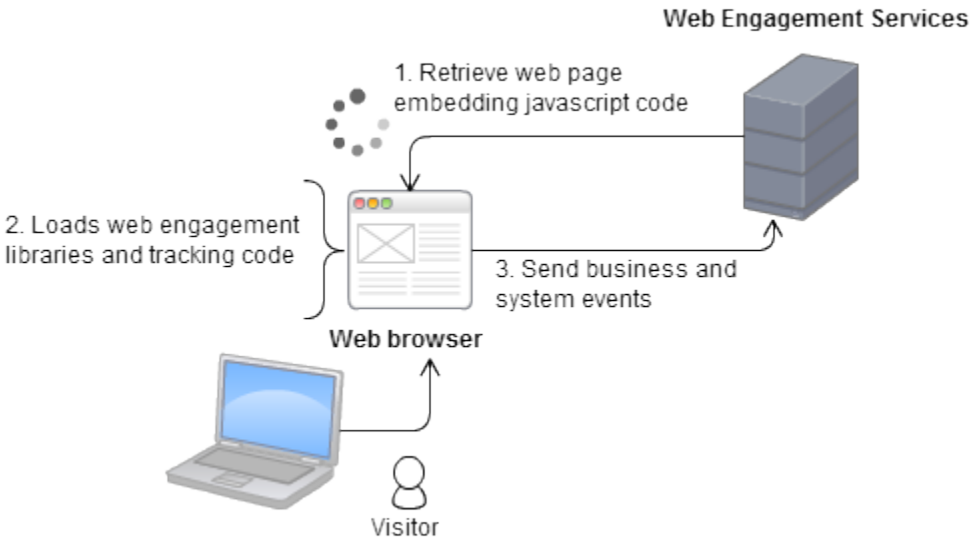
High level architecture of Genesys Web Engagement

Browser Tier Agents

The Browser Tier Agents are implemented as JavaScript components that run in the customer's browser. To enable the monitoring of a web page, you create a short standardized section of JavaScript code with the Genesys Administrator Extension plug-in and then, you add this code to the `<html></html>` section of your web page. When a customer visits the webpage, the code retrieved within the page loads all the necessary artifacts like the JavaScript libraries and Domain Specific Language (DSL) rules embedded in the JavaScript code. The DSL rules cover:

- The HTML elements to monitor;
- The custom business events to send to the Frontend Server;
- The data to include in the events.

The Browser Tier generates categorized standard System and custom Business events, defined in the DSL definitions, and sends them to the Frontend Server over HTTP.



Browser interactions at runtime.

Genesys Web Engagement provides the following browser tier agents:

- The **Monitoring Agent** service records the web browsing activity. It generates basic system events such as `visit-started`, `page-entered`, and additional custom business events, such as 'add-to-shopping cart'. These events are sent to the Web Engagement Frontend Server for further processing. For further information about events, see Event Workflow.
- The **Notification Agent** provides the browser with the asynchronous notification of the engagement offer by opening an engagement invite. It opens the engagement window.
- The **Engagement Agent** provides the engagement mechanism, chat communication or web callback initialization.

If you are interested in monitoring features only, you do not need to install the Notification and Engagement Agent modules. Note that you cannot dynamically activate or deactivate the Notification and Engagement Agents on a visitor-by-visitor basis.

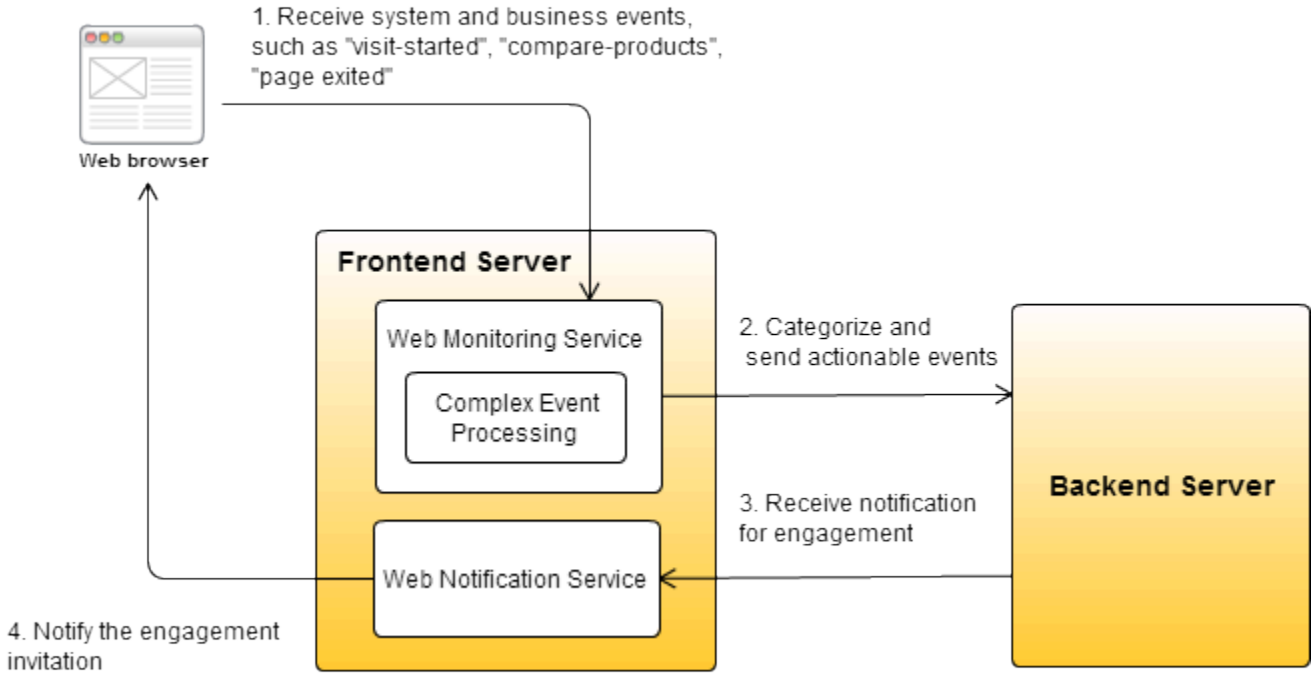
Web Engagement Frontend Server

The Genesys Web Engagement Frontend Server receives system and business events from the browser's Monitoring Agent through its RESTful interface.

- **SYSTEM** events are constants which cannot be customized. Two types of system events are available:
  - **Visit-related** events, such as *VisitStarted* or *PageEntered*;
  - **Identity-related** events, such as *SignIn*, *SignOut*, *UserInfo*.
    See Visitor Identification for further details.

- **BUSINESS** events are additional events that you can define with the Monitoring DSL to customize your application according to your requirements. For further details on their implementation, see Creating Business Events.

The Frontend Server performs an analysis of event correlation and attributes (such as the event name, event type, URL, or page title) and then assigns categories to the events. The integrated Complex Event Processing (CEP) engine validates these categorized events against the business rules and creates actionable events, which the Frontend Server sends to the Backend Server.

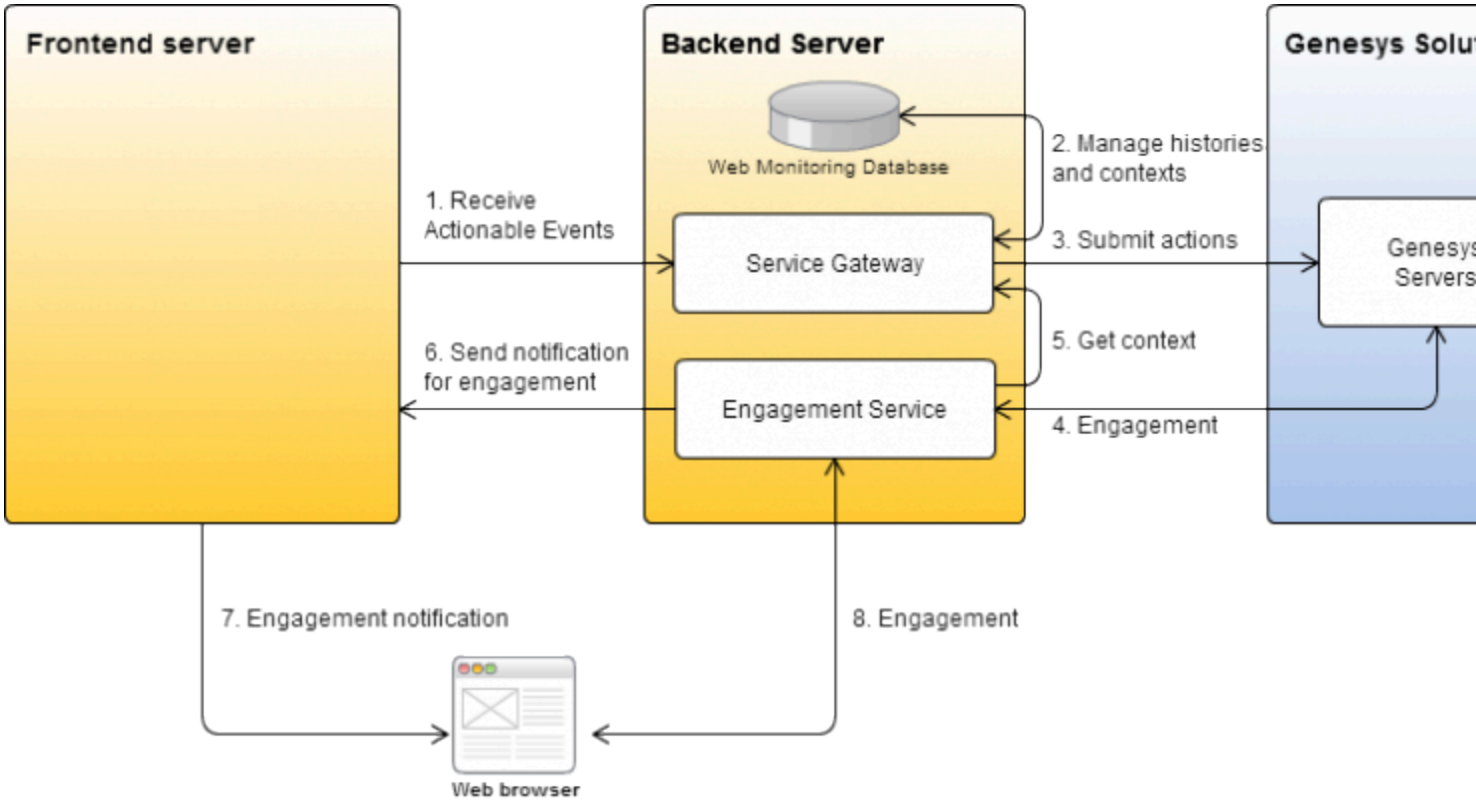


In addition, the Web Engagement Frontend Server also sends real-time invitation notifications to the Web Notification Agent of the web browser.

Web Engagement Backend Server

The Web Engagement Backend Server is the engagement's entry point for the Genesys Servers and delivers web information to the contact center, allowing correlation with contact information.

The Service Gateway stores the events received from the Frontend Server, manage contexts and histories in the database, and submits them to the Genesys Servers. Then, when the Engagement Service is notified to present a proactive offer, it retrieves the engagement context with the help of the Gateway, based on the visit attributes, and if the Orchestration rules authorizes it for the given web page, the proactive offer is displayed in the web page.



If the visitor accepts, the Engagement Service connects to the Genesys Servers. Once the connection is established, the Engagement Service manages the engagement context information across the visit.

Your Web Engagement Application

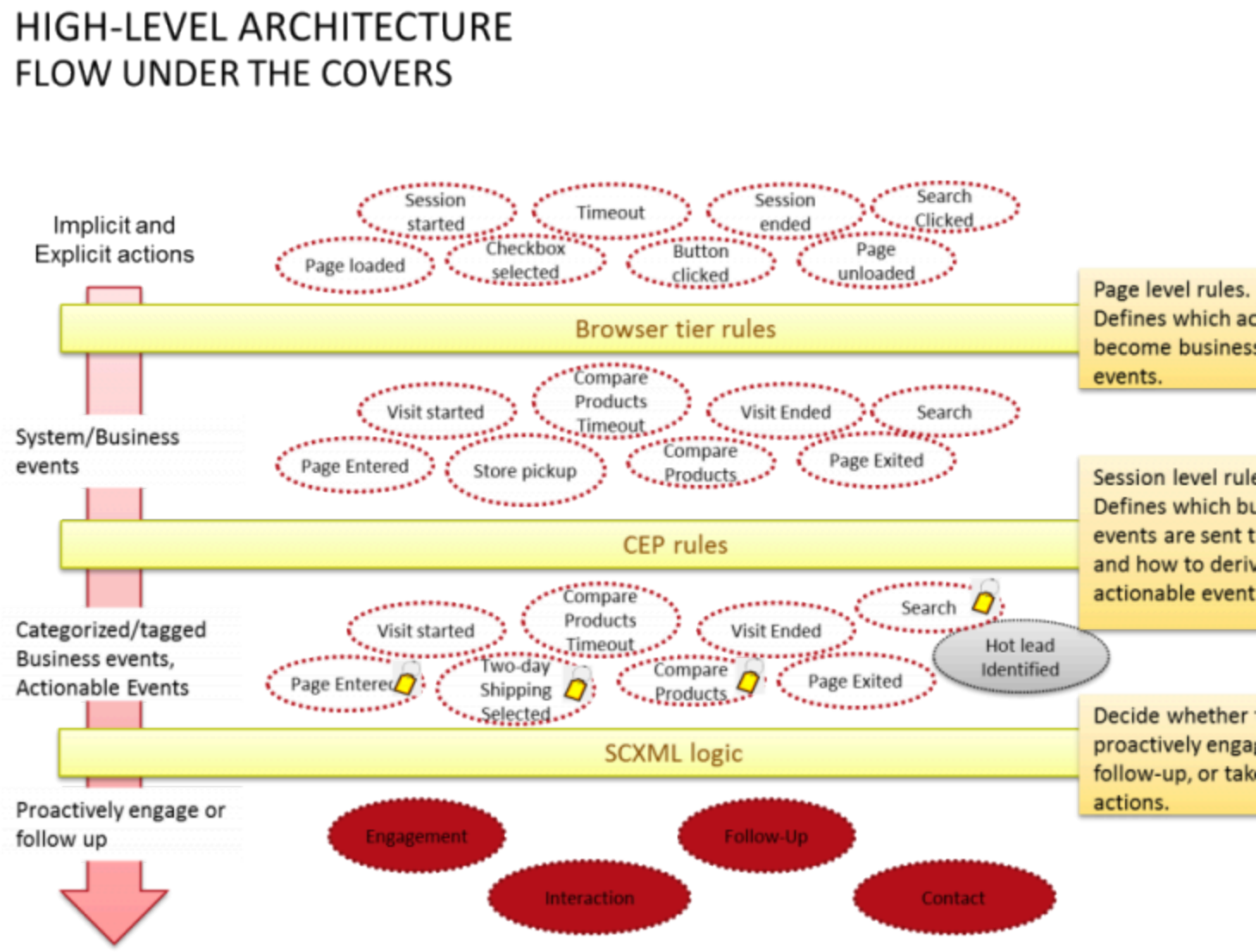
To enable web engagement on your website, you must implement one of the web engagement models available: simple or advanced. Genesys Web Engagement provides you with the Tools to develop and deploy your application specific to your website, as detailed in Application Development.

Event Workflow

	**Purpose:** To provide additional details about the event flow.

The Genesys Web Engagement Frontend Server receives system and business events from the browser's Monitoring Agent. This event flow is used to create actionable events which will generate requests to the Genesys Solution, and make possible the engagement, follow up, and additional actions with the Genesys Solution.

Event Flow Under the Covers



Event Flow Under the Covers

When the customer visits the website, the interactions with the webpages generate a flow of events, such as Session Started, Timeout, Button clicked, and so on. The browser's monitoring agent will submit only relevant events, according to the page rules, DSL, and category information. Additional Complex Event Processing Rules are used to manage the visit and the context information, and SCXML logic enable to proactively engage or follow-up or implement any other action.

Event Definition and Customizations

SYSTEM events are constants which cannot be customized. Two types of system events are available:

- **Visit-related** events, such as *VisitStarted* or *PageEntered*.
- **Identity-related** events, such as *SignIn*, *SignOut*, *UserInfo*.

See Visitor Identification for further details. **BUSINESS** events are additional custom events, that you can create by implementing Advanced Engagement:

- You can create and define them in the DSL loaded by the monitoring agents in the Browser, with the Monitoring DSL API: For details on their implementation, read Creating Business Events.
- You can submit them from your webpages, by using the Monitoring Javascript API.

Visitor Identification

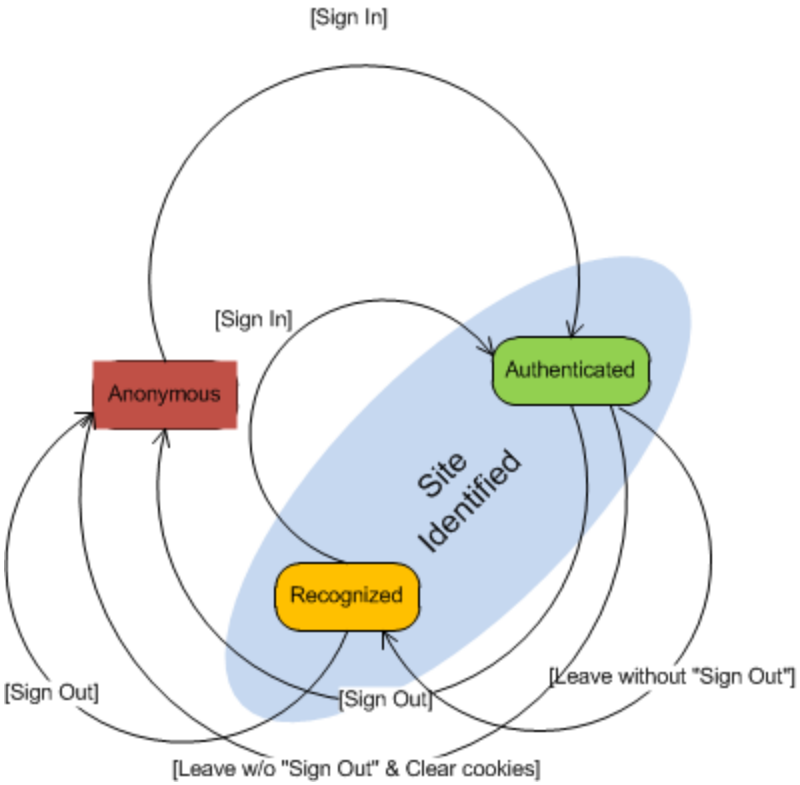
Overview

Genesys Web Engagement allows you to capture visitor activities on your website and to build a complete history of the visitor's interactions with your contact center.

When a visitor browses your website, the tracking code submits SYSTEM events to the Web Engagement servers that constitute a visit, such as `VisitStarted`, `PageEntered`, `SignIn`, `UserInfo`, and so on. The association or relationship between the visit and the visitor is based on the flow derived from SYSTEM events, in addition to the information retrieved from the Contact Server. In the end, you can access visit history through the Event Resource in the History REST API.

To associate the visitor with the visit, Genesys Web Engagement must "identify" the visitor as one of three possible states:

- `Authenticated`—The visitor logged in to the website with a username and password. The username can be an e-mail address, an account name or other similar identifier, depending on your website. When a user is authenticated, Genesys Web Engagement can maintain an association between the visitor and the visit.
- `Recognized`—The visitor closed the browser window and did not log out, but the website can submit user information with the sendUserInfo event, which contains the `userId`.
- `Anonymous`—The visitor is anonymous.



Visitor states

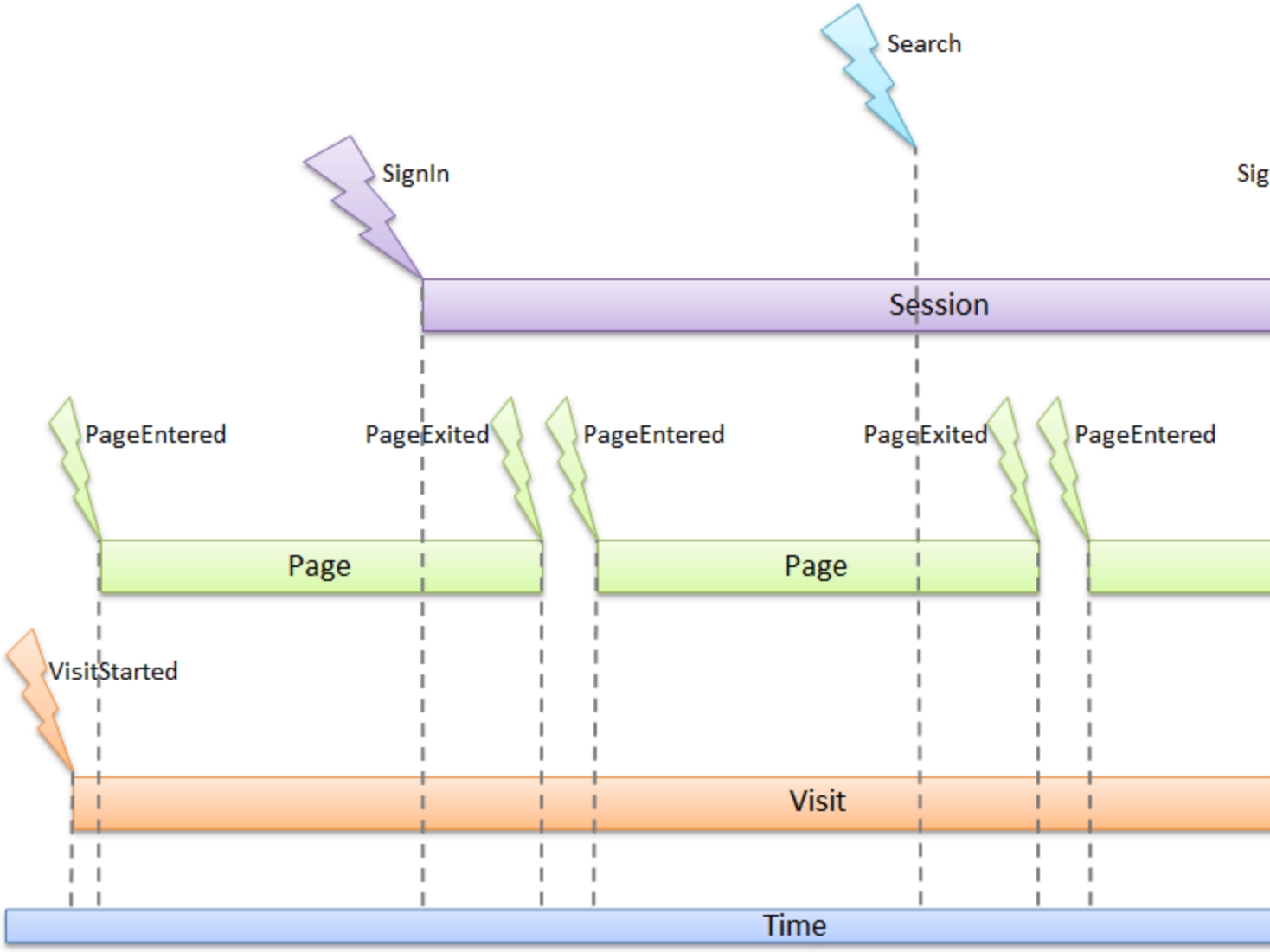
Genesys Web Engagement relies on your website to trigger the transitions between visitor states. You can do this by updating the tracking code with the following events in the Monitoring JavaScript API:

- sendSignIn—Send this event when the user is authenticated by the website. This allows the system to identify the user and creates a new "session" with a `sessionID` that is unique to a visit and will last the duration of the visit. Only Authenticated visitors have an associated `sessionId`.
- sendSignOut—Send this event when the user logs out of the website.

  Note: The `sessionId` lasts for the duration of the authenticated user's visit to your website. It is stored in a cookie and sent with every event that occurs between SignIn and SignOut, and is changed automatically after every SignIn event.
- sendUserInfo—Send this event when the user visits your website after closing the browser window on an authenticated session. For details, see Recognized Visitors.

Visitor Event Timeline

The figure below shows the timeline for events that take place when a visitor browses your website.



Visitor Event Timeline

All visitors to you website are identified with a `visitorId`, which can be used to associate the visitor to events, such as `PageEntered` or `PageExited`, during the span of the visit.

Accessing Visitor Information

The History REST API is a RESTful interface that allows you to manage a collection of JSON objects using `POST` and `GET` HTTP requests. You can create or access visitor information in the Backend Server database at any point during the event timeline by using the History REST API resources:

- The visit resource contains information about a visit that is started when a visitor navigates to your website. This visit is associated with the flow of pages visited (entered and then exited) by the visitor.

The visit ends when the visitor closes the browser or leaves the website.

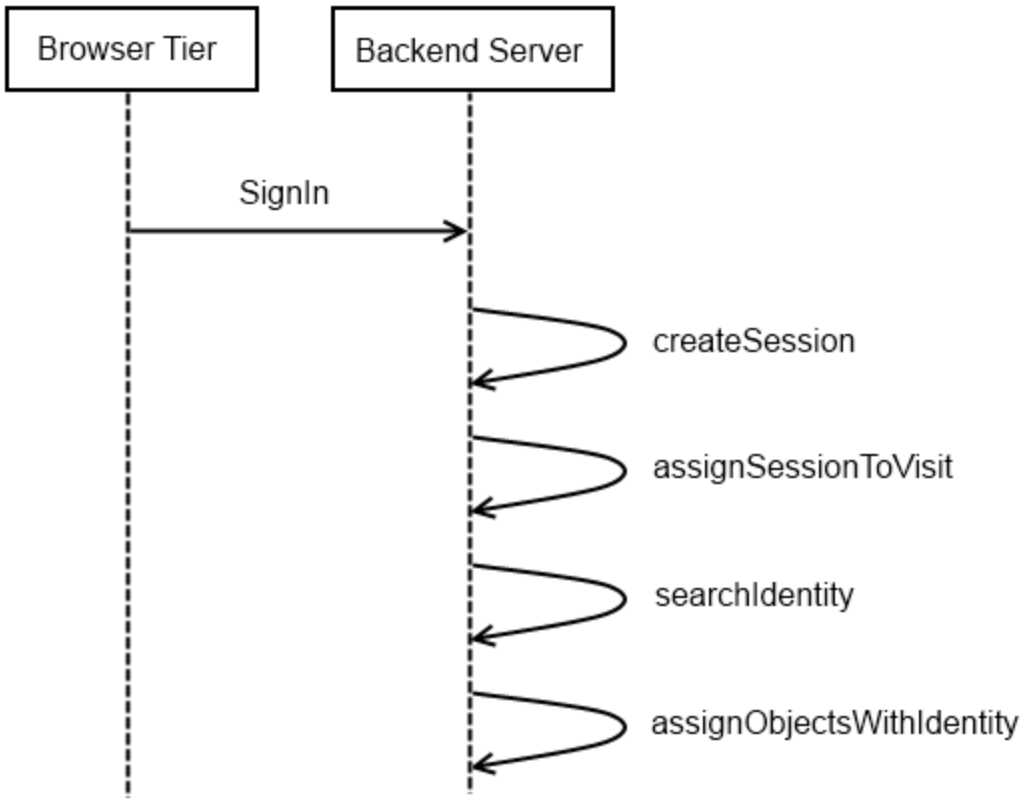
- The identity resource contains information for an authenticated visitor.
- The session resource contains information about the events and pages involved in the visitor's browsing activity during the time between SignIn and SignOut.
- The page resource contains information for a specific page entered at a given time. If the visitor leaves the page and then later revisits the page, a new page resource is created.
- The event resource contains information about SYSTEM and BUSINESS events.

When a visitor begins a visit on the website (a browser session is opened for the website), the Backend Server creates a visit resource and records the browsing history related to the visit by creating a page resource for each page entered and then exited. These events are also recorded as a collection of events associated with the visit and the pages.

Authenticated Visitors

When the visitor is Authenticated on the website, you should use the `sendSignIn` event so that Genesys Web Engagement can start a new session. When the Backend Server receives this event, it creates a session resource and an identity resource to store the visitor information. The identifying information used to login (for instance, the e-mail address) is available in the `SignIn` event and is used to:

- Create the `identityId` or search the visitor's identity resource.
- Associate the visitor with a contact in the Genesys Solution.



GWE SignIn event.png

The `SignOut` and `UserInfo` events are used to manage the collections of additional information. These collections, called `contacts` and `useridentifications`, are introduced in the Backend Server database.

- The `contacts` collection maintains the association between the `identityIds` and the `sessionId`, `visitId`, and `globalVisitId`.
- The `useridentifications` collection store all of the SYSTEM events.

Recognized Visitors

When an Authenticated visitor closes the browser window without signing out and then later revisits your site, you can use the sendUserInfo event to tell Genesys Web Engagement that the visitor is now Recognized.

You will need to send the `userId` in the `sendUserInfo` event. How you track the `userId` depends on your website. For example, you could create a persistent cookie to store the `userId` when a visitor logs in to you website. Then when a visitor first browses your site, you could check the cookie and call the `sendUserInfo` event if the cookie contains the `userId`. There are many possible scenarios - the best implementation is entirely dependent on your website and its workflow.

Note: The visitor's identity cannot be guaranteed in the Recognized state. For instance, another

member of the visitor's family could be browsing the website with the same computer.

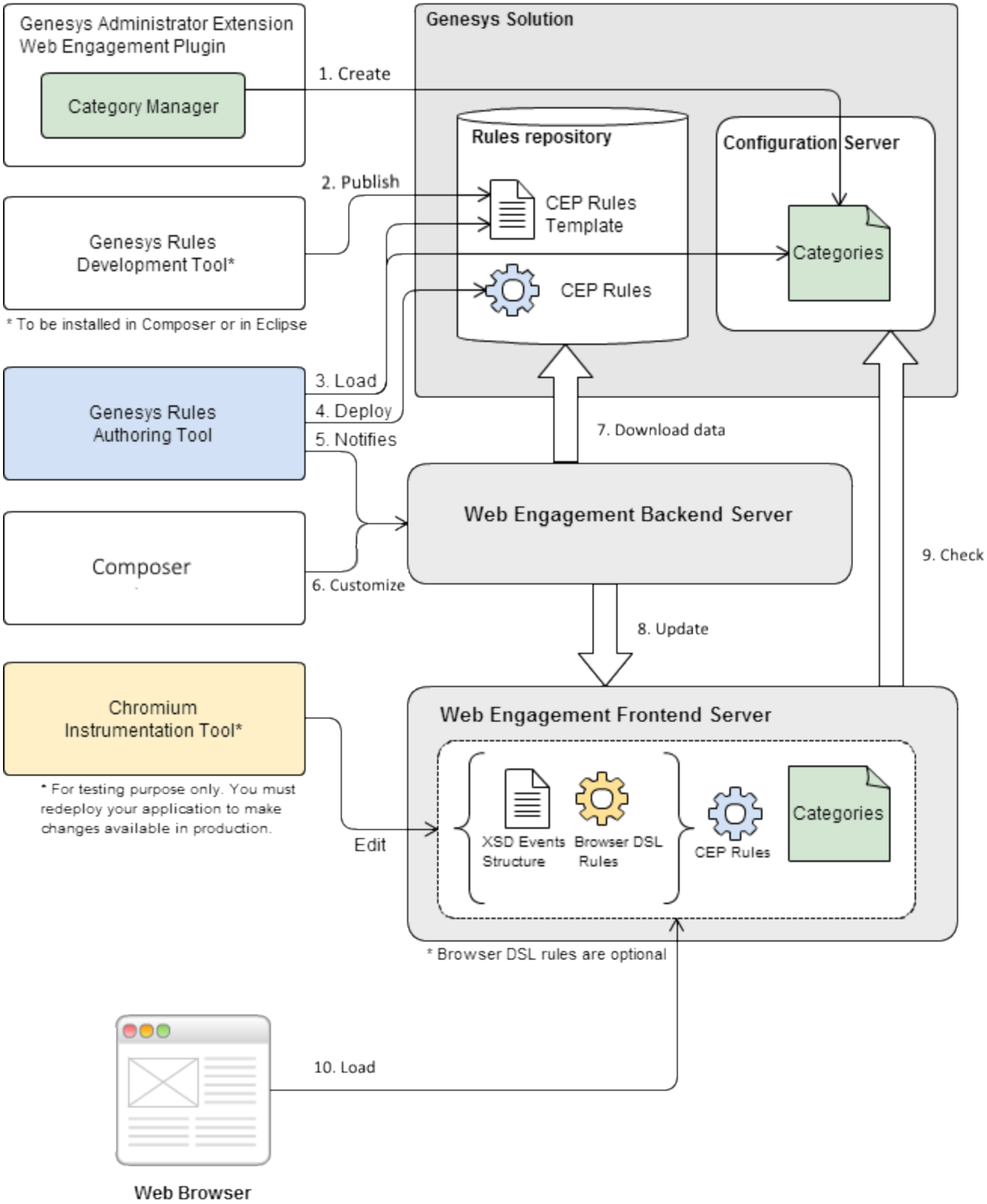
Anonymous Visitors

If the visitor is not Authenticated or Recognized, no identity is created. The visitor's activity on the website, including events and pages visited, are still associated with the visit.

Engagement Models

Implementation of the Models

When you develop a Web Engagement application, you start by creating your application with the script tools which generate default SCXML, rules template, and DSL code. All this material is available for customization through specific tools. The diagram below shows where you can customize the Web Engagement data used by your application.


Genesys Administrator Extension
Web Engagement Plugin
Category Manager
1. Create
Genesys Solution
Rules repository
Configuration Server
2. Publish
CEP Rules
Template
Categories
Genesys Rules
Development Tool*
CEP Rules
* To be installed in Composer or in Eclipse
3. Load
4. Deploy
5. Notifies
Genesys Rules
Authoring Tool
7. Download data
Web Engagement Backend Server
Composer
6. Customize
9. Check
8. Update
Chromium
Instrumentation Tool*
Web Engagement Frontend Server
* For testing purpose only. You must
redeploy your application to make
changes available in production.
Edit
XSD Events
Structure
Browser DSL
Rules
CEP Rules
Categories
* Browser DSL rules are optional
10. Load
Web Browser

Relationship between tools and application data.

1. You **create** categorization information with the **Category Manager** interface of the Genesys Administrator Extension. This information is added to the Configuration Server and retrieved by the Frontend Server. When the Frontend Server receives a browser request, it checks the category information.
2. You must **publish** the CEP rules template associated with your engagement model. You can modify this template before you publish it. Before you build, deploy, and start your application, you can also edit the SCXML files available in the **_composer-project** directory of your application folder.
3. The Genesys Rules Authoring Tool (GRAT) gets the CEP Rules template and allows you to create a package of CEP rules based on your **categories** (simple model) or on your **business events** (advanced model).
4. If the Web Engagement servers are built, deployed, and started, you can **deploy rules** with GRAT.
5. The GRAT notifies the Backend Server that rules are available in the Rules repository.
6. The Backend Server **downloads** the rules.
7. The Backend Server **updates** the Frontend Server.
8. When a browser submits a request to the Frontend Server, the Frontend Server checks the categories before providing the monitoring data.

Simple Engagement Model

Simple Engagement is deriving categories from the content of the SYSTEM and BUSINESS events. With this model, you do not need to create business events. You create rules and category information based on the available out-of-box system events. See Simple Engagement for further details.

Advanced Engagement Model

Advanced Engagement model uses Business events defined in the Browser-Tier DSL to create event-related rules. Once the business event gets generated by the Browser-Tier DSL, all the event attributes are available for the complex event processing and orchestration. See Advanced Engagement for further details.

Models combination

You can combine both the category and business event-based models to extract the most value out of the solution.

Simple Engagement

Simple Engagement is a simple solution to add Web Engagement to your website with limited effort. Through the Plug-in for the Genesys Administration Extension, you can define, in a few clicks, web engagement categories that contain business information related to URL or web page titles. Then, to implement web engagement, you can use or customize the category-based rules template and define rules in Genesys Rules Authoring tool. These rules implement conditions based on category information to define when to submit an actionable event (which will start the engagement process). For instance, you can define a category for a set of pages, identified by a category name, which contains tags associated with titles or parts of the URL. For instance, it is possible to define a Product category associated with the `http://www.genesyslab.com/products/index.aspx` page and several or all product subpages, such as `http://www.genesyslab.com/products/genesys-voice-platform/overview.aspx` or `http://www.genesyslab.com/products/proactive-contact/overview.aspx`.

- To associate the category with all the pages containing the "products" string in the URL, you can create the "products" tag which defines the "products" string as the plain text expression to search in the events triggered by the customer browsers.
- To set up a specific list of subpages for the Products category, you can create a tag for each of them:
  - The "genesys-voice-platform" tag which defines the "genesys-voice-platform" string as the plain text expression to search in the events triggered by the customer browsers.
  - The "proactive-contact" tag which defines the "genesys-voice-platform" string as the plain text expression to search in the events triggered by the customer browsers.

Categories and tags define the following parameters:

Element	Attributes	Description
category	name - category name (used for event categorization).	Defines the category processing information. Each category owns a collection of tags which contain the information related to URLs and titles

tag	• name - tag display name
	• expr - expression to search.
	• type - type of the expression to search:
	  • Regular Expression - regular expression search
	  • Plain Text - substring search. Default value.
	  • Google Like Expression - converts an expression using Google search operators to a regular
		Categorization tag. The tag contains specific business information (strings or regular expression) to search in URL and titles.

Element	Attributes	Description
	expression. • case-sensitive - true to make the expression case sensitive; false by default; • locale - localisation string; enables you to make the expression to search specific to the localization of the Browser.	

For guidelines to create categories with Genesys Administrator, see:

- Creating Categories
- Using the Plug-in for the Genesys Administration Extension

Default Templates for Category-Based Rules

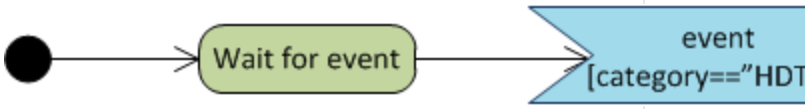
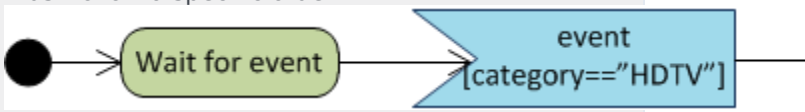
The templates for Category-Based Rules define how to process events sent from the Web Engagement Frontend Server. They define both the type of events to take into account and the action to perform. The Genesys Rules Authoring Tool loads the template and uses its content to help you define rules. These templates can be modified with the Genesys Rules Development Plug-in (in Composer or in Eclipse). The following table defines the possible types of rules available in the default WebEngagement_CEPRule_Template:

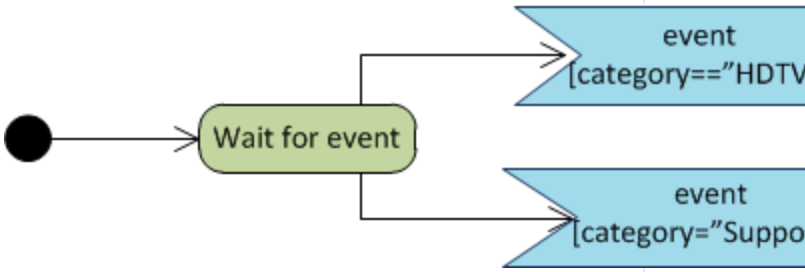
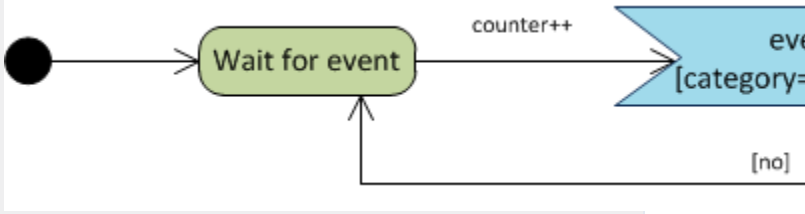
- The `Singleton` template allows a single event as input for the generation of the actionable event.
- The `Sequence` template enables a sequence of incoming events following a predefined order to generate the actionable event.
- The `Set` template allows an unsorted set of incoming events.
- The `Counter` template counts the events matching the same category.
- The `Search` template submits an event when the search box is hit.
- The `Timeout` template submits an event when the timeout event occurs.

To implement these templates, you must create rules and select specific sequence of conditions, as detailed in the table below. Note that the rules implementation is done through the Genesys Web Authoring Tool.

Template Implementation

Singleton	
Description	The template receives each single event as a formal parameter. If the event's value matches the right category, then the actionable event is sent to the Web Engagement Backend Server.

	 GWM single.png
Expression Example	**When** page transition event occurs that belongs to category $category **Then** generate actionable event
Sequence	
Description	This template analyses the event stream received from the categorization engine and builds the sequence of events by category values. As soon as the event sequence is completed, the actionable event is submitted. Note that the event sequence must follow a specific order.  Click to enlarge.
Expression Example	**When** page transition event occurs that belongs to category `$category1` save as `$event1` **and** event following `$event1` with category `$category2` save as `$event2` (...) **and** event following $\$event^{n-1}$ with category $\$category^{n}$ save as $\$event^{n}$ **Then** generate actionable event based on $\$event^{n}$
Set	
Description	This template analyses the event stream received from the categorization engine and collects the events by category values. As soon as the event set is completed, the actionable event is submitted. If you use this template, the event order is not taken into account.

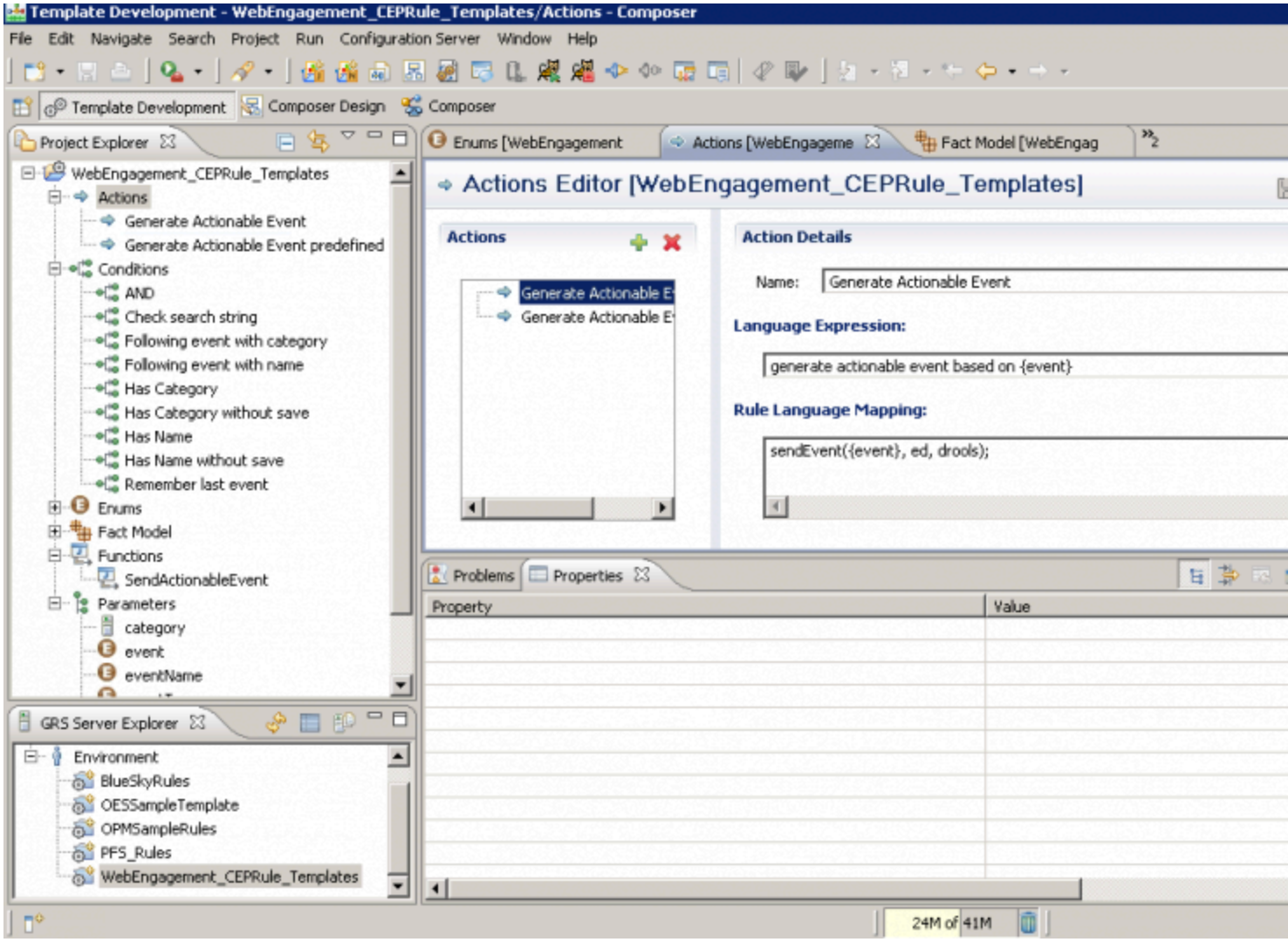
	 GWM Set.png
Expressions	**When** page transition event occurs that belongs to category `$category1` **or** page transition event occurs that belongs to category `$category1` (...) **or** page transition event occurs that belongs to category `$category`^{n} **Then** generate actionable event
Counter	
Description	This template analyses the event stream received from the categorization engine and counts events which occur for a given category. As soon as the counter is reached, the actionable actionable event is submitted.  GWM Counter.png
Expressions	**When** Category `$category` counts `$count` times **Then** generate actionable event
Search	
Description	The actionable event is submitted if a `Search` event occurs.

Expressions	**When** event with name `Search` save as `$event1` **Then** generate actionable event based on `$event1`
Timeout	
Description	The actionable event is submitted if a `Timeout` event occurs.
Expressions	**When** event with name `Timeout` save as `$event1` **Then** generate actionable event based on `$event1`

Details about the CEP Rules Template

The CEP Rules Template is created with your application at the following default location: You can edit the CEP Rules Templates in Composer or Eclipse with the Genesys Rules Development Tool; see:

- Configuring Genesys Rules Development Tool
- Publish the CEP Rules Template.



CEP Rules Template in Composer

The CEP Rules Templates in Composer.

The content of this project defines the actions, conditions, functions, and enumerations, any information that will be part of the rules that you will create upon this template with the Genesys Authoring Tool, as detailed in Create a Rules Package. Note that if you customize your rules templates, you must republish them. See Publish the CEP Rules Template.

Actions

The list of actions available in the template is listed in `WebEngagement_CEPRule_Templates > Actions`. You can edit, add, or remove these actions. In the Genesys Rules Authoring Tool, when you create a rule based on the template, you can add an action by clicking on the `Add action` button; the GRA Tool will display all the actions defined in the template. See Create a Rules Package for an example of implementation. The default action lists is the following:

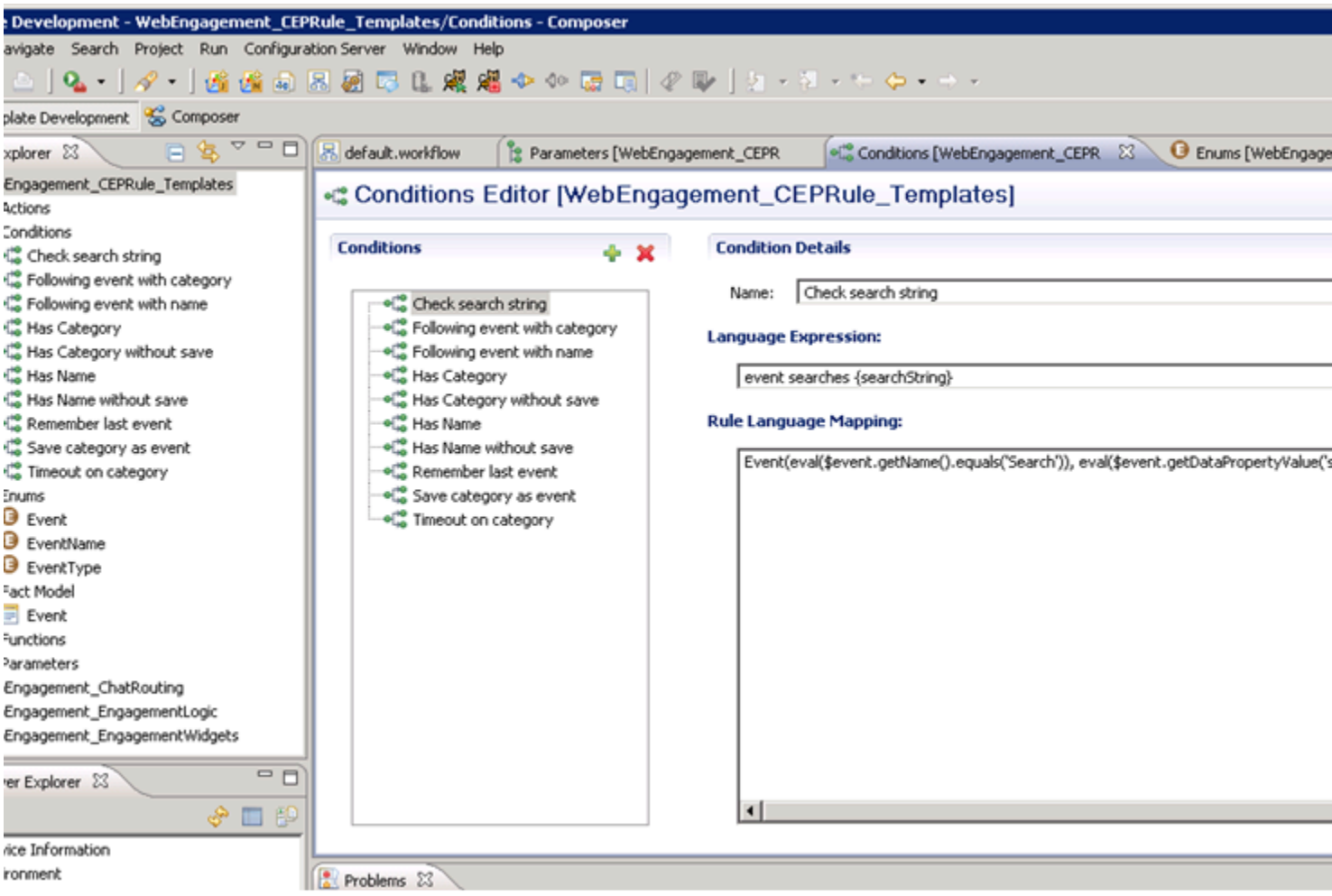
- Generate Actionable Event
- Generate Actionable Event Predefined

Enums

The enumerations available in the template is listed in `WebEngagement_CEPRule_Templates > Enums`. You can edit, add, or remove these enumerations. When you create a rule based on the template, you can specify a Phase by clicking on the `Add Linear Rule` button in the GRA Tool. The GRA Tool will display all the enumerates available in the template. In the default template, no specific enumeration is available. See Create a Rules Package for an example of implementation.

Conditions

The conditions are listed in `WebEngagement_CEPRule_Templates > Conditions`.



List of conditions in the CEP Rules Template.

You can edit, add, or remove these conditions. Each condition associates a name with an expression.

When you create a rule based on the template, you can add one or more condition to this rule by clicking on the `Add condition` button in the GRA Tool. The GRA Tool will display all the condition expressions available in the template. Note that for complex templates, you need several conditions to implement a rule. See Create a Rules Package for an example of implementation.

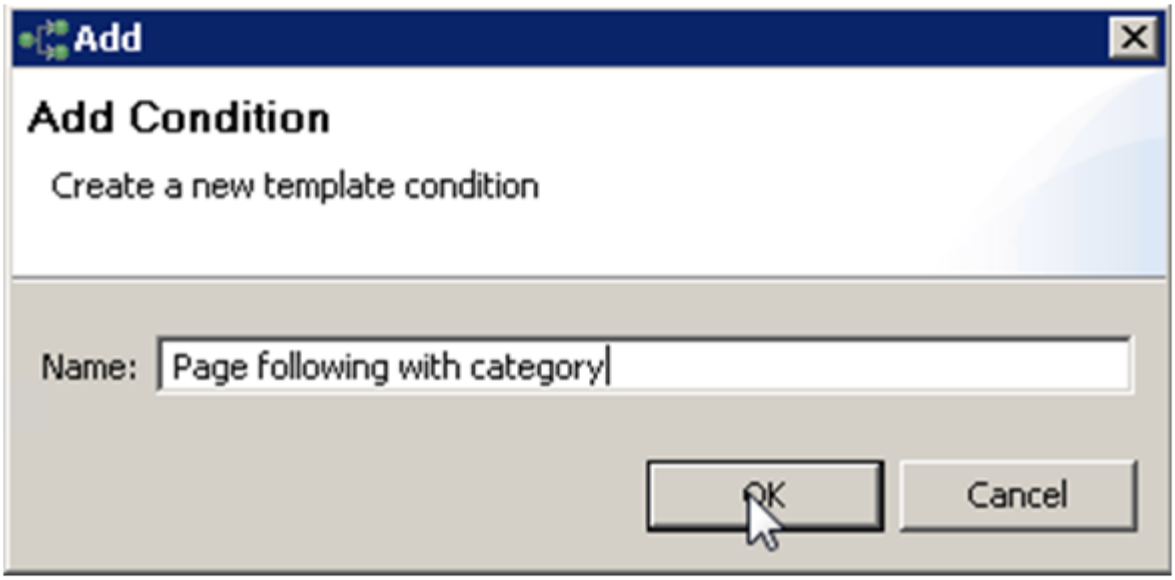
Condition Details

Condition Name	Expression	Condition details
Check search string	event searches {searchString}	Returns true if the event Search occurs and if the {searchString} label is found, this event's result is saved in the {event} label.
Following event with category	AND event following {prevEvent} with category {category} save as {event}	If the event follows {prevEvent} and contains the {category} label, this event's result is saved in the {event} label.
Following event with name	AND event following {prevEvent} with name {eventName} save as {event}	If the {eventName} follows {prevEvent} in parameter, this event's result is saved in the {event} label.
Has Category	page transition event occurs that belongs to category {category} save as {event}	If the event is a page transition for the given category, this event's result is saved in the {event} label.
Has Category without save	page transition event occurs that belongs to category {category}	Returns true if the event is a transition to the given category's page.
Has Name	event with name {eventName} save as {event}	If the {eventName} occurs, this event's result is saved in the {event} label.
Has Name without save	AND event with name {eventName}	Returns true if {eventName} occurs.
Remember last event	Precondition: save last event	Saves the last event.
Save category as event	category is {category} save as {event}	If the event contains the given category, this event's result is saved in the {event} label.
Timeout on category	Timeout event occurs with category {category}	Returns true if the Timeout event occurs for the given category.

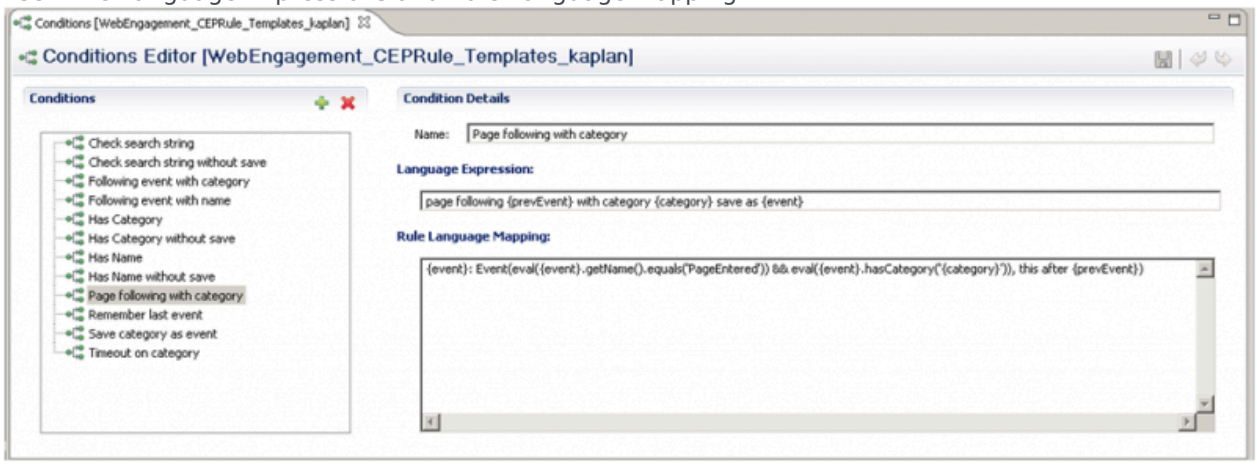
Customization Sample: Creating a New Condition Template

Purpose: Create a new condition for a page following a given category name. **Start**

1. Open the CEP Rules Template project with the Genesys Rules Development Tool (in Composer or in Eclipse) and navigate to the Conditions item.
2. Expand Conditions to open the Conditions editor.
3. In the Conditions tab, click on the + button. The Add Condition dialog box opens.



4. Enter a name and click OK. The condition is added and selected in the condition list; the condition detail panel opens.
5. Insert the Language Expressions and Rule Language Mapping:



6. Click on the Save button.
7. Publish the CEP Rules Template in the Rules Repository.

Stop

The rule is now available in the Genesys Rules Authoring Tool:


Genesys
GENESYS RULES AUTHORING

Advanced Engagement

Advanced Engagement enables customization based on business events. You can create new business events by writing new DSL code in the DSL files of your application, as detailed below. In current versions, only `Timeout` and `Search` events are available. To customize this model, you must first define your own events using the DSL loaded in the Browser-Tier Agents. Then, you can use the Advanced Rules Templates to create rules based on these events.

Creating Business Events

Customizing the DSL Files

When you create a new application with the tools script as detailed here, a default set of DSL files is created and used by your application. These files are defined in the `apps\<Your application name>\frontend\src\main\webapp\resources\dsl` directory. You can edit the **domain-model.xml** and add there a list of events, with specific conditions, related to your web pages' content. **Default domain-model.xml**

```
<?xml version="1.0" encoding="utf-8" ?>
<properties debug="false">
    <events>
        <event id="" name="">
            <!-- Add your code here -->
        </event>

        <event id="SearchEvent" name="Search">
                <trigger name="SearchTrigger" element="" action="click" url="" count="1" />
                <val name="searchString"                value="" />
        </event>
        <event id="TimeoutEvent10" name="Timeout-10" condition="">
                <trigger name="TimeoutTrigger" element="" action="timer:10000" type="timeout"
url="" count="1" />
        </event>
        <event id="TimeoutEvent30" name="Timeout-30" condition="">
                <trigger name="TimeoutTrigger" element="" action="timer:30000" type="timeout"
url="" count="1" />
        </event>

    </events>

</properties>
```

By using the `<event></event>` element, you can create as many BUSINESS event that you need. These events can be tied to the HTML components of your page. For instance, in the genesyslab sample, the **domain-model.xml** file declares the **SearchEventClick**, which is related to the `#top-search-input` textbox, the search box on top of all the genesyslab.com pages. **Source code of genesyslab.com**

```
<fieldset  id="top-search">
   <div class="search-bar clearfix">
      <input type="text" id="top-search-input" value="Search"></input>
```

```
      <a href="#" id="top-search-submit" class="sprite"></a>
      <input type="hidden" value="1" name="StartRow">
   </div>
//...
```

Event in the domain-model.xml file

```
<?xml version="1.0" encoding="utf-8" ?>
<properties debug="false">
<events>
//...
   <event id="SearchEventClick" name="Search">
      <trigger name="SearchTrigger" element="#top-search-submit" action="click" url=""
count="1" />
      <val name="searchString"               value="$('#top-search-input').val()" />
   </event>
//...
</events>
</properties>
```

You can create several events with the same name but different identifiers. This is useful to associate several HTML components with the same event if these HTML components have the save function. For instance, you can define several events associated to the Search feature and these events will all have the same name. For each event, you can define triggers which describe the condition to match to submit the event:

- Triggers enable to implement timeouts.
- Triggers can be associated with DOM events.
- You can define several triggers for the same event (see <trigger> for further details).

For further details about each element, see the Monitoring DSL API.
User's Guide Next Step: Publish the CEP Rules Template.

Implementing Events in your Webpages

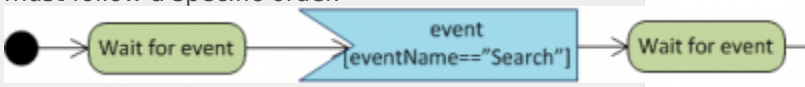
Another solution consists in using the Monitoring JavaScript API, which allows you to submit events and data from the HTML source code. In this case, you can use the _gt.push() method which allows you to decide when events should be submitted and which data they generate, directly from the webpages. See Monitoring JavaScript API Reference for further details.

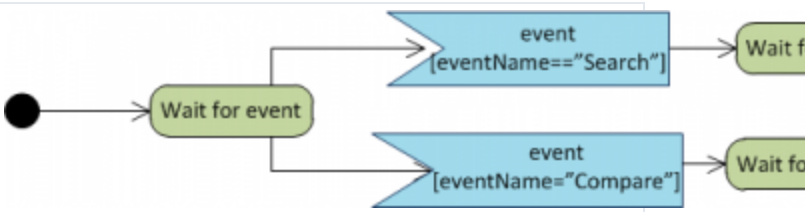
Default Event-Based Template

The templates below define conditions to fulfill in order to generate the actionable event:

- The `Singleton` template allows a single event identified by its name.
- The `Sequence` template defines a sequence of incoming events in a predefined order.
- The `Set` template specifies an unsorted set of incoming events.

All these templates associated with event names are available in the CEP Rules Templates. For further details, see Simple Engagement.

Singleton	
Description	The template receives each single event as a formal parameter. If the event's value matches the condition's event name, then the actionable event is sent to the Web Engagement Backend Server.  GWM single.png
Expression Example	**When** event with name $name **Then** generate actionable event
Sequence	
Description	This template analyses the event stream received from the categorization engine and builds the sequence of events by event names. As soon as the event sequence is completed, the actionable event is submitted. Note that the event sequence must follow a specific order.  Click to enlarge.
Expression Example	**When** event with name $name save as $event1 **and** event following `$event1` with name `$name2` save as `$event2` (...) **and** event following $event^{n-1} with name $name^{n} save as $event^{n} **Then** generate actionable event based on $event^{n}
Set	
Description	This template collects the events by event names. As soon as the event set is completed, the actionable event is submitted. If you use this template, the event order is not taken into account.

	 GWM Set.png
Expressions	**When** event with name $name1 save as $event1 **or** event with name $name2 save as $event2 (...) **or** event with name `$name`^{n} save as `$event`^{n} **Then** generate actionable event

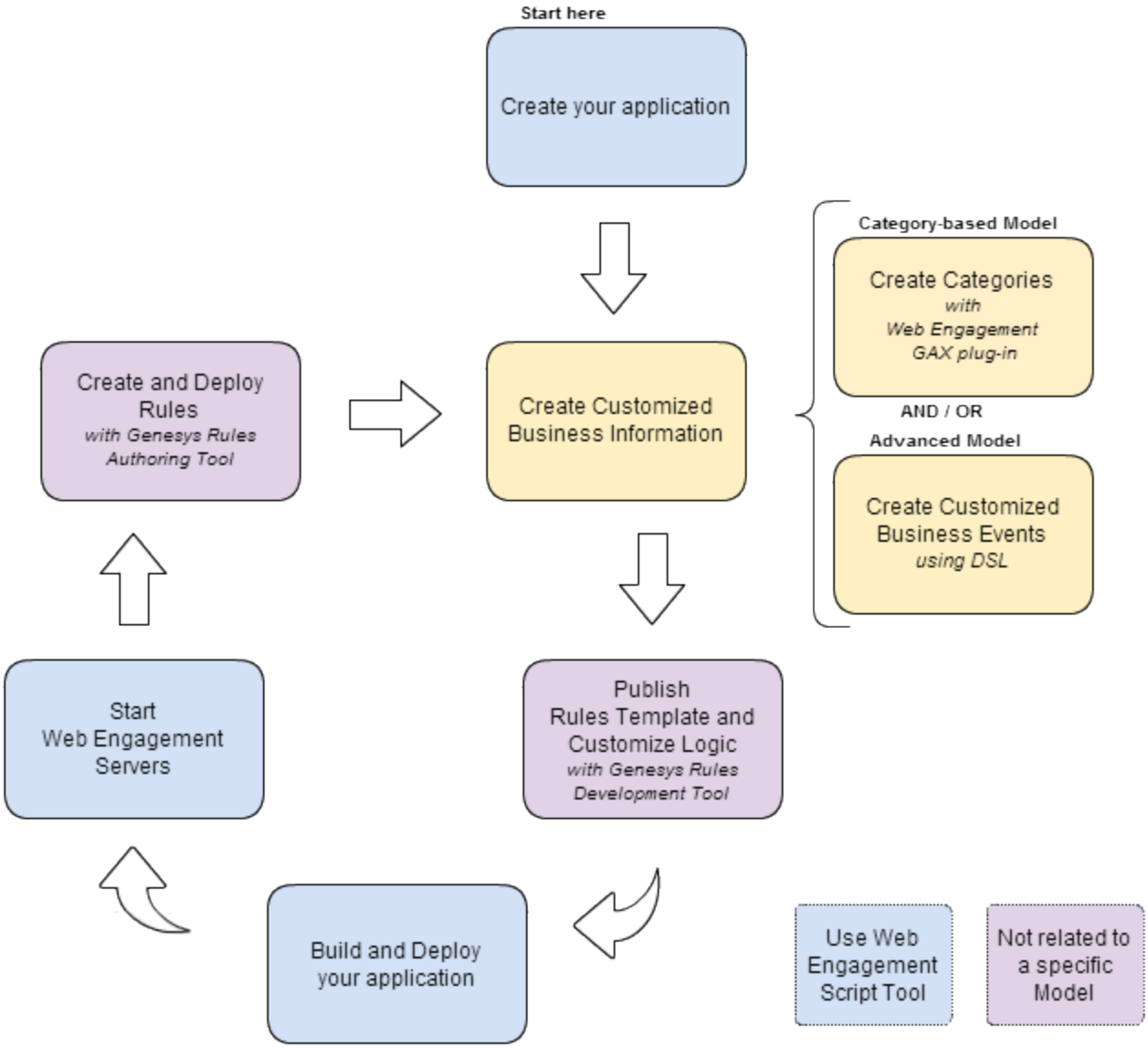
Application Development

Developing an application for Genesys Web Engagement is the process of defining all the components deployed through the Web Engagement Servers to implement Web Engagement features in your Genesys Contact Center, and to add Web Engagement to your website. When you create and configure your application, you create all the materials that will be used to generate the actionable events: customized business information, conditions, and engagement strategies. As a result of an actionable event, the Web Engagement servers will engage the customer with a chat or a web callback invite. Your application also contains the widgets for managing these invites, including a registration form submitted to anonymous customers who accept the invitation. The provided script tools create your application in the 'apps' folder where Web Engagement is installed. This creation includes all the default rules templates, logic (SCXML), events (DSL), in addition to web-specific data and engagement widgets. These materials must be compiled to generate the new Genesys Web Engagement Servers embedding your Genesys Web Engagement application. Then, you must deploy these servers to replace former versions of Genesys Web Engagement. You can develop two types of applications, tightly-coupled to engagement models:

- **Simple** application—Implement default web engagement capabilities (DSL scripts and rules), and provide customization through categories and rules.
- **Advanced** application—Implement business events (DSL) and use event-based capabilities to write implement web engagement to rules.

Application Development Workflow

The following diagram describes the development workflow for a Web Engagement application.



Application Development Lifecycle

1. **Create your application**
   - *Tool:* Web Engagement Scripts
   - *Description:* For each application you must use script tools to create and configure your customized Web Engagement application.
2. **Create Customized Business Information**
   Depending on the engagement model that you implement, you must define business information specific to your web pages that will be used to submit actionable events and web contexts to the Genesys Solution.
   - **Create categories** (Simple Model)

Tool: Web Engagement Plugin for Genesys Administrator Extension

Description: The categories contain business-related information to link your application with your web pages. They are used as parameters to set up conditions on events and generate actionable items.

Edition: You can modify category information at runtime. The Frontend Server checks category information when receiving web requests.

- **Create Business Events.** (Advanced Model)

  Tool: Text editor/Chromium Instrumentation Tool

  Description: You can create your own business events and DSL rules loaded in the monitoring agent. Then, these events are used to generate actionable items.

  Edition: To make DSL changes available for production, the Frontend Server must be rebuilt and restarted. You can also test the changes at runtime with Chromium Instrumentation Tool.

3. **Publish Rules Template and Customize Logic**
   Tool: Genesys Rules Development Tool

   Description: You must publish a Web Engagement Rules template before you can create rules. Optionally, you can customize the rules template, and you can also customize the SCXML files describing the Web Engagement logic before you deploy the Frontend server.

4. **Build and deploy your application**
   Tool: Web Engagement Scripts

   Description: If you create a new application or modify the SCXML, the DSL, or the logic of your application, you must build and deploy before you start the Web Engagement Servers.

5. **Start the Web Engagement Servers**
   Tool: Web Engagement Scripts.

   Description: To enable your application, you must start or restart the Web Engagement Servers.

6. **Create and Deploy Rules**
   Tool: Genesys Rules Authoring, version 8.1.300.xx

   Description: You must create rules to optimize the event flow and create complex conditions to generate actionable events sent to the Genesys Solution. These rules link with the categories containing the business information.

   Edition: You can deploy rules only if the Web Engagement servers are started.

Tasks Summary

This table summarizes the tasks that are required to create a Genesys Web Engagement application.

Task Objective	Related Procedures and Actions
Create a Genesys Web Engagement Application	• Create a New Application Project • Define the Application's Monitoring Domains

Task Objective	Related Procedures and Actions
Implement your Genesys Web Engagement Model	• Choose a Genesys Web Engagement Model • Define Business Information
Publish the CEP Rules Template	• Import the CEP Rules Template of your Application in Genesys Rules Development Tool • Configure the CEP Rules Template • Publish the the CEP Rules Template in the Rules Repository
Customize the SCXML Strategies	• Import the chat routing and engagement logic strategies
Build and Deploy your Genesys Web Engagement Application	• Build your Application • Deploy your Application
Start the Genesys Web Engagement Servers	• Configure the Web Engagement Channel • Configure the Registration Form • Start the Web Engagement Servers
GWE Create a Rules Package	• Create a Rules Package • Create Rules in the Rules Package • Publish the Rules Package

Create an Application

	**Purpose:** To describe the creation of an application.

As detailed in Develop your Application, you need to create an application to run Genesys Web Engagement. The current page details the application creation and first configuration steps.

Create a New Application Project

 **Purpose:** To run the `create` script to create your project structure. This script creates all the files required to run Genesys Web Engagement on your website.

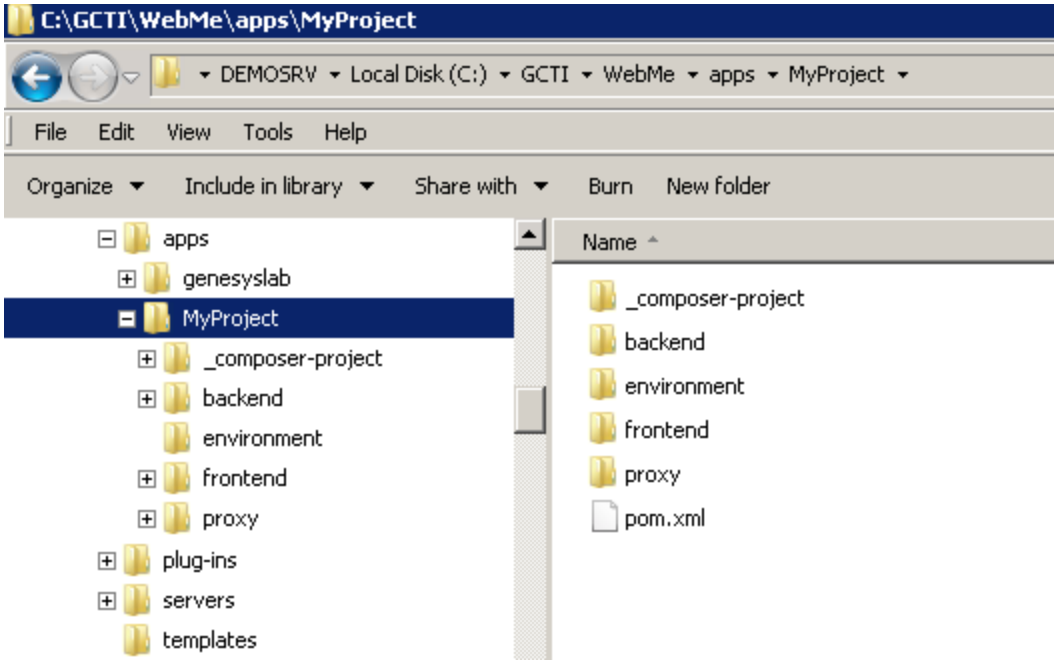
Start
Open the Windows Console of commands (cmd.exe) and type:

```
create <application name>
```

.

Stop

A folder named `<application name>` is created in the installation directory of Genesys Web Engagement.



The <application name> is MyProject.

This folder contains all the materials used to build and deploy your application:

- `_composer_project` contains all the SCXML default templates for the routing strategies and GRS rule template project. In addition, it contains the source code of the web widgets used for engagements.
- `backend` contains the Backend Server Application once the application is built.
- `environment` contains an environment property file.
- `frontend` contains the Frontend Server application once the application is built.
- `proxy` contains the proxy application used for testing purpose.

Next Steps

Define the Application's Monitoring Domains

Define the Application's Monitoring Domains

Purpose: To fill in the map.properties file, which contains all the information about the proxy and the monitoring domains of your application.
Start

1. Open the `\apps\<application name>\proxy\map.properties` file with a text editor.
2. Fill in the domain properties:
   - `gwmp.domainName` with the name of the domain—for instance, `genesyslab.com`.

- `gwmp.domainList` with the list of domains, separated with semicolons—for instance, `genesyslab.com;www.genesyslab.com;www-ssl.genesyslab.com`.

3. Fill in the frontend properties:
   - `frontend.server.host` with the name or IP address of your Frontend Server.

     Note: 127.0.0.1 or localhost is not allowed!
   - `frontend.server.http.port` with 8081.
   - `frontend.server.https.port` with 8443.
4. Save.

End

Next Steps

Create Customized Business Information

Create Business Information

	**Purpose:** To describe the steps for the implementation of your Genesys Web Engagement model.

Choose a Genesys Web Engagement Model

Genesys Web Engagement is delivered with predefined rules templates, DSL script, and routing logic that enable you to implement two web engagement models:

- Simple model: This model implements web engagement with minimal effort. In a few clicks, you can set up categories which define business information related to your web pages, for instance, a string to parse in the URL to generate an actionable event. Then, you create a set of rules based on categorized events, to process web engagement.
  Choose this model for a quick start with Genesys Web Engagement.
- Advanced model: This model requires the development of DSL scripts to customize at the business event level. Then, you create a set of rules which define conditions, based on events internal (business) information, to process web engagement.
  Choose this model if you need a customized event flow.
- For detailed information about the implementation of the Web Engagement models, read Genesys Web Engagement Models.
- For detailed information about the application development process, read Application Development.

Define Business Information

Once you have chosen your web engagement model, you must create your specific business information:

Model	Task	Additional details
Simple Model	Creating Categories with Genesys Administrator Extension	You can delay this step and create categories once the servers are started. At runtime, Genesys Web Engagement Frontend Server checks the categories each time it receives a request from a browser.
Advanced Model	Creating Business Events	You must write your DSL business events before you build and deploy, or you must restart the servers if you decide to extend your business events later.

Next Steps

Publish the CEP Rules Template

Create Categories

	Purpose: To create Categories with Genesys Administrator Extension. Each category contains business information based on URL and webpage titles, used in conditions to generate actionable events. For further information, read Simple Engagement.

Creating a Category

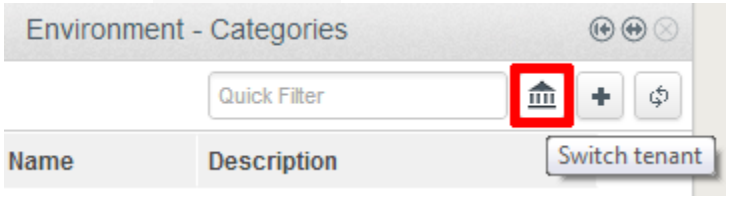
Purpose: To create categories to implement the simple engagement model on your website.

Prerequisites

- Genesys Administrator Extension, 8.1.301.02 or later, is installed;
- Web Engagement Plug-in for Genesys Administrator is installed.

Start

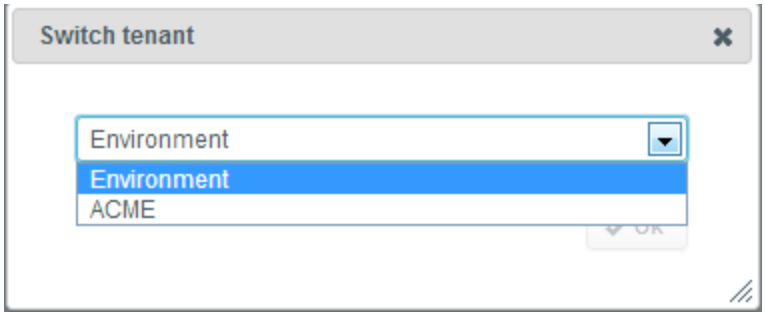
1. Open Genesys Administrator Extension and login.
2. Navigate to `CONFIGURATION > Categories`. The Categories interface opens.
3. Select the application's tenant.
   - Click on the `Switch Tenant` button:



Click the Switch tenant.

The Switch tenant dialog box opens.

   - In the drop-down list, select the Tenant where you deployed Genesys Web Engagement.



Select the application's tenant.

- Click OK.

4. In the Categories menubar, click + to add a new category. The New panel appears.
5. Enter a Category name—for example, Products;
6. (Optional) Enter a description;
7. Enable the Show category in Interaction Workspace option to display this category in the Interaction Workspace, if an agent opens interactions related to this category.
8. Click Save. The category appears in the Categories panel.

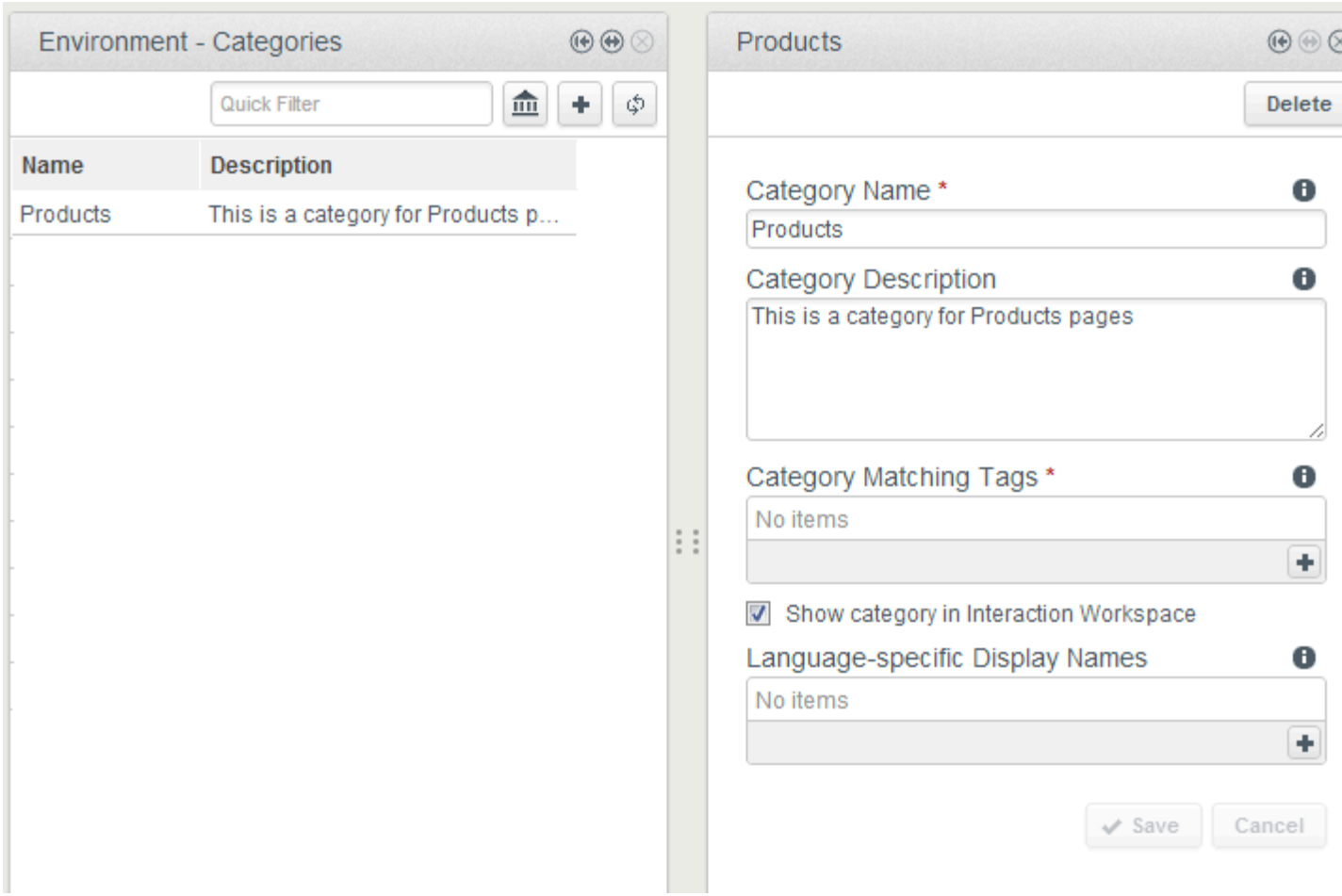
End

Create Category Matching Tags

Purpose: To add the business information which should raise events for a category. Each matching tag contains an expression to search in URLs and titles submitted with the events of the browser. For instance, a tag to identify the http://www.genesyslab.com/products/genesys-inbound-voice/overview.aspx page could be the plain expression 'genesys-inbound-voice' or the regular expression 'Inbound Voice'.

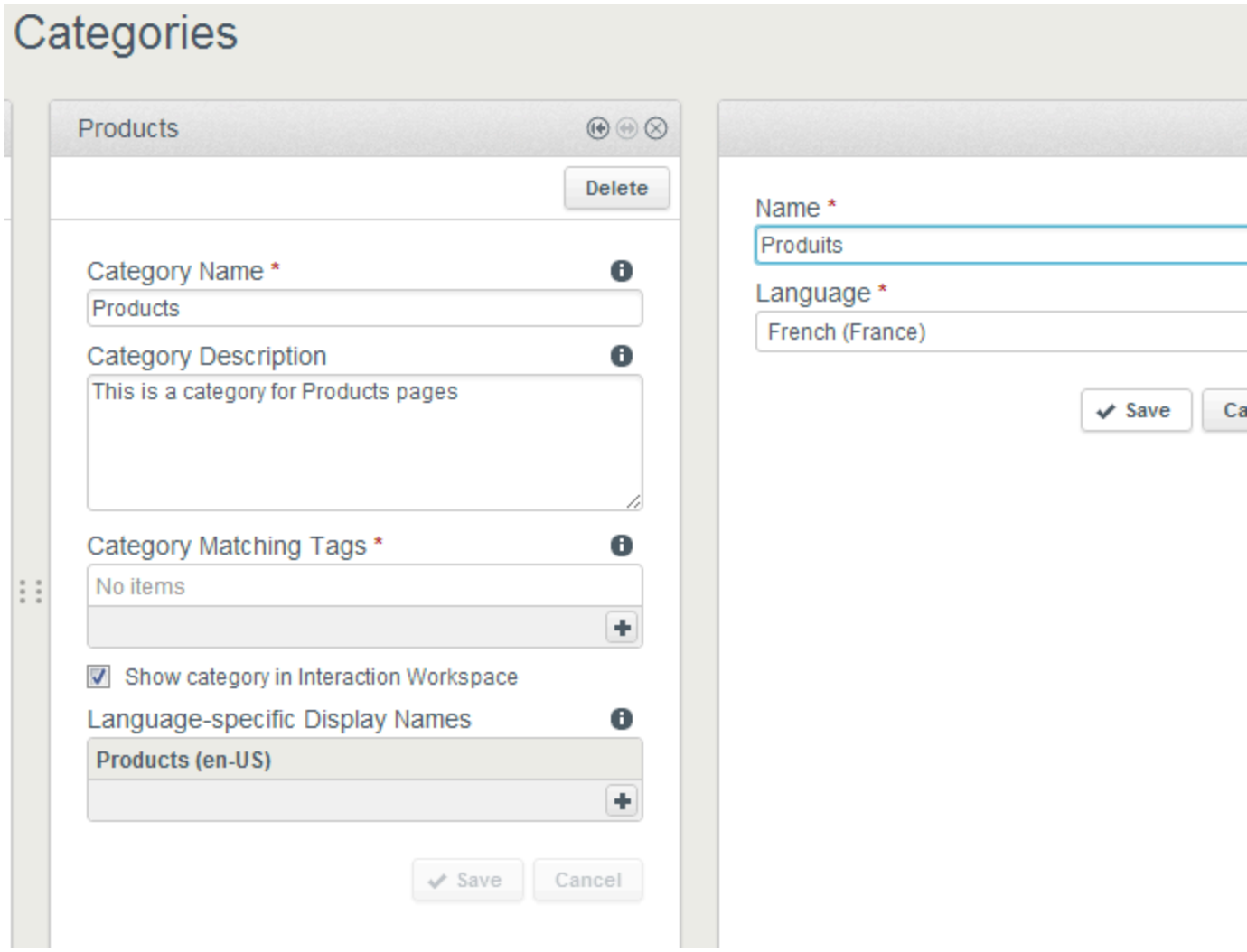
Start

1. In the Categories panel, select your category. The <category name> panel opens.



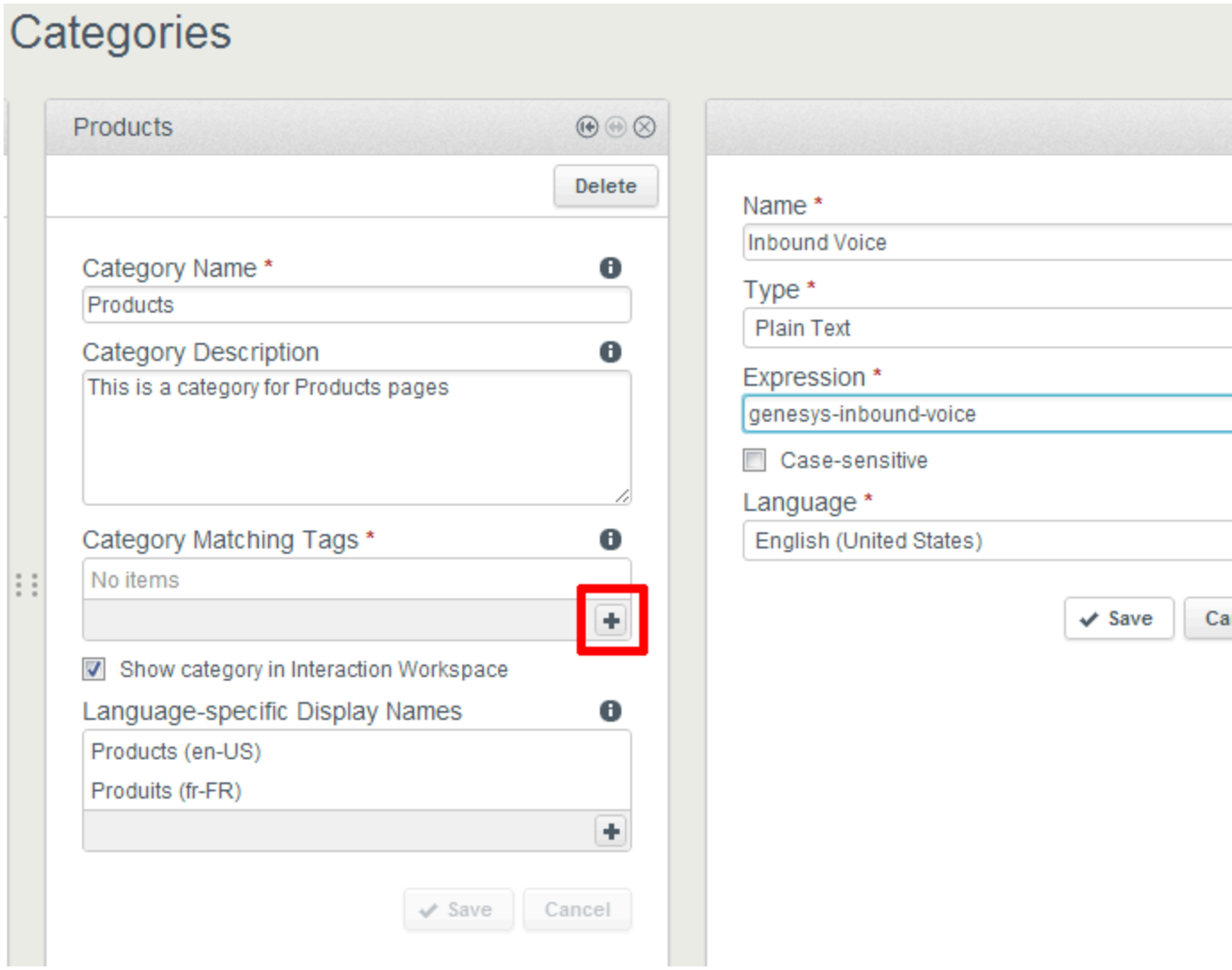
The selected category is opened and ready for customization.

2. In `Language-specific Display names`, click +. The `Name details` panel opens.
   - Enter a Name—for example, Products.
   - Select a Language—for example, en-US.
   - Click Save. Interaction Workspace and other Genesys Tools will display `Products` for english users.
   - Define additional `Display Names` if needed, then close the panel.



Adding French Display name.

3. In the `Category Matching Tags` section, click +. The `Tags details` panel opens.
4. Fill in the form to create a tag. For example, let's create a tag which refers to a specific product, such as Genesys Inbound Voice:
   - Enter a name—for example, Inbound Voice;
   - Select a type—for example, `Plain Text`;
   - Enter an expression according to the selected type; for a plain expression example, genesys-inbound-voice; for e regular expression example, `Inbound*Voice`
   - Optionally, enable `case-sensitive` to enable a case-sensitive search.
   - Select a locale language—for example, English (United States).



Create as many tags as needed.

- Click Save. The new tag is added to the Category Matching Tags list.

End For further details about this tool, see also:

- You can use regular expressions to create tags.
- You can create as many tags as you need to complete your category. Note that all events raised for the given tags will be associated with this specific category.
- You can create as many categories as needed.

Next steps

- Publish the CEP Rules Template

Publish the CEP Rules Template

	**Purpose:** To publish the CEP Rule templates which enable the rules creation.

Import the CEP Rule Templates of your Application in the Genesys Rules Development Tool

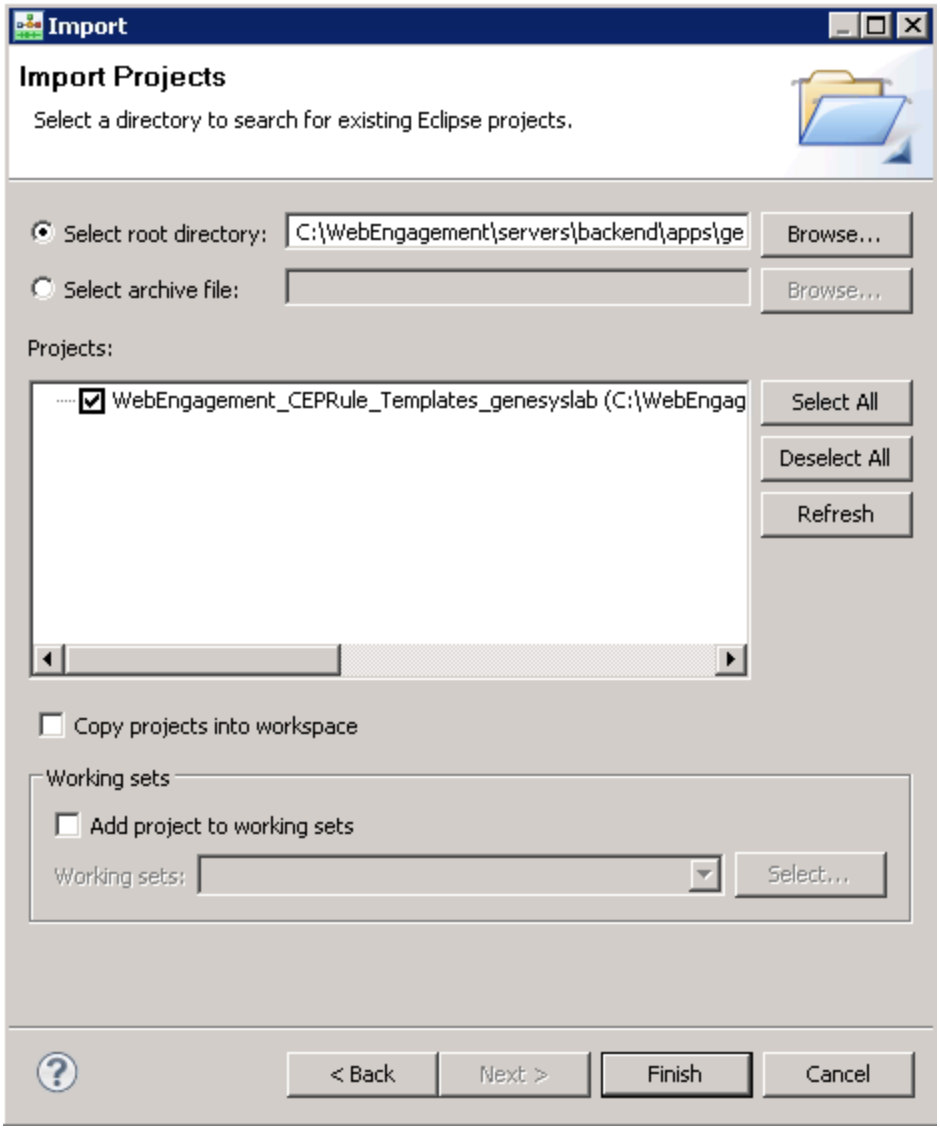
 **Purpose:** To import the template created by the create script in the Genesys Rules Development Tool. Even if you do not plan to customize this template, your rules template must be published in the Rules System Repository before you try to create rules.

Prerequisites

- The Genesys Rules Development Tool is installed, configured, and opened in Composer or in Eclipse.
- The following section uses Composer.

Start

1. Navigate to `Window > Open Perspective > Other > Template Development` to switch to the Template Development perspective of the Genesys Rules Authoring Developer Tool.
2. Select `File > Import...`.
3. In the `Import` dialog window, navigate to `General > Existing Projects into Workspace`. Click `Next`.
4. Select `Select Root Directory:`, then click `Browse`.
5. Import your project:
   - Browse the `\apps\<application name>` folder of the Genesys Web Engagement Installation directory and navigate to the `\_composer-project\WebEngagement\WebEngagement_CEPRule_Templates` subdirectory.
   - Click OK. The `WebEngagement_CEPRule_Templates_<application name>` is added to the `Projects` list view of the `Import` Dialog Window.
   - Select the `WebEngagement_CEPRule_Templates_<application name>` project.
   - Warning: Do **not** enable the option `Copy projects into workspace`.

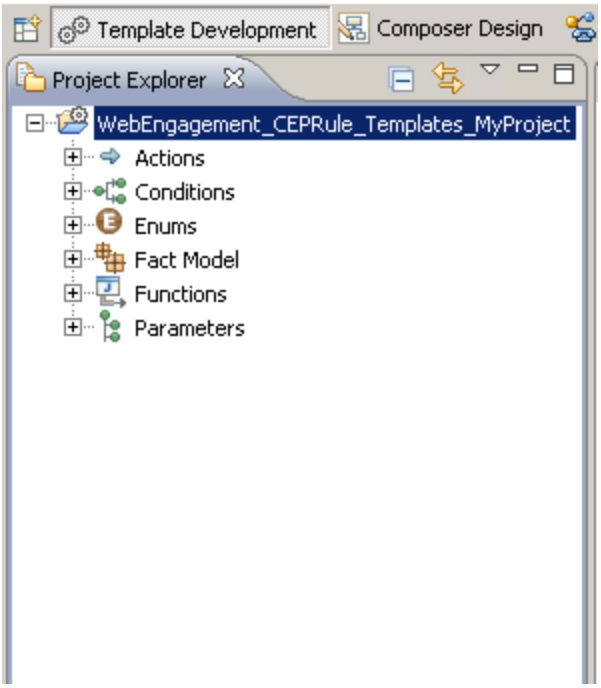


Import the default templates by clicking finish.

- Click `Finish` to import the project.

End

As a result, the `CEP_Rule_Templates_<application name>` is added to the `Project Explorer`.



The CEPRule Template MyProject is added to the Project explorer.

Next Steps

Configure the CEP Rule Templates

Configure the CEP Rule Templates

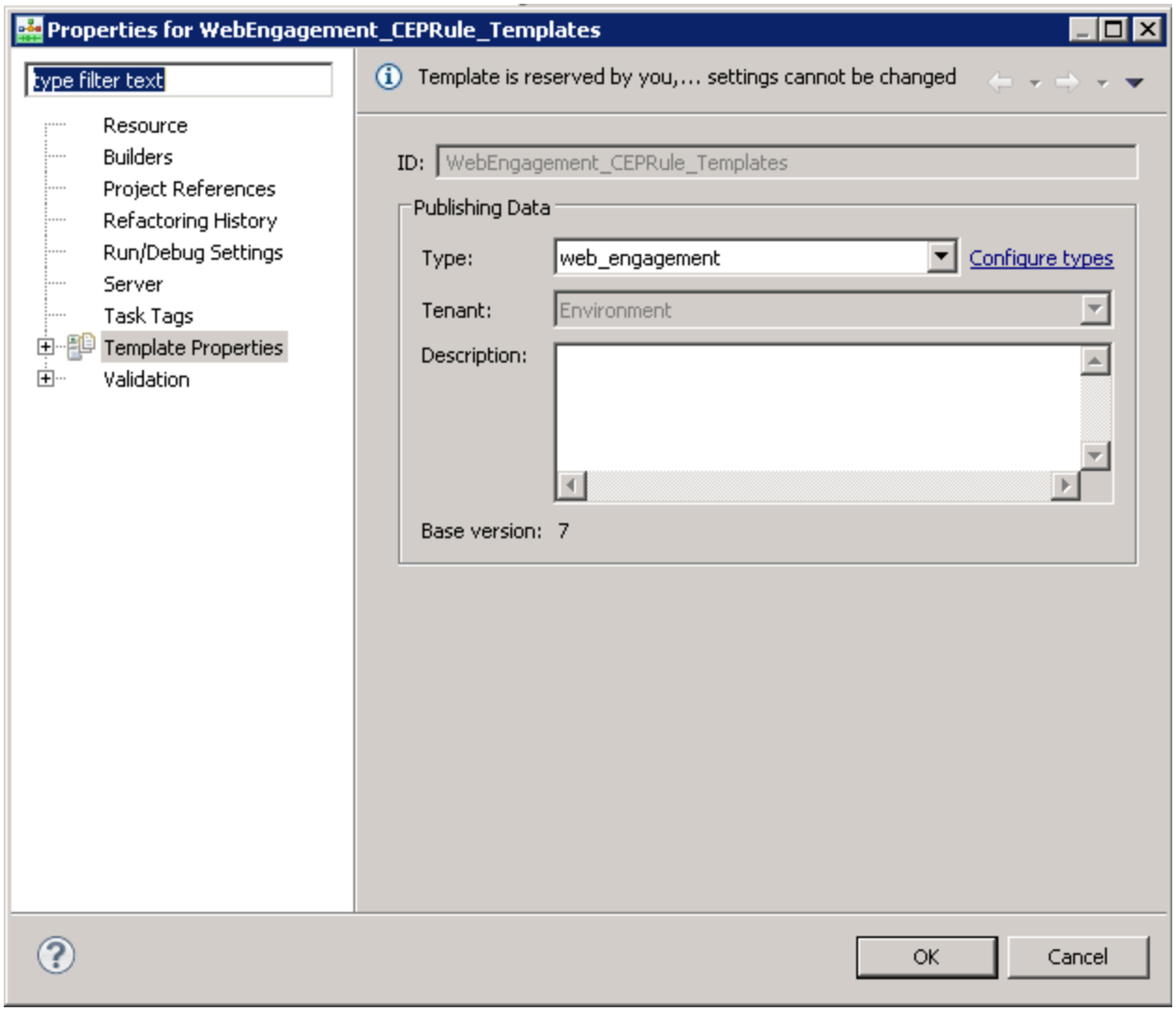
Purpose: To configure the project properties.

Prerequisites

- The `Web Engagement Categories` business attribute was previously defined in Genesys Administrator.

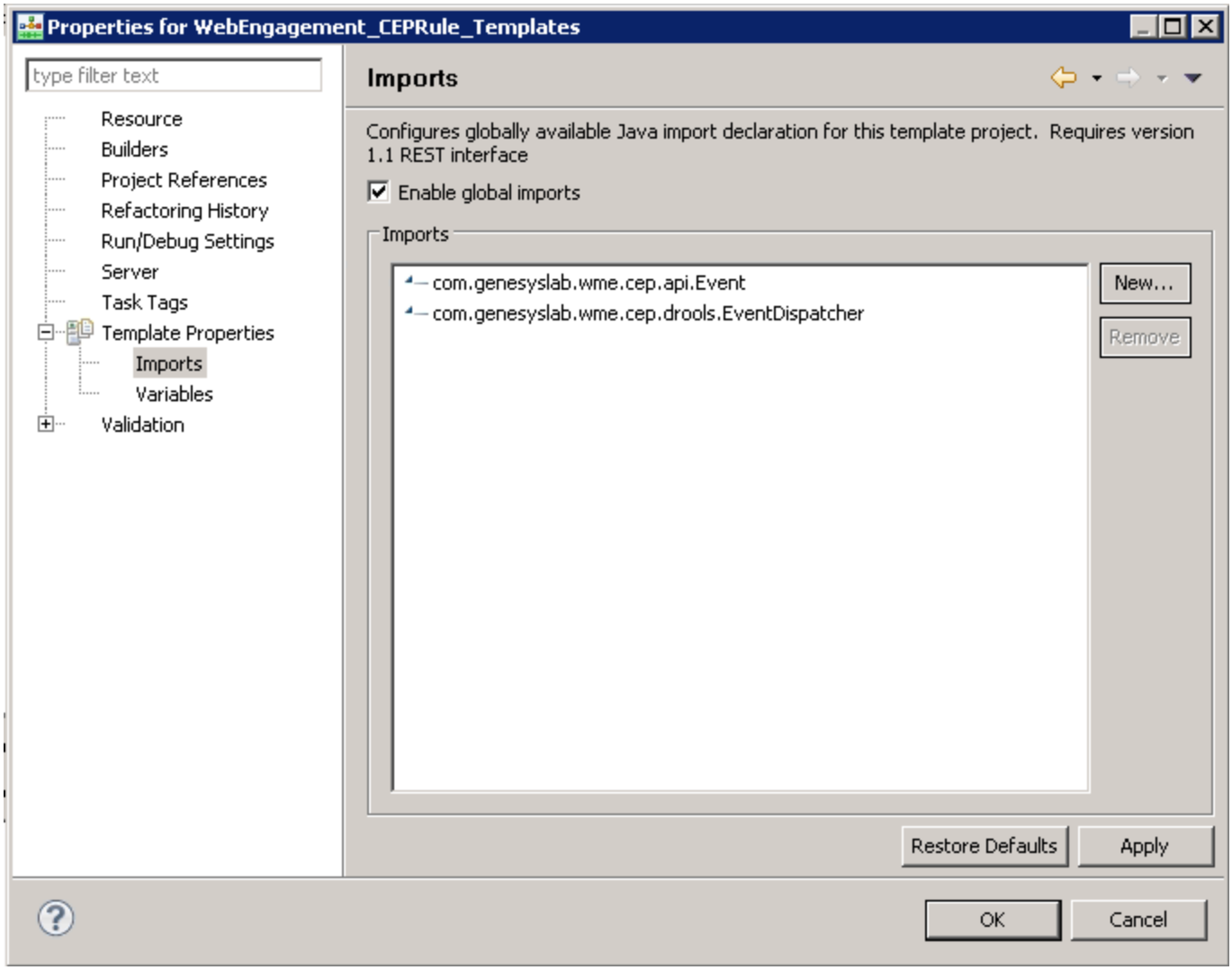
Start

1. In the `Project Explorer`, right-click on the `WebEngagement_CEPRule_Templates` project. Click on `Properties`.
2. In the `Properties` dialog window:
   - Navigate to `Template Properties`. In `Publishing Data`, set `Type` to `web_engagement`.



GWE-CEPRulesTemplateComposerProjectProperties.png

- Navigate to `Template Properties > Imports`. The `Imports` panel shows up.
- Select the option `Enable global imports`.

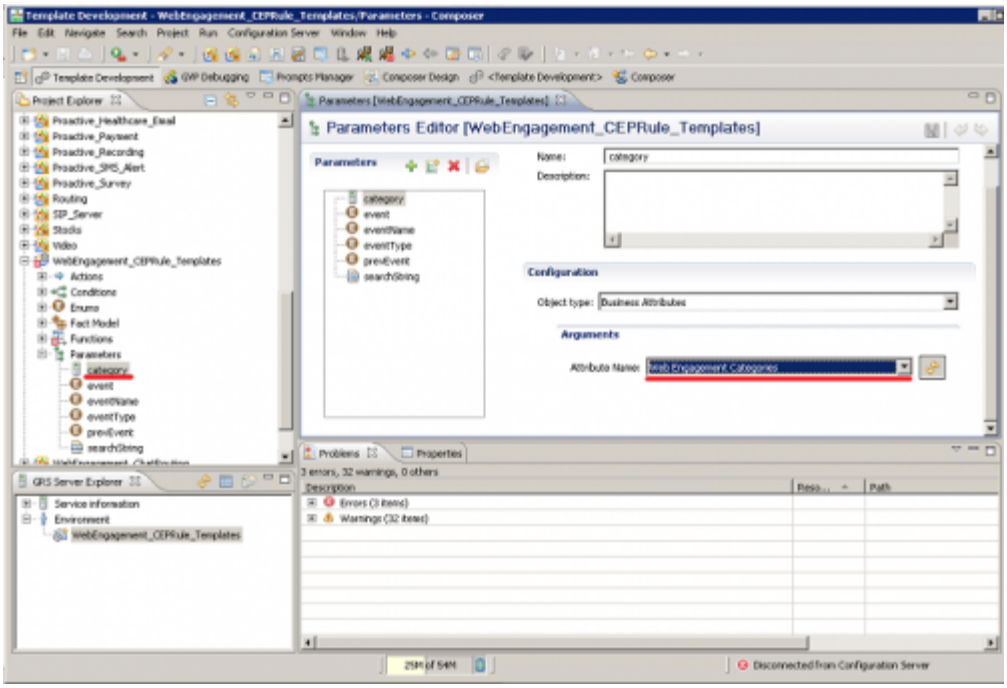


GWE-ImportsPropertiesCEPRules.PNG

Note: The `com.genesyslab.wme.cep.api.Event` and `com.genesyslab.wme.cep.drools.EventDispatcher` packages must be present.

- Click OK.

3. In the `Project Explorer`, navigate to `WebEngagement_CEPRule_Templates > Parameters > category`.
4. In the `Parameters Editor` Panel, set the `Web Engagement Categories` value for the `Attribute Name` option.



Set the Attribute Name for Web Engagement.

5. Save.

End

Next Steps

Publish the CEP Rule Templates in the Rules Repository

Publish the CEP Rule Templates in the Rules Repository

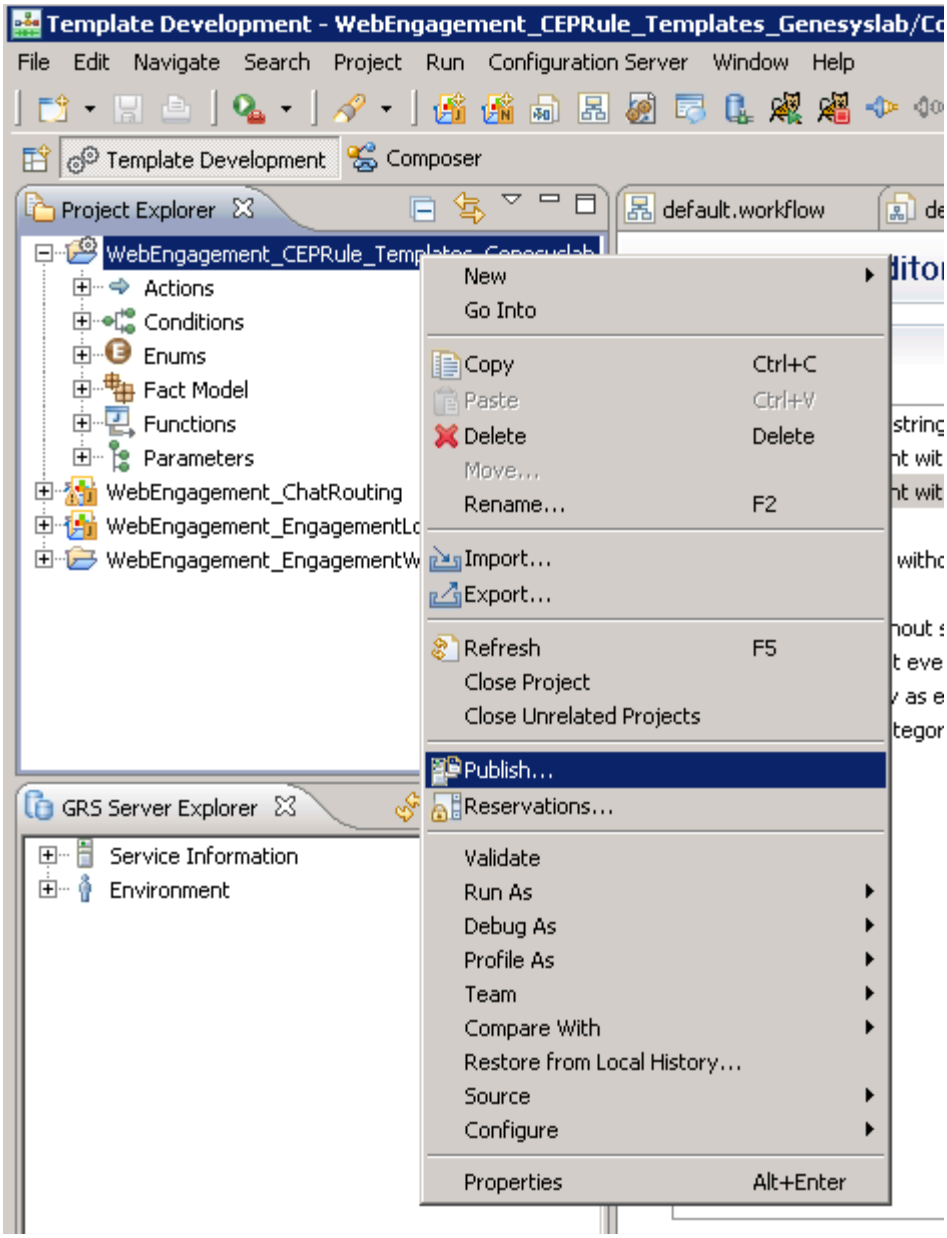
Purpose: To publish the template in the repository to enable the rules creation in Genesys Rules Authoring Tool.

Prerequisites

- Your user owns rights to manage rules in Genesys Rules Authoring Tool, as detailed in Installing the GRDT Component in the Genesys Rules System Deployment Guide.
- You configured the Genesys Rules Development Tool to enable connection to the Configuration Server and Rules Repository Server.

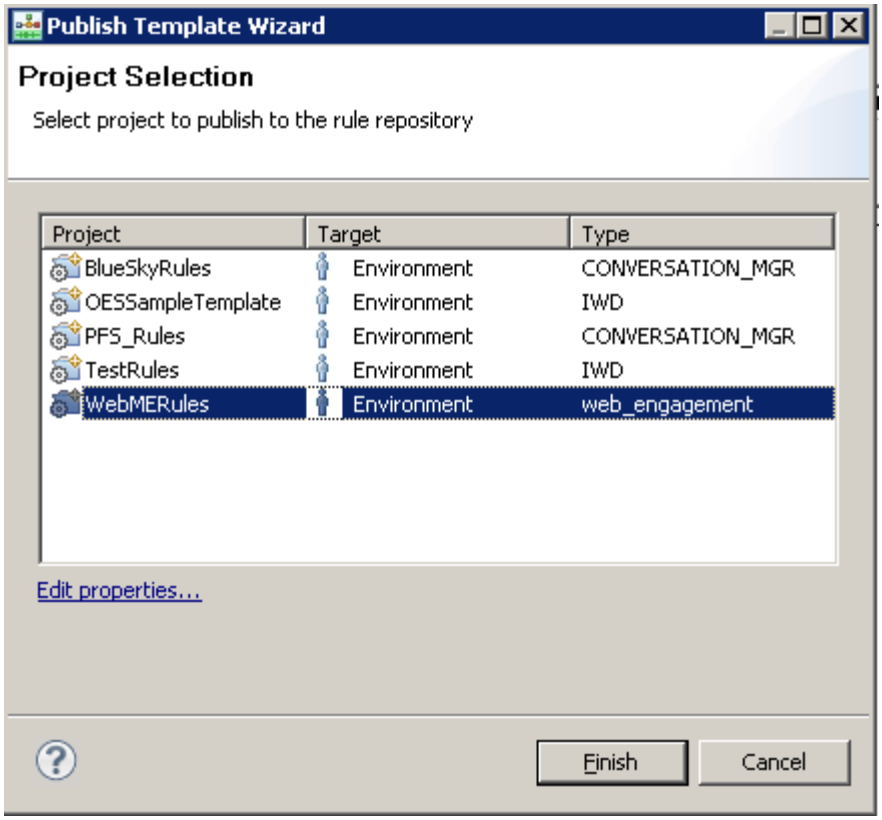
Start

1. In `Project Explorer`, right click on `WebEngagement_CEPRule_Templates`.
2. Select `Publish`. The `Publish Template Wizard` dialog box opens.



GWE-publishInComposer.PNG

3. Select `WebEngagement_CEP_Rule_Templates`.



GWM Publish selectRulesInComposer.png

Click to enlarge

4. Click `Finish`.

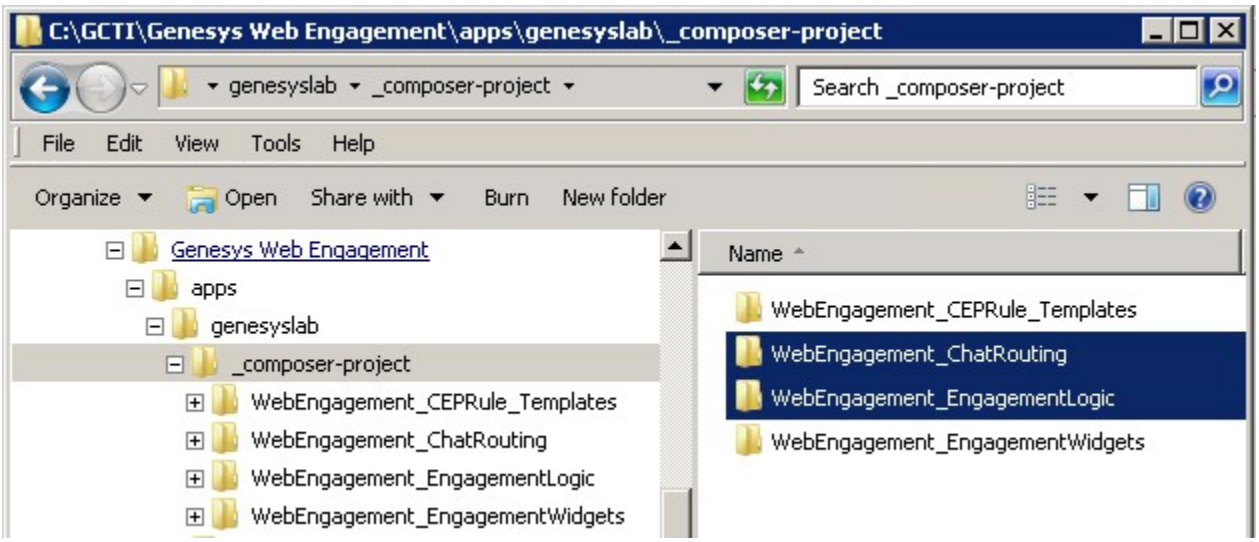
End

Next Steps

Build and Deploy

Customize the SCXML

When you create your application, Genesys Web Engagement also creates default chat routing and engagement logic strategies in the `\apps\<application_name>\_composer-project\` folder.



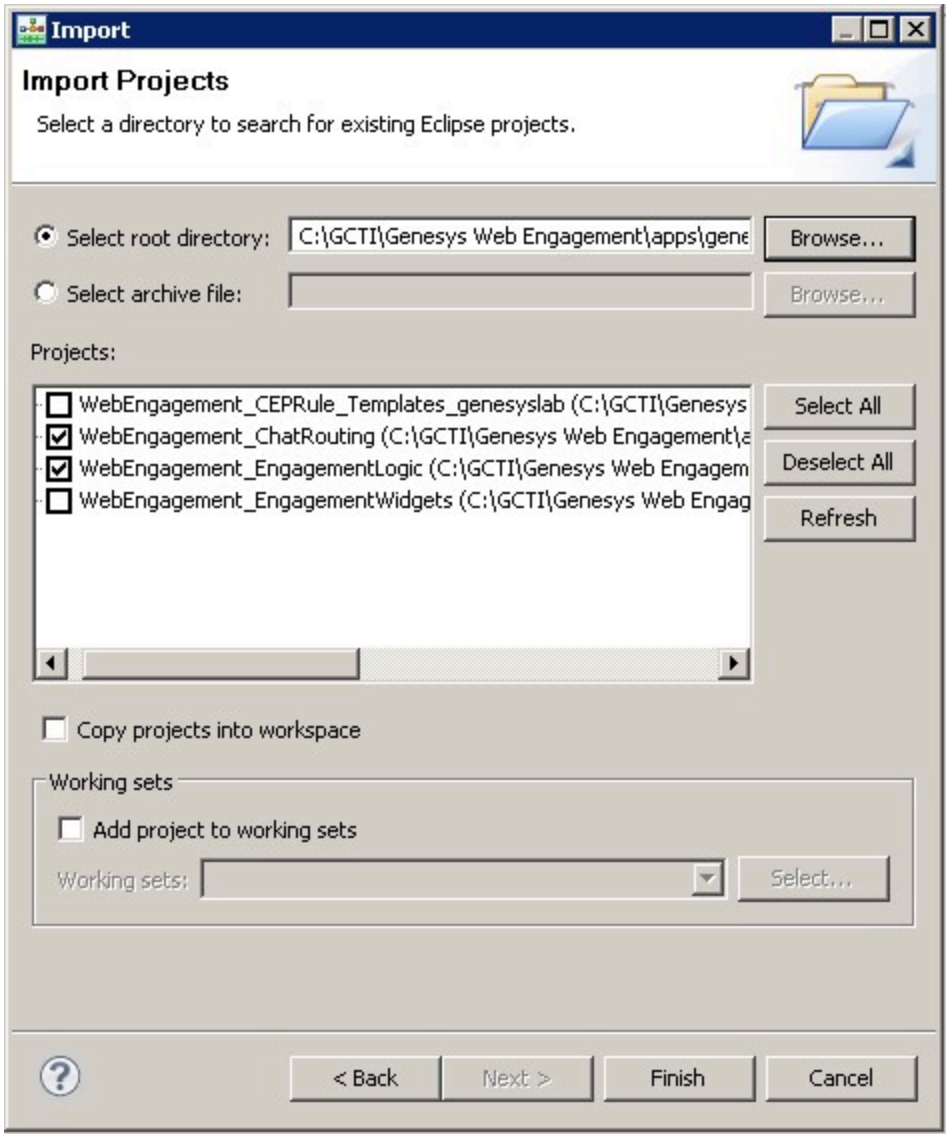
The default strategies for chat and engagement are in the _composer-project folder.

Orchestration Server (ORS) uses these strategies to decide whether and when to make a proactive offer and which channels to offer (chat or web callback).

You can modify these strategies by importing them into Composer.

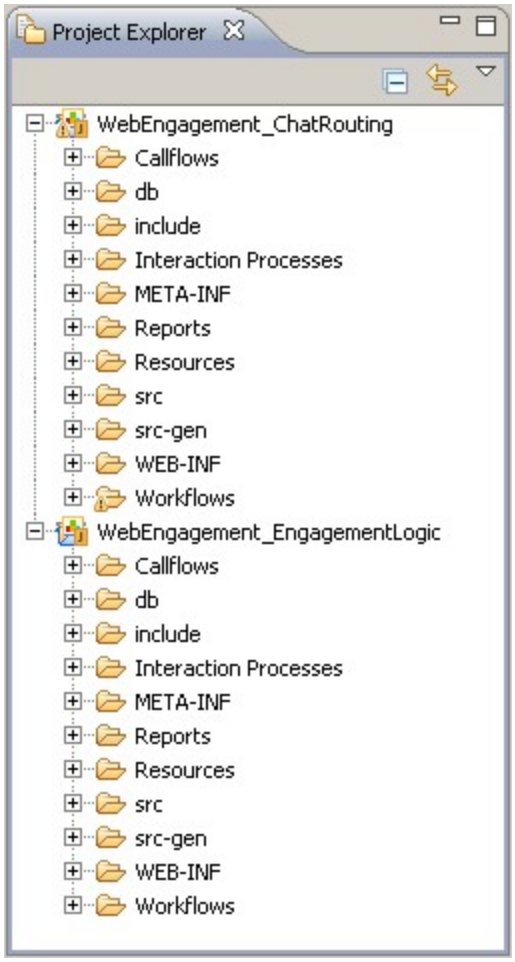
Import the chat routing and engagement logic strategies

1. Open Composer.
2. Select `File > Import...`.
3. In the `Import` dialog window, navigate to `General > Existing Projects into Workspace`. Click `Next`.
4. Select `Select Root Directory:`, then click `Browse`.
5. Import your project:
   - Navigate to the `\apps\<application name>\_composer-project` folder and click `OK`. The list of projects available is added to the `Projects` list.
   - Select the `WebEngagement_ChatRouting` and `WebEngagement_EngagementLogic` projects.
   - Warning: Do **not** enable the option `Copy projects into workspace`.



Import the projects by clicking Finish.

- Click `Finish` to import the project. The `WebEngagement_ChatRouting` and `WebEngagement_EngagementLogic` project are added to the `Project Explorer`.



The chat routing and engagement logic projects are added to the Project Explorer.

End

Web Services

You can use web services in your SCXML in the following ways:

- Use a `State` block with `session:fetch`. For example:

```
<session:fetch srcexpr="BackendURL + '/data/gateway/engage'" method="'post'"
type="'application/json'" enctype="'application/json'">
   <content _expr="ixnProfile"/>
</session:fetch>
</onentry>
```

- Use a `Web Request` or `Web Service` block. In this case, Composer requires this logic to be hosted as a web application, which means the entire Composer project must be hosted outside of the Web Engagement application. With Composer, you can export the project as a web application in WAR

format.

You must set the Backend Server configuration option wmsg.connector.scxml.appUrl to the SCXML resources hosted outside of the Web Engagement application.

If you use queue-based routing, you must also set the wmsg.connector.scxml.incomingInteractionQueue option to your new queue name. Alternatively, you can go to the `Scripts\WebEngagement_EngagementLogic.queueBased.Incoming.Routing\Annex\Application` and change "url" to the new server.

Configure Authentication in the default SCXML strategy

Purpose: To add security credentials to the default SCXML strategy to support authentication for the REST API.

Prerequisites

- Your Web Engagement Backend Server supports authentication. See Authentication for details.
- Your strategy uses the REST API.

Start

1. Go to `\apps\<application_name>\_composer-project\WebEngagement_EngagementLogic\src-gen` and open the `default.scxml` file in Composer, a text editor, or an XML editor.
2. Find the variables for `user` and `password` and set your authentication credentials. For example:

```
var user = 'user1';
var password = 'password1';
```

3. Save your changes.

End

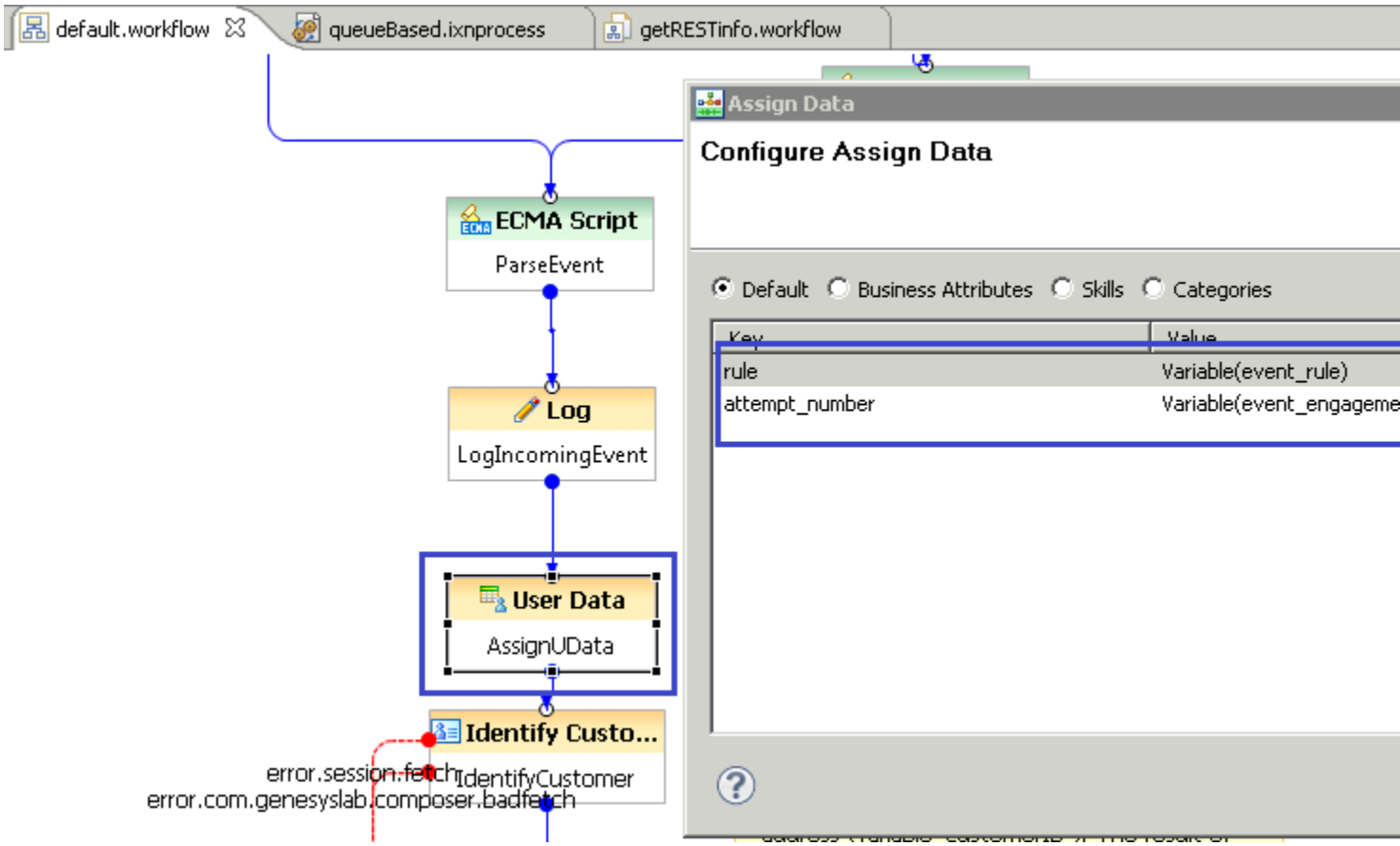
Propagate UserData

You can propagate user data from the Open Media (webengagement) interaction into the engagement interaction (chat or webcallback) by following two steps:

- Attach UserData to the Open Media webengagement interaction inside of the engagement strategy.
- Copy the filtered UserData from the Open Media interaction into the engagemet strategy.

Attach UserData to the Open Media interaction

All data that comes from System events is stored in the Open Media webengagement interaction as a KVlist under the key `jsonEvent`. You can access this data from the engagement strategy, but the `jsonEvent` key is not stored in the Universal Contact Server (UCS) database, which means it cannot be copied into the engagement interaction. If you want to store this user data in the UCS database or copy it into the chat or webcallback engagement interaction, you must attach it manually to the Open Media webengagement interaction in the engagement strategy. For example, you can do this with the User Data block:



Assigning UserData

This allows you to store all attached user data in the UCS database, but you can also control exactly which data to copy into the child chat or webcallback engagement interaction.

> **Important**
>
> Genesys recommends that you collect all the data you need and attach it to the interaction in a single Assign Data block. You should avoid using the multiple Assign Data blocks unless is it absolutely necessary.

Copy filtered UserData to the engagement interaction

When a chat or webcallback interaction is created, Genesys Web Engagement attaches the UserData available in its parent Open Media webengagement interaction. You can control how this data is attached by using the wes.connector.interaction.copyUserData option in the `service:wes` section of the Backend Server application. This option has three modes:

- Copy all UserData
- Don't copy UserData
- Copy only specific KV pairs from UserData

The following tables provides example values for the wes.connector.interaction.copyUserData. In these examples, the Open Media webengagement interaction UserData contains the keys **`ORS Data, rule, strategy, some data`**.

Value of wes.connector.interaction.copyUserData	Data in the engagement interaction
all	All keys are copied: `ORS Data, rule, strategy, some data`
no	No keys are copied.
rule;strategy	The `rule, strategy` keys are copied.
<blank or empty>	If the value of wes.connector.interaction.copyUserData is absent or has an empty value, no keys are copied.
my_key1;ORS Data	The `ORS Data` key is copied. `my_key1` is ignored because it is not part of the keys in the Open Media webengagement interaction UserData.

Attaching data for a negative engagement scenario

The default engagement strategy uses UserData propagation to store UCS database information about why a negative decision occurred.

The following keys are attached to the Open Media webengagement interaction and then saved in the UCS database:

- **noEngageCode** — The code for the negative decision reason.
- **noEngageDescription** — The extended description of the negative decision reason.

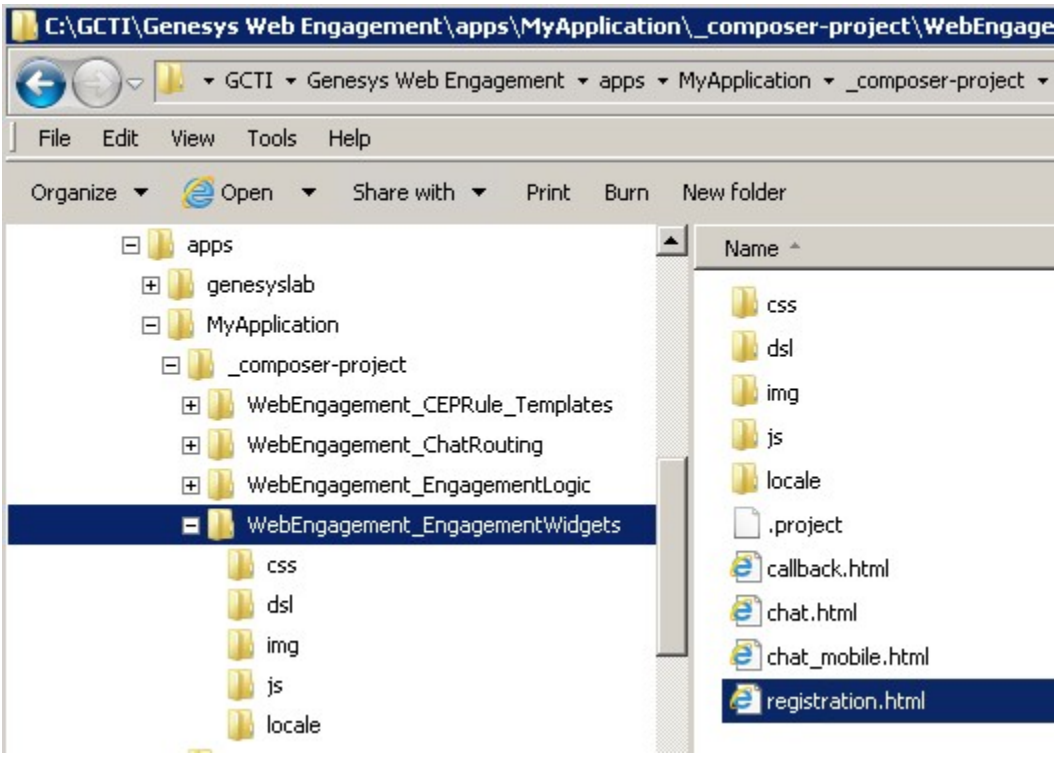
You can modify this in the engagement strategy to suit the requirements for your website.

Customize the Browser Tier Widgets

Genesys Web Engagement includes pre-integrated Browser Tier widgets that are used for engagements. These widgets are based on HTML, CSS, and Javascript, and can be customized to suit the look and feel of your website.

When you create an application, Genesys Web Engagement creates a Composer project that has a directory dedicated to the widgets: `apps\<application_name>\_composer-project\WebEngagement_EngagementWidgets`.

This directory contains all the files you can use to customize your widgets:



The WebEngagement_EngagementWidgets directory structure

- `css` — The individual CSS files used to style the widgets.
- `dsl` — The DSL for your application.
- `img` — The logo image used in the widgets.
- `js` — A Javascript file you can use to customize the dimensions and placement of the widgets on your web pages.
- `locale` — The files used to localize the widgets.

Examples

The following examples provide instructions for ways you might need to customize the widgets:

- Add localization files to your Web Engagement application
- Modify the invite message
- Modify the logo used in the widgets

Add Localization Files

	**Purpose:** To add Localization Files to your Web Engagement application.

The localization files must be compliant with jquery.localize.js, a jQuery plug-in that makes it easy to internationalize and localize your static web site. See the official website for further information on jquery-localize.

Add Localization Files to Your Web Engagement Application

 **Purpose:** To enable additional languages for internationalization on your web engagement widgets.

Start

1. Open the Web Engagement Installation folder and navigate to the `[application name]\_composer-project\WebEngagement_EngagementWidgets\locale` directory. This folder contains the localization resources for each widget:
   - `registration-<lang>.json` for the registration form displayed to anonymous users.
   - `chat-<lang>.json` for the chat invite.
   - `callback-<lang>.json` for the callback invite.
2. To add a new supported language `<lang>` for these widgets, where <lang> is the short locale name of the language (en, fr, ru etc.) or full locale name in IETF (en-US, fr-FR), follow these steps:
   - Create a copy of the `<name of the widget>-en.json` locale file.
   - Rename it to: `<name of the widget>-<lang>.json`.
   - Edit `<name of the widget>-<lang>.json` and replace all the text values with your translations.
   - Save.
3. To deploy these localization files:
   - Stop the Web Engagement Servers.
   - Build your application.
   - Deploy your application.
   - Start the Web Engagement Servers.

End

Customize Invites

Purpose: To describe how to customize the engagement invites.

Modifying the Invite Message

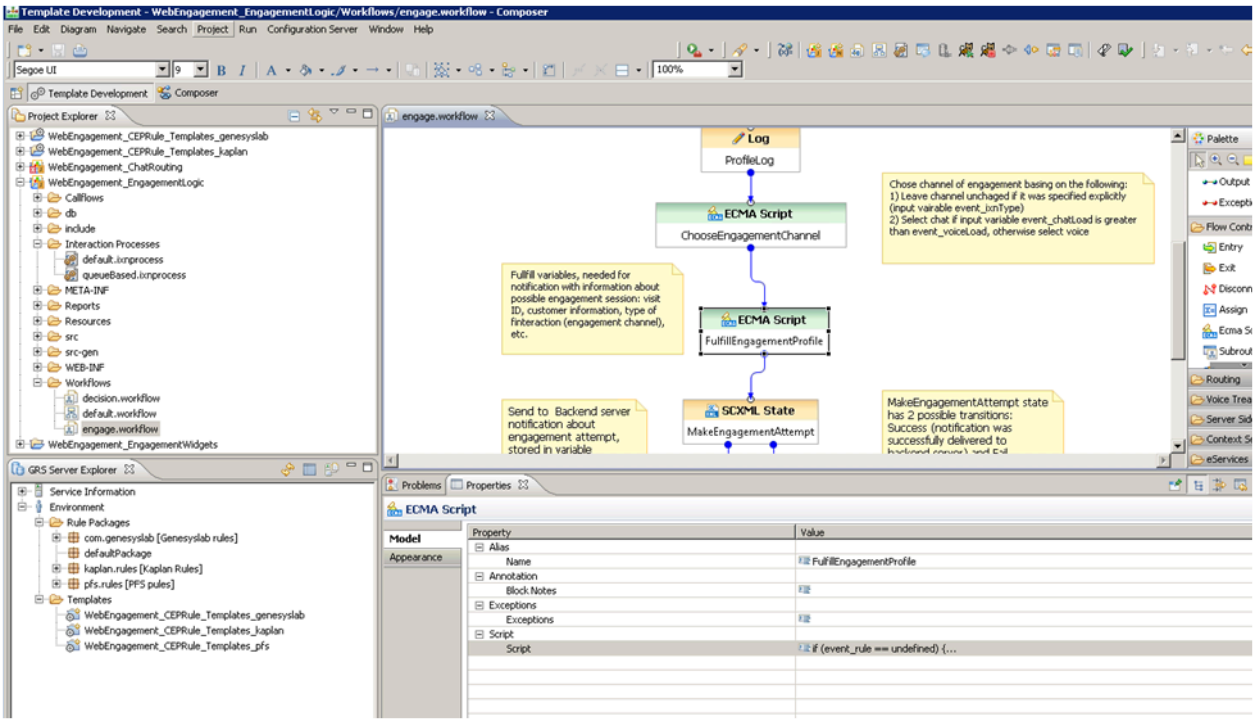
If an actionable event is submitted to the Web Engagement Backend Server, the Genesys Web Engagement application displays a chat or web callback invitation to engage the customer. This invite message in the dialog box is defined in the `WebEngagement_EngagementLogic` project, located in `apps/<application name>/_composer/`. Complete the following procedure to modify the invite message:

Prerequisites

- The Genesys Rules Development Tool is installed, configured, and opened in Composer or in Eclipse.
- The following procedure uses Composer.

Start

1. In the Genesys Rules Authoring Developer Tool, select `File > Import...`.
2. In the `Import` dialog window, navigate to `General > Existing Projects into Workspace`. Click `Next`.
3. Select `Select Root Directory:`, then click `Browse`. Select the `WebEngagement_EngagementLogic` project located in `apps/<application name>/_composer/`.
   Mandatory: Do NOT enable the `Copy projects into workspace` option.
4. Import your project.
5. Right-click the project and select `Upgrade Composer Project`. Wait until the upgrade is complete.
6. In the Project explorer, expand `WebEngagement_EngagementLogic/Worflows/engage.workflow` and click on the ECMA Script object named `FulfillEngagementProfile`. In the properties pane, click on Script properties.



7. Edit the ECMA Script and update the content of the message field in the variable engageProfile: `var engageProfile = { 'visit_id': event.visitID, 'nick_name': profile.FirstName, 'first_name': profile.FirstName, 'last_name': profile.LastName, 'email_address': customerAddress, 'subject': channelName, 'message':'Hello. Would you like assistance with the topic of current page? Agents are available now to answer your questions.', 'time_zone_offset': 8, 'wait_for_agent' : false, 'routing_point':sipRoutingPoint, 'ixn_type': channelType, 'pageId': event.pageID, 'inviteTimeout': 30};`
8. Save your changes and generate the code. **Note:** This step must be completed to make your changes available in the Genesys Web Engagement servers.
9. To make the changes available in the Genesys Web Engagement servers:

- Stop the servers in Genesys Administrator.
- Build and Deploy your Application.
- Start your Web Engagement Servers.

You may have to clean your cache to see the changes.

Modify the Logo used by the Frontend Server

1. To modify the logo:
   - Rename your logo image to `logo_small.png` and copy the file in `apps\<application name>\frontend\src\main\webapp\resources\img`.

     Or:

- Edit the `weinvite.css` file in the `\apps\<application name>\frontend\src\main\webapp\resources\css\` directory, and modify the name and relative path of your logo here:

```
.wedialog .dialog-content .branding-content{/*...*/    background-image: url(../img/logo_small.png);/*...*/}
```

2. Save your changes and generate the code. **Note:** This step must be completed to make your changes available in the Genesys Web Engagement servers.
3. To make the changes available in the Web Engagement servers:
   - Stop the servers in Genesys Administrator.
   - Build and Deploy your Application.
   - Start your Web Engagement Servers.

You may have to clean your cache to see the changes.

Build and Deploy

	**Purpose:** To build and deploy your Genesys Web Engagement application.

Once you have created and configured a Genesys Web Engagement application and implemented a Genesys Web Engagement model, you must build and deploy your application, before you start your Servers.

Build your Application

 **Purpose:** To create the application's files that will be deployed in the next step.

Start

1. Navigate to the installation directory of Genesys Web Engagement and open a new Windows Console.
2. Type:

   ```
   build <application name>
   ```

End

The script starts building `.war` files used for deployment. If the build is successful, the console output displays a `BUILD SUCCESSFUL` messages at runtime, and the `.war` files are created in the following Genesys Web Engagement sub-directories:

- `apps\<application name>\backend\target\MyProject-wmbackend-0.1`
- `apps\<application name>\frontend\target\MyProject-wmfrontend-0.1`

In addition, a new `map.xml` file is created in the `\apps\<application name>\proxy\target\` directory.

Next Steps

Deploy your Application

Deploy your Application

 **Purpose:** Copy the application's file at the proper location.

Prerequisites

- Your build was successful. If your build fails due to errors, try to fix them, then rebuild. You can neither deploy nor start your Web Engagement servers if the build is not successful.

Start

1. Navigate to the installation directory of Genesys Web Engagement and open a new Windows Console.
2. Type:

   ```
   deploy <application name>
   ```

End

The deploy script copies files to the appropriate locations. If the deploy is successful, the script output displays the following messages:

```
Application '<application name>' is used
1 file(s) copied. 1 file(s) copied. 1 file(s) copied.
```

Do not deploy if errors occurred during the building step! See Build your Application.

Next Steps

Start your Servers

Start your Servers

	**Purpose:** To describe how to start the servers.

Configure the Web Engagement Channel

 **Purpose:** To configure the engagement channel in the Backend Server's configuration options.

Start

1. Open Genesys Administrator and navigate to `PROVISIONING > Environment > Applications`. Select the application defined for the Web Engagement Backend Server and click `Edit...`
2. Select the options panel. In the `service:wmsg` section , select the `wmsg.connector.defaultEngagementChannel` option and enter the engagement mode that you wish to implement for your application: `proactiveChat`, `proactiveCallback`.

End

Next Steps

Configure the Registration Form

Configure the Registration Form

 **Purpose:** Set up the Registration Form in the Backend Server's configuration options. This option displays the registration form to anonymous customers when they are engaged by the Solution.

Start

1. Open Genesys Administrator and navigate to `PROVISIONING > Environment > Applications`. Select the application defined for the Web Engagement Backend Server and click `Edit...`
2. Select the options panel. In the `service:wmsg` section, select the `wmsg.connector.wns.showRegistrationForm` option. Set to all to display this form for all types of engagement. See the wmsg.connector.wns.showRegistrationForm for details about possible values.

End

Next Steps

Start the Web Engagement Servers

Start the Web Engagement Servers

Purpose: If you've been through all the previous steps successfully, you can start your Web Engagement Servers from either **Genesys Administrator**, or the **Start.bat** script.

Start

- You can start servers from the Genesys Administrator.
  1. Navigate to `PROVISIONING > Environment > Applications`.
  2. Select the Web Engagement Servers.
  3. Click `Start applications` in the Runtime panel.
- Or, use the provided `Start.bat` script.
  1. Navigate to the Web Engagement installation directory and launch the Windows Console of commands (cmd.exe).
  2. Type:
     start.bat

End

The Web Engagement Backend Server and Web Engagement Frontend Server are started in Genesys Administrator.

Next Steps

Summary of Tasks

Create a Rules Package

	**Purpose:** To create a rules package for your application.

Rules are mandatory for managing actionable events generated from the SYSTEM and BUSINESS event flows submitted by the Browser Tier. To add rules, you must create a package, then a set of rules.

Create a Rules Package

 **Purpose:** To create the Rules package associated with your Web Engagement application. The following section is an example of Rules package creation. For further information about rules creation, refer to the Genesys Rules System Deployment Guide.

Prerequisites

- Genesys Rules Authoring, version 8.1.200.17 and later, is installed.
- Roles are configured to enable your user to create rules.

Start

1. Navigate to `Environment > Solution > New Rule Package`.
2. In the `General` tab:
   - Enter a Package Name—for example, `myproject.rules.products`;
   - Enter a Business Name—for example, Products;
   - Select web_engagement for `Package Type`. `WebEngagement_CEPRule_Templates` appears in the `Template` table;
   - Optionally, enter a description;
3. Select `WebEngagement_CEPRule_Templates` in the `Template` table.

GWENewPackageGRA.PNG

Click Save.

4. Click Save.

End

Only one package of rules can be active in the selected Web Engagement Backend Server. Create all your rules in the same package for your application.

Next Steps

Create Rules in the Rules Package

Create Rules in the Rules Package

Purpose: To create the rules according to your model.

Prerequisites

- You created the rules package; for instance, `myproject.rules.product`.

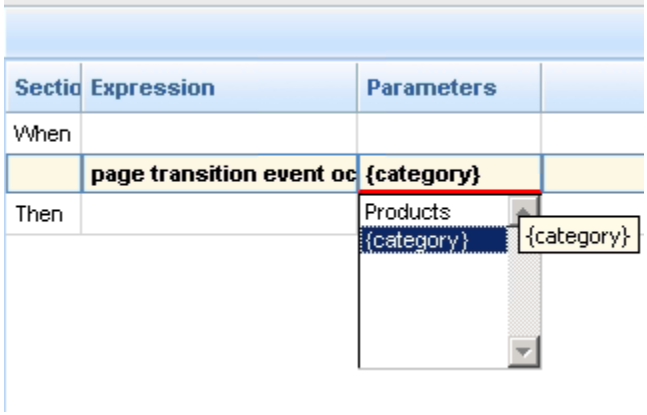
Start

1. Select the `myproject.rules.product` package.
2. Select the `Rules` tab.
3. Click `New Linear Rule`. This creates a new rule in the `Rules` table.
4. Select the created rule:
   - Enter a Name—for example, Products;
   - Enter a Phase—The list of rule phases can be modified by changing the values of the enumeration that is called Phases, in the CEP rules Template. In beta, the single value available is `*`.
5. Click `Add Condition`:
   - Scroll down to select a condition—for example, `page transition event occurs that belongs to category`, which launches the actionable event anytime that a user enters or leaves a page of your website.

Select your rule's condition.

Select your rule's condition.

- Select a category in `Parameters`—for example, `Products`. The `Parameters` list displays the categories that you previously created in the Genesys Administration Extension.



Set the condition's parameters.

6. Click `Add Action` and select an action in the list—for example, `generate actionable event`;
7. Click `Save...`.

End

You can create as many rules as you need in your Rules package.
For details about the available rules templates, see the available templates for:

- The category-based Rules
- The event-based Rules

Next Steps

Publish the Rules Package

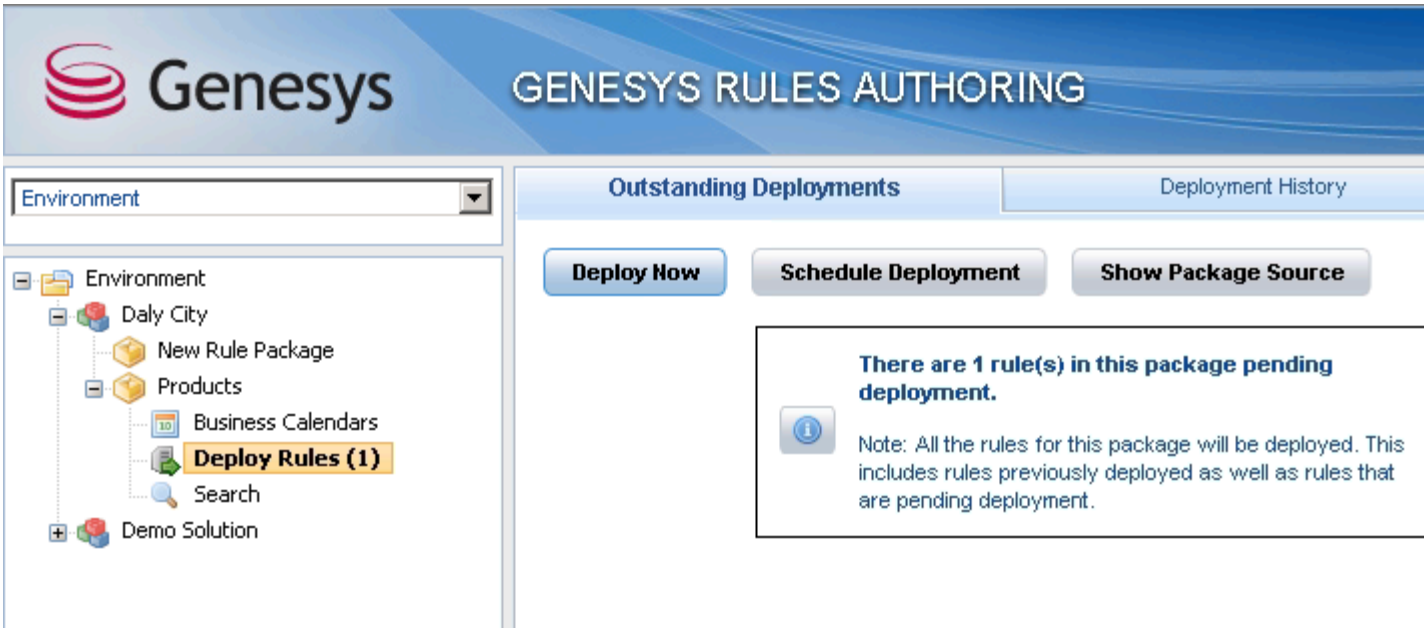
Publish the Rules Package

Prerequisites

- The Web Engagement Backend Servers are started.

Start

1. In Genesys Rules Authoring Tool, navigate to `Solution > Your_Rules_Package > Deploy Rules`.



Click Deploy Rules.

2. Select your Web Engagement Backend Server.
3. Click `Deploy Rules`.

End

If the deployment is successful, the following message appears: `There are 0 rule(s) in this package pending deployment.`

Next Steps

Back to Task Table

Test with GWM Proxy

Purpose: To configure and run the GWM Proxy.

This proxy is a development tool, which enables you to test your application without adding the JavaScript Tracking code to your website. Once you have configured this proxy, you can launch it and start the Genesys Web Engagement servers to start testing your application by emulating visit on your website.

Configure and Start GWMProxy

Retrieve the GWMProxy Port

1. Navigate to `C:\Users\<current user>\GWMProxy`.
   If this folder does not exist, navigate to your Web Engagement installation directory and launch `servers\proxy\startserver.bat`. GWMProxy appears automatically.
2. Edit `config.xml` and find the tag <proxy>.
3. Check that the value of <ip> tag is set to your host IP address;
   Values 127.0.0.1 or localhost are not allowed!
4. Note the value of the <port> tag (usually 15001), required to set up your browser.

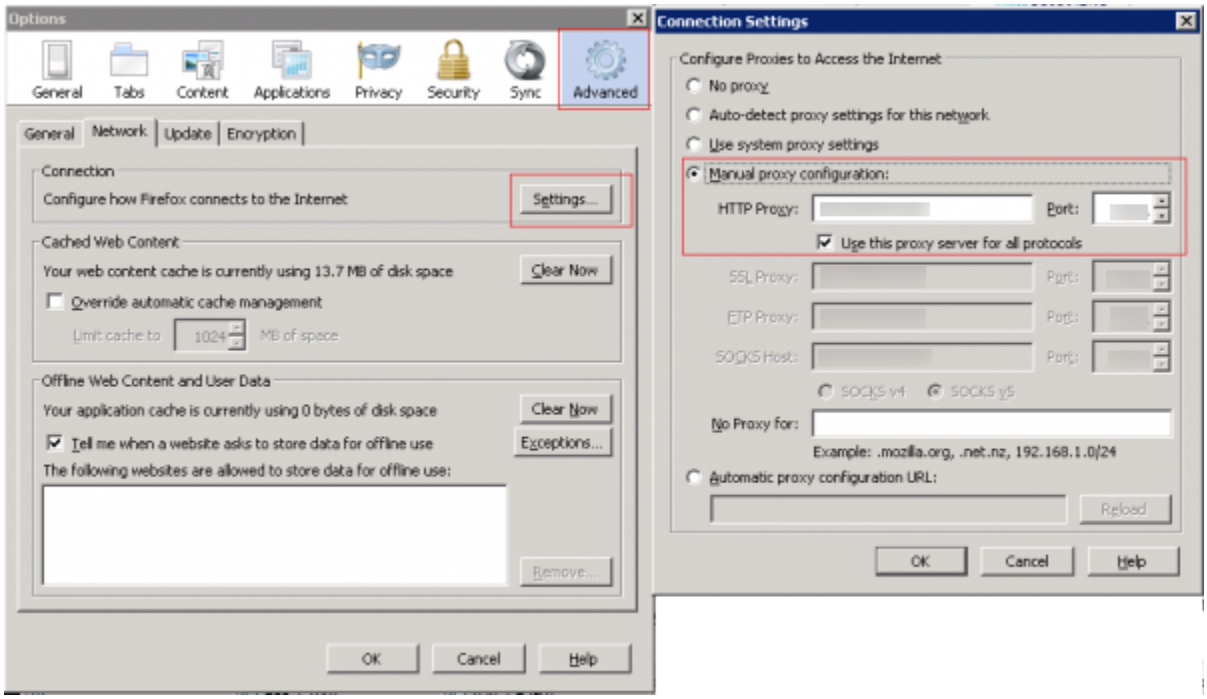
Start the Proxy

Navigate to your Web Engagement installation directory and launch `servers\proxy\startserver.bat` The GWMProxy starts.

Set Up your Web Browser

1. Start your Web Browser.
2. Open your Internet settings. For instance, in Mozilla Firefox, select `Tools > Options`. The `Options` dialog window appears.
3. Select `Advanced`, and in the `Network` tab, `click Settings...` The `Connection Settings` dialog windows appears.
4. Select the `Manual proxy configuration` option:
   - Enter your host IP address in the HTTP proxy text box;
   - Enter the port used by the GWMProxy in the Port text box;
   - Select the option `Use this proxy server for all protocols`.

5. Click OK. Now your browser is using the GWMProxy which will inject the Web Engagement tracking code to the webpages of the monitored domain.



GWMProxy used in Firefox

Injecting code in the HTTP Response of the GWMProxy

Going further, you can edit the `map.xml` file available in the `/tools/proxy` directory to customize the code injected in the HTTP response retrieved through the proxy. In that purpose, you must insert your code under the `<content></content>` elements of map.xml with CDATA masking. If you want to browse a secure domain, insert your code under `<secure><content>…</content></secure>` elements, otherwise, use the `<simple><content>…</content></simple>` element.

In the root <map> tag, the "replace" attribute uses regular expressions to specify where the code must be injected. For instance, the string "%s </head>" means that the "%s"code must be added before </head> tag "</head>".

Do not forget to restart proxy when you are done with your modifications.

The following `map.xml` file injects the DSL code in the HTTP response.

```
<?xml version="1.0"?>
<mapping>
    <map replace="%s </head>" domains="genesyslab.com;www.genesyslab.com;www-
ssl.genesyslab.com;default.proxy.lucent.com">
        <simple>
            <content>
                <![CDATA[<script>
```

```
                var _gt = _gt || [];
                _gt.push(['config', {
                                'name': 'genesyslab.com',
                                'domainName' : 'genesyslab.com',
                                'languageCode': 'en-US',
                                'mobile': false,
                                'dslResource': 'http://demosrv:8081/frontend/resources/dsl/
domain-model.xml',
                                'secureDslResource': 'https://demosrv:8443/frontend/resources/
dsl/domain-model.xml',
                                'httpEndpoint': 'http://demosrv:8081',
                                'httpsEndpoint': https://demosrv:8443'}]);
                (function() {
                                var gt = document.createElement('script'); gt.type = 'text/
javascript'; gt.async = true;
                                gt.src = ('https:' == document.location.protocol ?
'https://demosrv:8443' :
                                                                        'http://demosrv:8081') +
'/frontend/resources/js/GTC.js';
                                var gts = document.getElementsByTagName('script')[0];
gts.parentNode.insertBefore(gt, gts);
                })();

</script>
]]>
            </content>
        </simple>
    </map>
</mapping>
```

Checking Hosts in Window\System32 Configuration

Edit the `c:\Windows\System32\drivers\etc\hosts` file with a text editor. Make sure that your IP address is correctly associated with your host's complete name. For instance:

```
192.168.3.103 MyHostName MyHostName.MyWebSite.com
```

If you are using a Genesys Demo image and if you changed your machine IP Address, demosrv.genesyslab.com may still be referred in some genesys server configuration and prevent theses servers from working correctly. Add the following declaration to bypass this issue:

```
127.0.0.1 demosrv.genesyslab.com
```

In addition, navigate to `C:\Users\Administrator\My Documents\Documentation\DemoPlatform\Changing IP Address` and read machine.txt for further information.

Enable Monitoring

	**Purpose:** To enable the monitoring of your website.

You must add a short standardized section of JavaScript code to your pages. This snippet of JavaScript is independent from the Web Engagement Model that you implemented (rules, categories, and DSL). When this code is evaluated at document load time, it loads the JavaScript libraries that implement the three agents, including the DSL-specific rules. These libraries then submit events according to the implemented model. For further information on the high level design of Genesys Web Engagement, see Architecture.

Retrieve the JavaScript Tracking Code

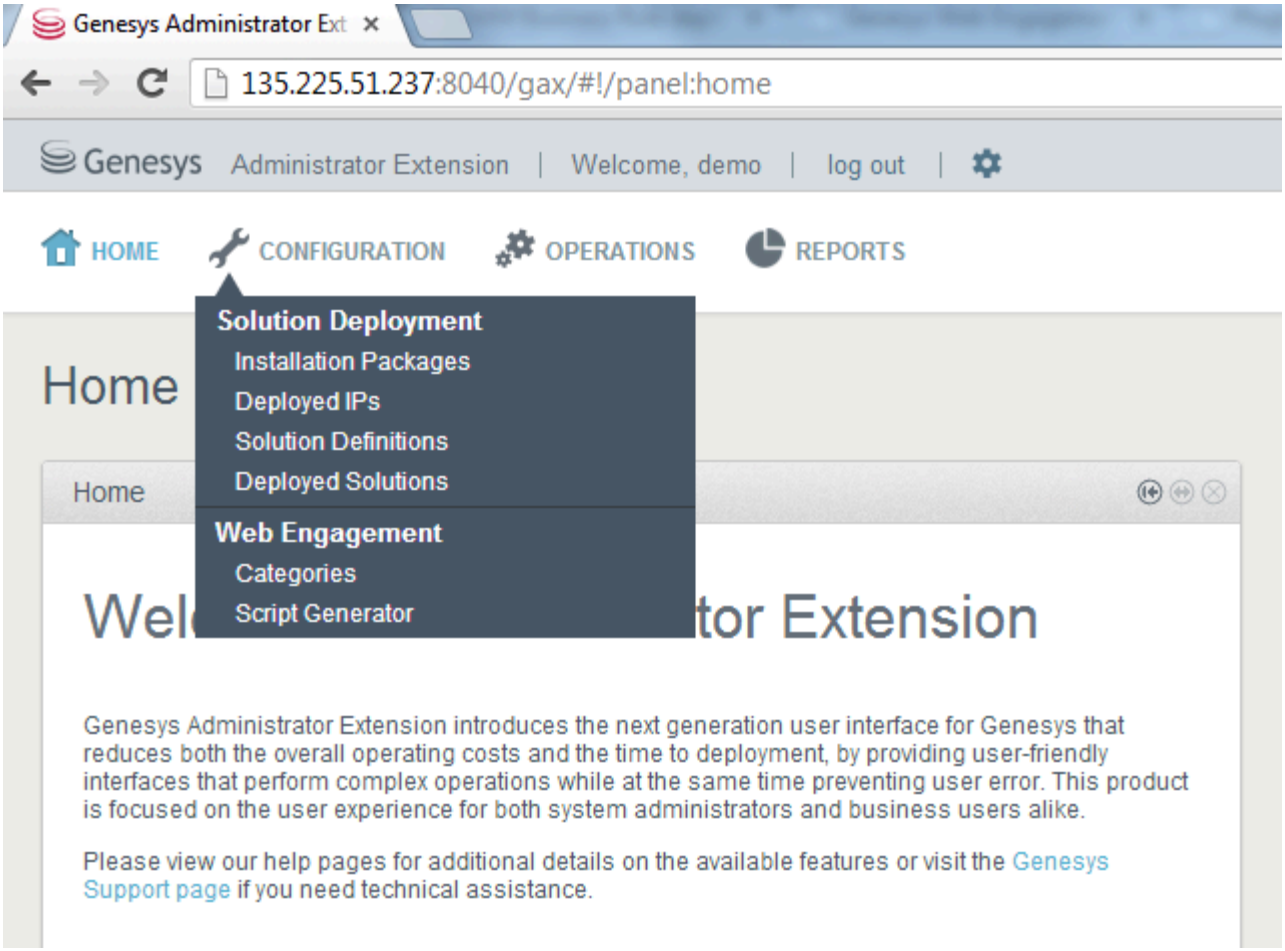
 **Purpose:** To retrieve the tracking code, available in the Genesys Web Engagement Plug-in for Genesys Administrator Extension that you installed previously.

Prerequisites

- The Plug-in for the Genesys Administrator Extension extension is already installed.

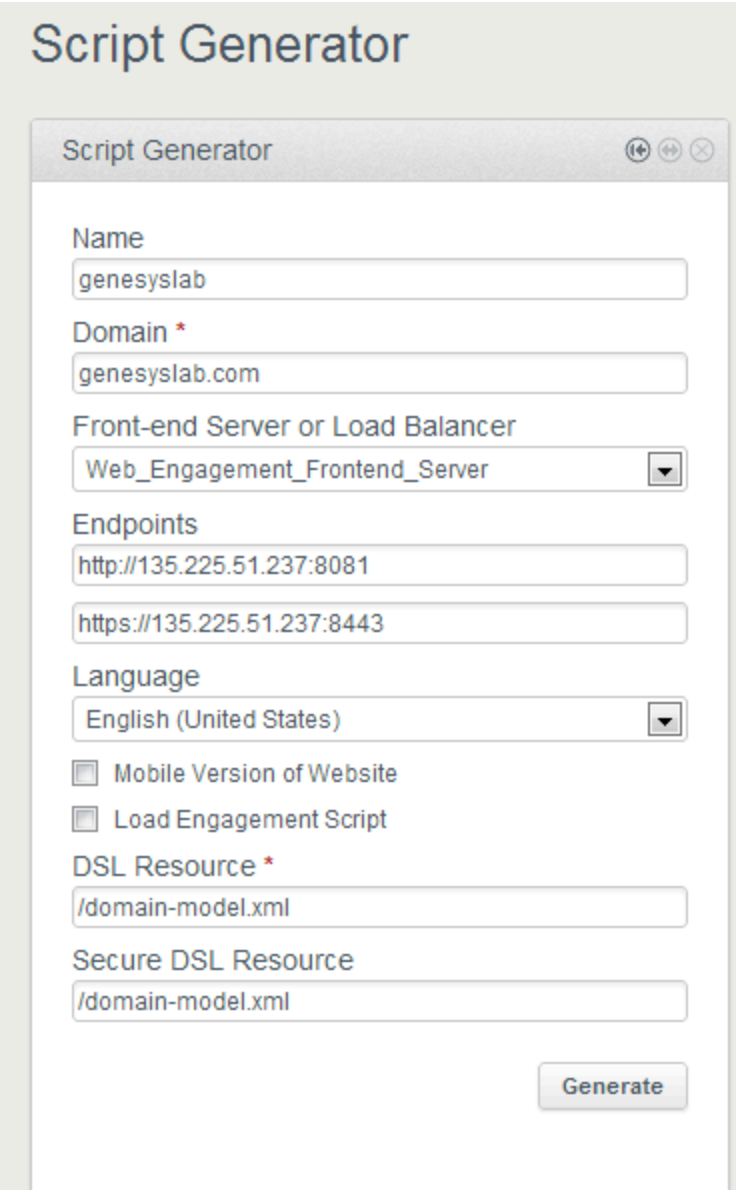
Start

1. Open Genesys Administrator Extension.



Main Home Panel of the Genesys Administrator Extension

2. Navigate to `CONFIGURATION > Script Generator`. The `Script Generator` interface opens.
3. Fill in the following fields:
   - Enter your application name for `Name`; for instance, `genesyslab`.
   - Enter the `Domain name`; for instance,`genesyslab.com`;
     Note: These parameters must be identical to the parameters specified when your created your application; see Define the Monitoring Domains
   - Select the correct Frontend Server. If you deployed Load Balancer and nodes, you must select the Load Balancing Frontend Server, not a node.
   - In `Endpoints`, enter the IP address and listening ports associated to your Load Balancing Frontend Server;
   - Enter the paths to the DSL resources; the paths are relative to the`/frontend/resources/dsl` directory of your web engagemennt application; you can add your DSL resources to this directory or sub-directories.



Example for the genesyslab application

4. Click on the `Generate` button. The Generated Script panel opens.

End

- You can generate your script as many time as you need.
- You can customize this script as detailed in Plug-in Help

Next Steps

Add the JavaScript Tracking Code to your Website

Add the JavaScript Tracking Code to your Website

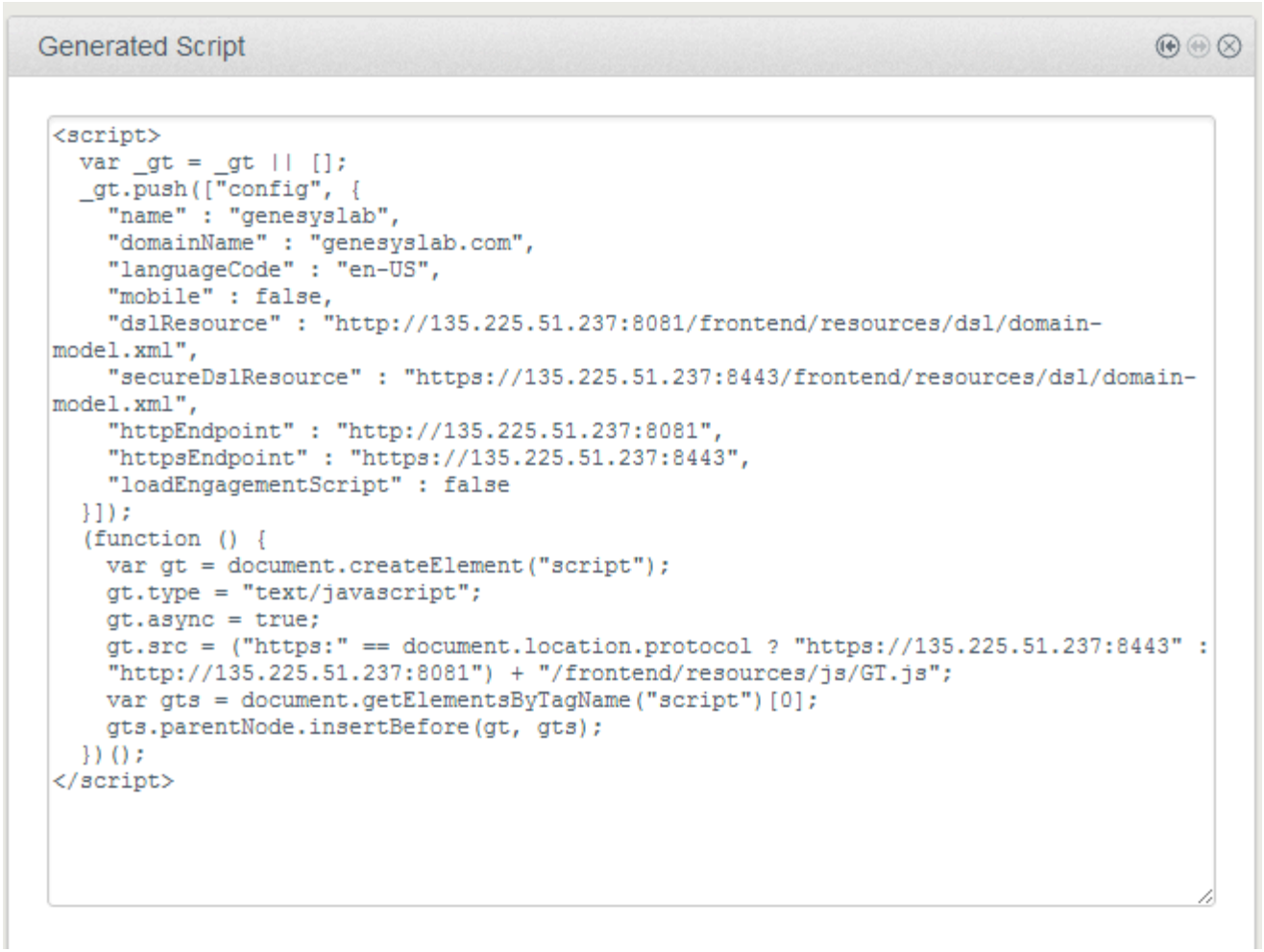
Purpose: To add the instrumentation script and tracking code to your website or your mobile application and start submitting data to Web Engagement Servers.
To set up tracking, you need access to the source code for your website.

Prerequisites

- You must remove any former or older tracking code snippet from your webpages.
- You generated a script with the Script Generator interface of the Genesys Web Engagement plug-in for Administrator Extension.

Start

1. Select and copy the script generated in the Generated Script Panel of the Administrator Extension.



The Tracking Script for genesyslab

2. Paste the JavaScript Tracking Code at the bottom of the `<head></head>` section of your webpages:
   - You can edit manually the webpages that you wish to monitor.
   - You can edit the header template of your website if you have one.
3. If your website includes additional scripts, modify the position of these scripts, as described here, to ensure best performance across all the browsers.
   - Reorder the scripts of the `<head></head>` section to ensure that the JavaScript Tracking Code is the last one of the section.
   - Make sure that additional scripts are located after the webpage contents: at the bottom of the `<body></body>` section, or in the `<footer></footer>` section of your webpage.

End

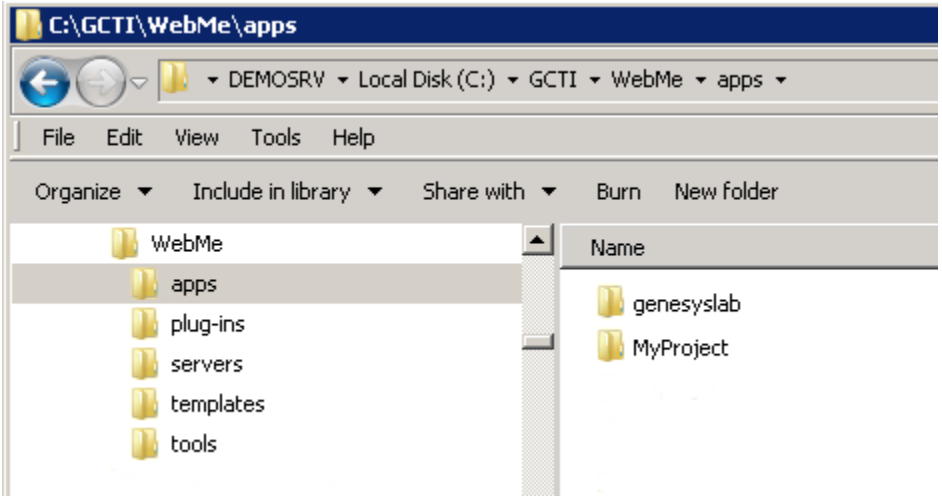
Genesyslab Sample

Purpose: To run and customize the Genesyslab Web Engagement Sample.

Description

The Genesyslab Engagement Sample provides scripts to build a Genesys Web Engagement application and test category-based capabilities through a specific proxy. Scripts create a custom application, including all the mandatory materials, such as the rules templates, the SCXML, the DSL scripts, and the Genesys Web Engagement servers, usable in a pre-packaged virtual environment. In a few clicks, without modifying your website, Genesys Web Engagement features will show up on a set of web pages, according to the rules and categories that you created.
For further information on the product's architecture, read Architecture. The Genesyslab Sample is provided as a zip file containing the following directory structure:

- `apps`—Contains the applications created with the .bat scripts.
- `plug-ins`—Contains the Interaction Workspace and Genesys Administrator eXtension plugins.
- `servers`—Contains the Genesys Web Engagement Servers.
- `templates`–Contains templates for the Genesys Web Engagement Servers.
- `tools`—Contains additional tools, including provisioning for the Genesys solution.
- `rules`—Contains the rules for Genesys Rules Authoring Tool.



GWE-WebMeFolderCapture.PNG

Get Started with the Genesyslab Example

The Genesyslab sample is a sub-directory in the `apps` folder, already configured. To play with this sample, you need to accomplish the Genesys Web Engagement Application Development Workflow, step-by-step, as detailed in Application Development:

1. In the `\apps\genesyslab\proxy directory`, update the `map.properties` file:
   - Set the `frontend.server.host` parameter to the FQDN or IP address of the host where GWMProxy will be run.
2. Build and deploy the application.
3. Start your Genesys Web Engagement servers.
4. Test with GWMProxy: You must use the proxy to add your sample code to Genesyslab.com webpages.

You do not need to complete the following tasks in the Application Development Workflow:

1. You do not need to create categories; provisioning automatically creates them.
2. You do not need to create additional business events; they are already created.
3. You do not need to Publish the CEP Rules Template to use this sample. After you build and deploy the application, the `rules.drl` file, which is created by default for the Genesyslab sample in `\apps\genesyslab\storage`, is copied to the Frontend Server.

Test Genesyslab Scenarios

To test the scenarios, you need to follow the steps detailed in the `\apps\genesyslab\readme.html` or `\apps\genesyslab\readme.doc` files.

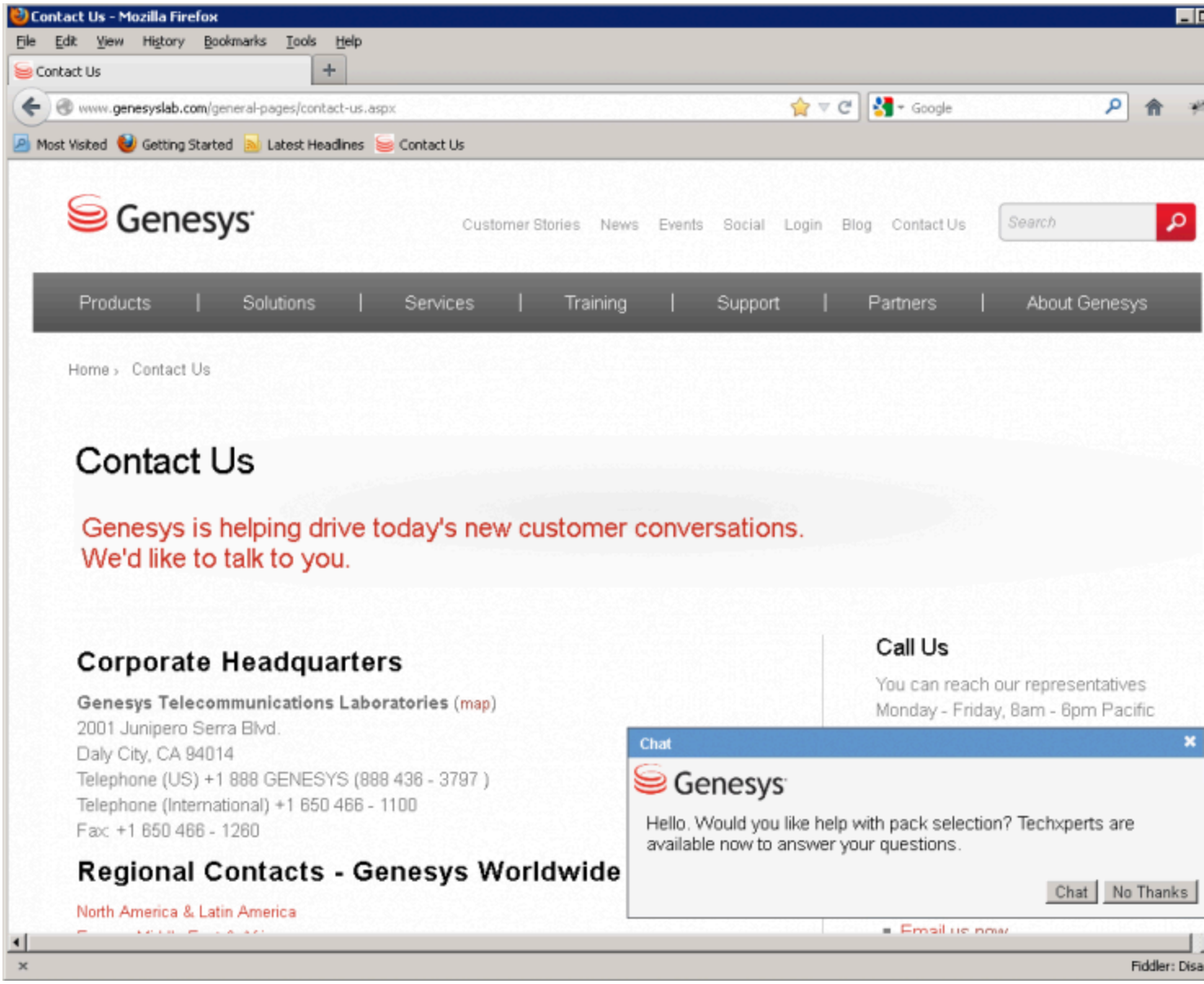
A pop-up window starts the engagement process. You can:

- Reject the engagement.
- Accept the voice callback or the chat conversation (if you modified the servers' options).

Then, by default, a registration form is displayed first to identify the customer. By default, Genesys Web Engagement is configured for the progressive pacing algorithm. To turn off the pacing algorithm and specify a particular engagement channel, you can use the wmsg.connector.defaultEngagementChannel option in the Web Engagement Backend Server application. You can modify this option in Genesys Administrator:

- Open Genesys Administrator. Navigate to `PROVISIONING > Environment > Applications`. Double-click on your server's application name.
- Edit the `wmsg.connector.defaultEngagementChannel` option in the `Options > service:wmsg` section. Valid values are `proactiveChat` or `proactiveCallback`. If no value is specified (default), the pacing algorithm will be used.

The following screenshot shows a proactive offer for chat, based on the "Singleton with Timeout" scenario.


Contact Us - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Contact Us
www.genesyslab.com/general-pages/contact-us.aspx
Most Visited Getting Started Latest Headlines Contact Us
Genesys
Customer Stories News Events Social Login Blog Contact Us
Search
Products | Solutions | Services | Training | Support | Partners | About Genesys
Home › Contact Us
Contact Us
Genesys is helping drive today's new customer conversations.
We'd like to talk to you.
Corporate Headquarters
Genesys Telecommunications Laboratories (map)
2001 Junipero Serra Blvd.
Daly City, CA 94014
Telephone (US) +1 888 GENESYS (888 436 - 3797)
Telephone (International) +1 650 466 - 1100
Fax: +1 650 466 - 1260
Regional Contacts - Genesys Worldwide
North America & Latin America
Call Us
You can reach our representatives
Monday - Friday, 8am - 6pm Pacific
Chat
Genesys
Hello. Would you like help with pack selection? Techxperts are available now to answer your questions.
Chat
No Thanks
Fiddler: Disa

Check Your Application Version

Purpose: To verify the version of the application that is running.

At runtime, you can retrieve details about your application and the Genesys Web Engagement servers, by viewing to the `about.jsp` files available in the Backend and Frontend Servers. This information is available at the following URLs:

- `http://<hostname>:<frontend port>/frontend/about.jsp` where:
  - <hostname> is the hostname for the Web Engagement Frontend Server;
  - <frontend port> is the listening port defined for your Frontend Server, for instance, 8081. See Configuring the Frontend Server for further details.
- `http://<backend hostname>:<backend port>/backend/about.jsp` where:
  - <hostname> is the hostname for the Web Engagement Backend Server;
  - <backend port> is the listening port defined for your Backend Server, for instance, 9081. See Configuring the Backend Server for further details.

For instance, the following output could be the `about.jsp` of the Frontend Server available for a demosrv host at the following URL:

```
http://demosrv:8081/frontend/about.jsp

Implementation-Title: genesyslab-wmfrontend
Implementation-Version: 0.1
Implementation-Vendor-Id: my.default.company
Built-By: Administrator
Build-Jdk: 1.6.0_27
Specification-Title: applicationName-wmfrontend
Build-Started-At: 20121220-0228
Created-By: Apache Maven 3.0.4
Specification-Version: 0.1
Archiver-Version: Plexus Archiver
-------
Implementation-Title: wmcommon
Implementation-Version: 8.1.100.16
Built-By: prodalu
Specification-Vendor: Genesys Telecommunication Laboratories, Inc.
Build-Started-At: 20121219-1653
Created-By: Apache Maven
Implementation-Vendor: Genesys Telecommunication Laboratories, Inc.
Build-Number: unknown
Implementation-Vendor-Id: com.genesyslab.wme
Build-Jdk: 1.6.0_27
Specification-Title: wmcommon
Specification-Version: 8.1.100.16
Archiver-Version: Plexus Archiver
-------
```

For instance, the following output could be the `about.jsp` of the Backend Server available for a demosrv host at the following URL:

```
http://demosrv:9081/backend/about.jsp

Implementation-Title: applicationName-wmbackend
Implementation-Version: 0.1
Implementation-Vendor-Id: my.default.company
Built-By: Administrator
Build-Jdk: 1.6.0_27
Specification-Title: genesyslab-wmbackend
Build-Started-At: 20121220-0228
Created-By: Apache Maven 3.0.4
Specification-Version: 0.1
Archiver-Version: Plexus Archiver
-------
Implementation-Title: wmcommon
Implementation-Version: 8.1.100.16
Built-By: prodalu
Specification-Vendor: Genesys Telecommunication Laboratories, Inc.
Build-Started-At: 20121219-1653
Created-By: Apache Maven
Implementation-Vendor: Genesys Telecommunication Laboratories, Inc.
Build-Number: unknown
Implementation-Vendor-Id: com.genesyslab.wme
Build-Jdk: 1.6.0_27
Specification-Title: wmcommon
Specification-Version: 8.1.100.16
Archiver-Version: Plexus Archiver
```

Using the Plug-in for GAX

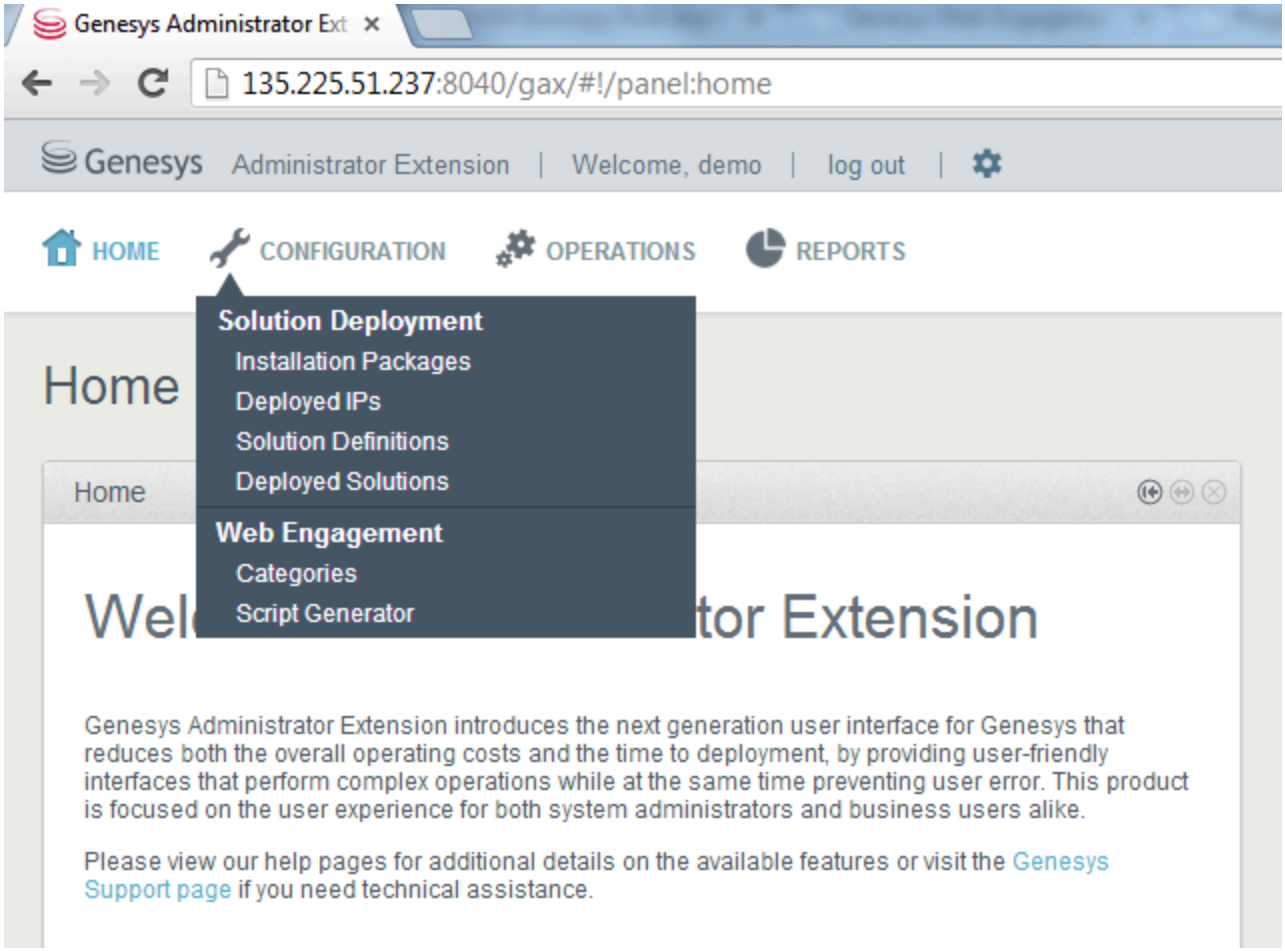
The Web Engagement Plug-in for Genesys Administrator extension includes two tools used in the creation of your Web Engagement application. Basic usage of this tool is described in:

- Create Categories
- Enable Web Engagement Monitoring

The following page provides detailed information about these tools.

Plug-in Description

Once installed, as detailed here, the Genesys Web Engagement Plug-in for Administrator Extension adds the Web Engagement section to the CONFIGURATION menu of the Genesys Administrator Extension.



Web Engagement menu in Genesys Administrator Extension

The plug-in provides two tools, used in Web Engagement developments:

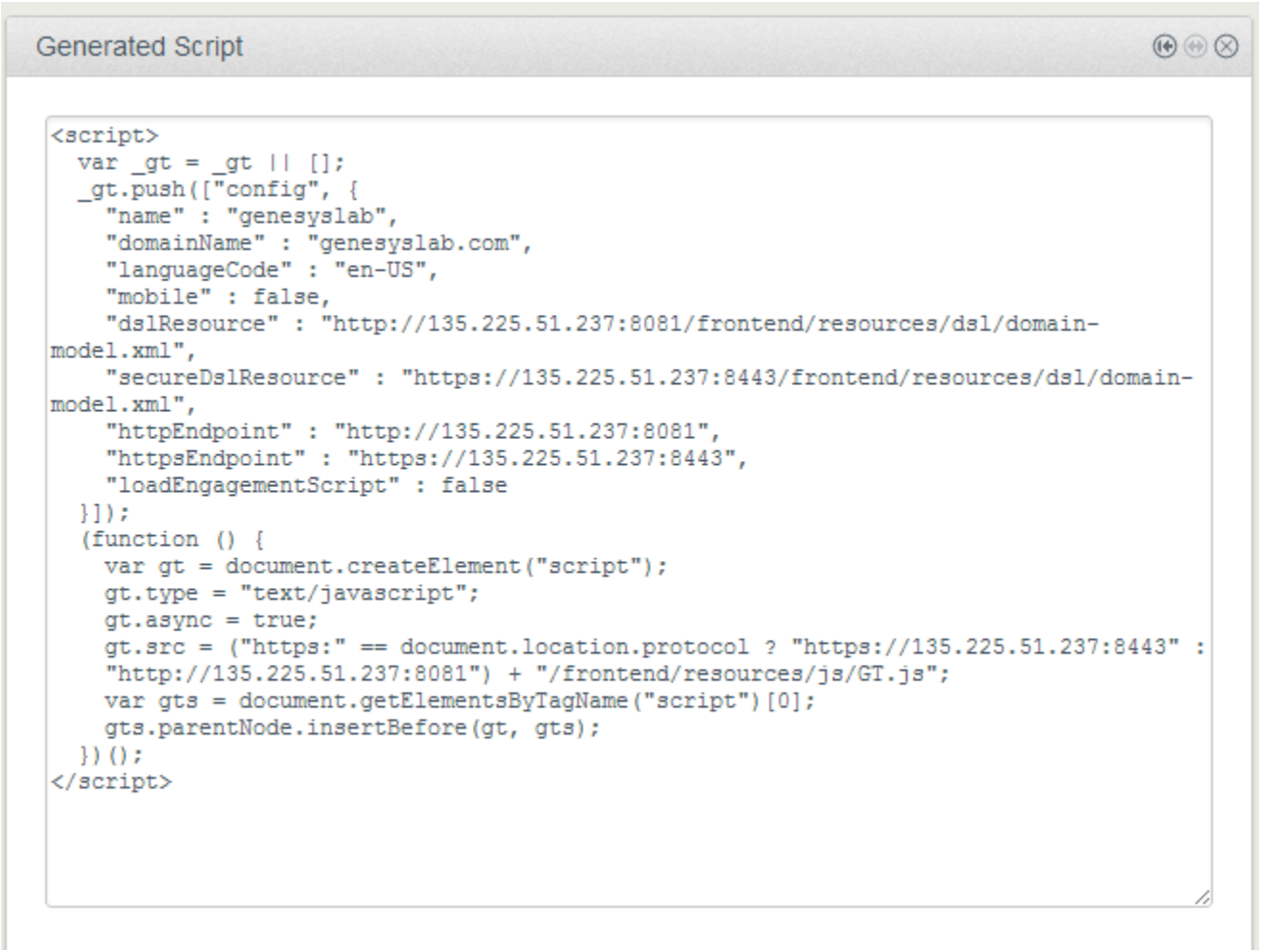
- Categories, used to create categories used for tagging browser events;
- Script Generator, used to create JavaScript Tracking code, to add to your webpages.

Script Generator

Script Generator creates the JavaScript Tracking code with mandatory information to enable tracking on your website. You enter configuration information and the tool generates a basic JavaScript code snippet. As specified in Enable Monitoring, you must add this JavaScript code snippet to your webpages, at the end of the <head> section of your HTML.

Generate a New Script

To create a new script, you must open the Script Generator interface, available in the CONFIGURATION > Script Generator menu of the Genesys Administrator Extension. Then, enter configuration information, as described in Enable Monitoring and click Generate.



Example of a script generated for the genesyslab application

The JavaScript Tracking code uses the `_gt` (Genesys Tracker) object which submits information asynchronously to ensure that information is submitted to the Web Engagement Frontend server before the users leave the webpages. The _gt (Genesys Tracker) object behaves like a FIFO (First In, First Out) collecting the API calls until the `GTC.js` library is ready to execute them. This enables you to add events to the queue if you wish to submit information with the Monitoring API, by using the `_gt.push()` method. See Monitoring JavaScript for further details on event submission. **Note:** Additional scripts should be placed at the end of the <body> section of your HTML page.

Customizable Parameters

The following table provide details about the possible fields available to customize your Tracking Script.

Parameter	Type	Default value	Mandatory	Description
name	String	no	yes	Name of the application; for instance, `genesyslab`.
domainName	String	<current domain name>	yes	Name of the domain where the cookie is stored; for instance, `genesyslab.com`.
dslResource	String	no	yes	DSL resource location via HTTP; for instance, `http://genesyslab.com:8081/frontend/resources/dsl/domain-model.xml`
secureDslResource	String	no	no	DSL resource location via HTTPS; for instance, `https://genesyslab.com:8443/frontend/resources/dsl/domain-model.xml`
httpEndpoint	String	no	yes	URL of the Frontend Server; for instance, `http://genesyslab.com:8081`
httpsEndpoint	String	no	no	Secured URL of the Frontend Server; for instance, `https://genesyslab.com:8443`
jQueryAutoDetect	BOOL	true	no	If true, the script will detect the loading of the jQuery library and will automatically load this library if there is no jQuery library available on the current page.
jQueryPath	String	`ajax.googleapis.com/ajax/libs/` jquery/1.8.3/jquery.min.js	no	URL for the jQuery library if `jQueryAutoDetect` is set to true; for instance, `genesyslab.com/js/`

Parameter	Type	Default value	Mandatory	Description
				`jquery.min.js`.
languageCode	String	en-US	no	Localisation tag for language and region; for instance, `en-US`.
mobile	BOOL	false	no	Set to true if the monitored website is designed for mobile devices.
debug	BOOL	false	no	Set to true to show the monitoring agent debug information in the console.
debugComet	BOOL	false	no	Set to true to show the cometd debug information in the console.
secureUserData	BOOL	true	no	Set to true to send all user information (SignIn and UserInfo events) via HTTPS.
preventIframeMonitoring	BOOL	false	no	Set to true to prevent monitoring of the generated system and business events if the monitoring agent is loaded in an iframe.
backendUrl	String	null	no	If set, all engagement traffic uses this endpoint. This URL is used by the callback, registration, and chat widgets. For instance, `http://serverName:9081/backend`. If not set, the endpoint for engagement traffic is read from the notification message.

Script Examples

This is the default script generated for the Genesyslab example.

```
<script>
    var _gt = _gt || [];
    _gt.push(['config', {
        'name':         'genesyslab',
        'domainName' :  'genesyslab.com',
        'dslResource':      'http://genesyslab.com:8081/frontend/resources/dsl/domain-
model.xml',
        'secureDslResource':    'https://genesyslab.com:8443/frontend/resources/dsl/domain-
model.xml',
        'httpEndpoint':             'http://genesyslab.com:8081',
        'httpsEndpoint':    'https://genesyslab.com:8443'
        }]);
    (function() {
        var gt = document.createElement('script'); gt.type = 'text/javascript'; gt.async =
true;
        gt.src = (  'https:' == document.location.protocol ? 'https://genesyslab.com' :
                    'http://genesyslab.com') + '/frontend/GTC.js';
        var gts = document.getElementsByTagName('script')[0]; gts.parentNode.insertBefore(gt,
gts);
    })();
</script>
```

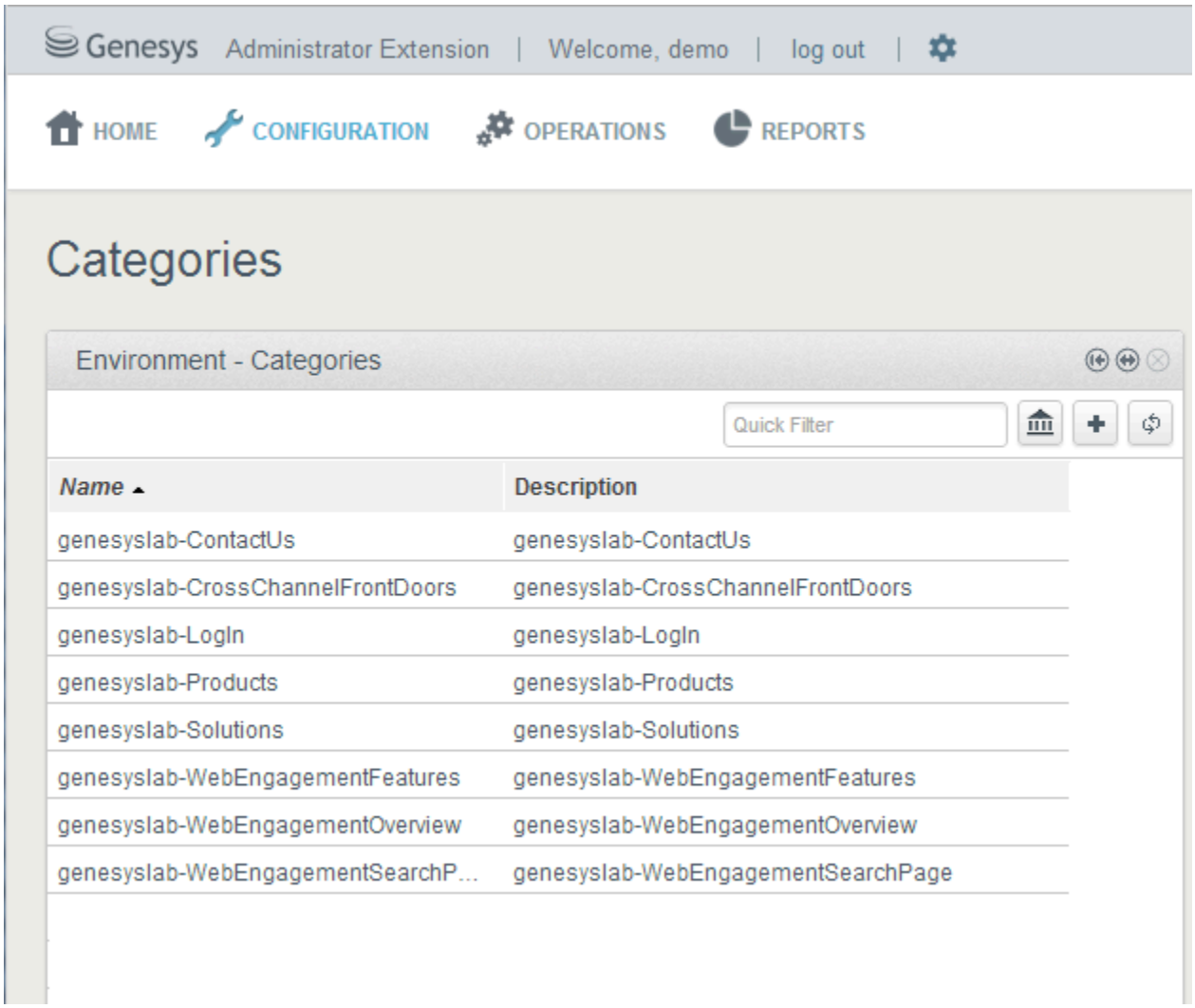
You can customize the generated script by adding fields (see Customizable Parameters). Below, the generated script for the Genesyslab example has been customized to include the `debug`, `jQueryAutoDetect`, `jQueryPath`, `languageCode`, `mobile`, `debugComet`, and `secureUserData` parameters.

```
<script>
    var _gt = _gt || [];
    _gt.push(['config', {
        'name':             'genesyslab',
        'domainName' :      'genesyslab.com',
        'dslResource':          'http://genesyslab.com:8081/frontend/resources/dsl/domain-
model.xml',
        'secureDslResource':        'https://genesyslab.com:8443/frontend/resources/dsl/
domain-model.xml',
        'httpEndpoint':                 'http://genesyslab.com:8081',
        'httpsEndpoint':            'https://genesyslab.com:8443',
        'debug':            true,
        'jQueryAutoDetect':         true,
        'jQueryPath':           'ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js'
        'languageCode':     'en-US',
        'mobile':           true,
        'debugComet':           true,
        'secureUserData':    true,
        }]);
    (function() {
        var gt = document.createElement('script'); gt.type = 'text/javascript'; gt.async =
true;
        gt.src = 'http://genesyslab.com:8070/tracker/GTC.js';
        var gts = document.getElementsByTagName('script')[0]; gts.parentNode.insertBefore(gt,
gts);
    })();
</script>
```

Categories

The `Categories` interface is a tool for creating the categories used in the simple model for

engagement. Each category is compliant with the category definition and include tags to define business information related to your website. To access the `Categories` interface, open Genesys Administrator Extension and navigate to `CONFIGURATION > Categories`.



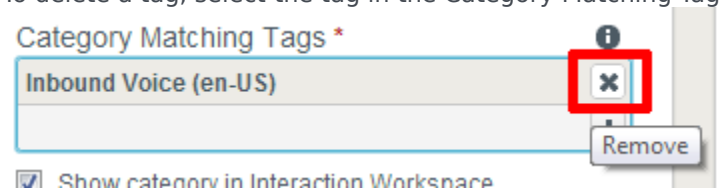
A list of Categories

Main Features

The main features are the following:

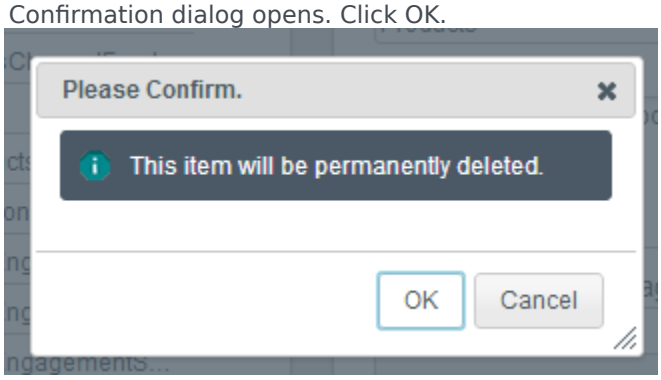
- Create categories and matching tags; instructions are available in Creating Categories
- Delete matching tags and categories:

- To delete a tag, select the tag in the Category Matching Tag section and click the x button.



click the Remove button.

- To delete a category, select the category in the list and click the Delete button. the Delete Confirmation dialog opens. Click OK.



Delete Confirmation.

Note: Categories are also displayed in the Configuration Manager. You should not edit or delete them through the Configuration Manager, to avoid synchronization issues with the Categories interface.

Regular Expressions in Tags

A regular expression is a sequence of elements. An element is either a word or expression inside quotes. Each search element can be preceded by exclusion '-'. The Search Request is case sensitive. A wildcard symbols '*' can be used inside or out the quotes. This symbol means any symbol in the regular expression.

Search Request Usage

A Search Request is applied to a text line. Each word included in the search can be surrounded in any number of symbols. All expressions inside the quotes will be copied to the resulting Regular Expression without any changes. The search in the line can be successful or not. The search is successful if all elements without exclusion symbol are included to the text line. Otherwise, the search is unsuccessful.

Search Request Patterns

The following table describes the patterns in search requests.

Search Options	Description
Search for all exact words in any order. *search query*	The result must include all the words. These words can be substrings attached to other words—for example, [Web-search query1].
Search for an exact word or phrase. "*search query*"	Use quotes to search for an exact word or set of words in a specific order without normal improvements such as spelling corrections and synonyms. This option is handy when searching for song lyrics or a line from literature—for example, ["imagine all the people"].
Exclude a word. *-query*	Add a dash (-) before a word to exclude all results that include that word. This is especially useful for synonyms like Jaguar the car brand and jaguar the animal. For example, [jaguar speed -car].
Include "fill in the blank". *query *query*	Use an asterisk (*) within a query as a placeholder for any terms. Use with quotation marks to find variations of that exact phrase or to remember words in the middle of a phrase. For example, ["a * saved is a * earned"].

Option

Option 'Mode' that is one of the input parameters equals 0, if the compilation function works as described above. 'Mode'= 1 switches to regime where all double-quotes considered as a regular symbols. It means that they no more serve as brackets of substrings to be searched. Checking of even number of double-quotes is also removed.

Compilation Result

The compilation procedure should transform the Search Request to the equivalent Regular Expression. It returns an empty string, if search request is incorrect.

Using the Plug-in for IW

The Web Engagement Plug-in for Interaction Workspace extends the interfaces of the Interaction Workspace to support Genesys Web Engagement information available in interactions.

Plug-in Description

Once installed, as detailed here, the Genesys Web Engagement Plug-in for Interaction Workspace enables the Interaction Workspace to support Web Engagement Interactions created when actionable events are submitted to the Web Engagement Backend Server. The Web Engagement Plug-in integrates in the following views of the Interaction Workspace:

1. Interaction Case Data
2. Contact Interaction History Details
3. Web Activity Tab

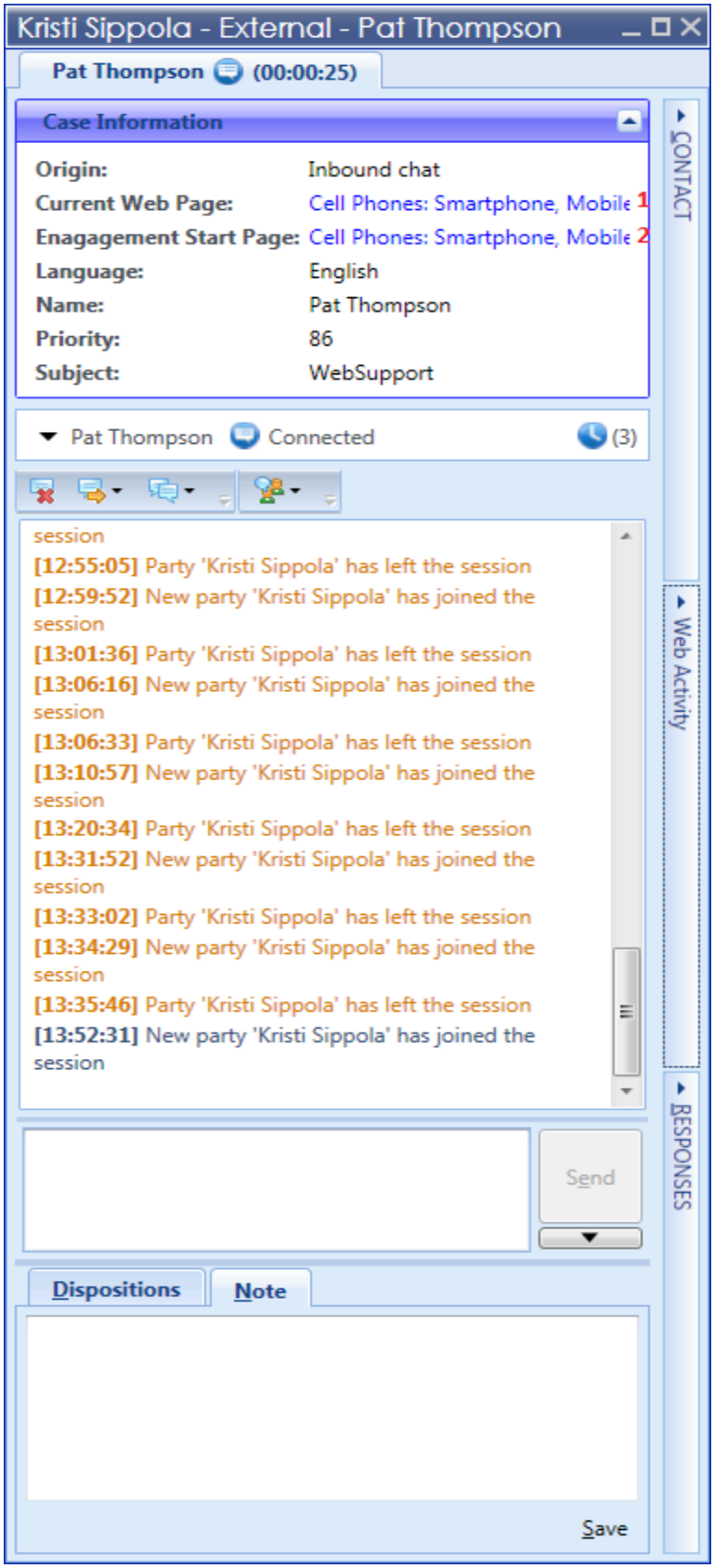
You can also customize views to integrate your own business information.

Default Customized Views

Interaction Case Data

The Web Engagement Plug-in adds two additional information fields to the default Case Data region:

1. Current web visitor browsing page - updated during visitor browsing
2. Engagement start web page - locked for current interaction


Kristi Sippola - External - Pat Thompson
Pat Thompson (00:00:25)
Case Information
Origin: Inbound chat
Current Web Page: Cell Phones: Smartphone, Mobile 1
Enagagement Start Page: Cell Phones: Smartphone, Mobile 2
Language: English
Name: Pat Thompson
Priority: 86
Subject: WebSupport
Pat Thompson Connected (3)
session
[12:55:05] Party 'Kristi Sippola' has left the session
[12:59:52] New party 'Kristi Sippola' has joined the session
[13:01:36] Party 'Kristi Sippola' has left the session
[13:06:16] New party 'Kristi Sippola' has joined the session
[13:06:33] Party 'Kristi Sippola' has left the session
[13:10:57] New party 'Kristi Sippola' has joined the session
[13:20:34] Party 'Kristi Sippola' has left the session
[13:31:52] New party 'Kristi Sippola' has joined the session
[13:33:02] Party 'Kristi Sippola' has left the session
[13:34:29] New party 'Kristi Sippola' has joined the session
[13:35:46] Party 'Kristi Sippola' has left the session
[13:52:31] New party 'Kristi Sippola' has joined the session
Send
Dispositions
Note
Save
CONTACT
Web Activity
RESPONSES

Both fields can be managed as standard Interaction Workspace Case Data fields (see Customer Case Interaction Workspace Deployment Guide (Customer Case)). Web Engagement fields are using the following Case Data business attribute values:

1. Current Web Page
2. Engagement Start Page

Contact Interaction History Details

The Web Engagement Plug-in provides the browsing history in this section for specific interactions of type "webengagement". The browsing history is displayed for the whole visit associated with the selected Web Engagement interaction.
This presentation can be configured in the Module Configuration file (Genesyslab.Desktop.Modules.WebEngagement.module-config) as a special activity type with the fixed name "History".

Web Activity Tab

The Web Engagement Plug-in provides the on-going web visit or the full browsing history information about the current visit. The Web Activity tab (extension) is extended with or through the following visual divisions:

1. Tags Panel
2. History View
3. Details View

The Tags Panel contain the Web Engagement Categories which are created by Genesys Web Engagement Plug-in for Administrator Extension. The tags (or categories) are used for additional filtration in the History View objects (events or pages). The current selection in this panel displays details, including Web Engagement data, in the History and Detail views.