NASTEL
Driving Business Transaction Performance ®

M6-CAL 100.002.1

AutoPilot® M6 Plug-in for Calendar Integration Installation and User's Guide

Version 1.0

Published by:

Title: **AutoPilot M6 Plug-in for Calendar Integration Installation and User's Guide**

Document release date: **May 2022**

Nastel Document Number: **M6-CAL 100.002.1**

Table of Contents

Figures

Tables

This Page Intentionally Left Blank

Chapter 1: Introduction

Welcome to the *Nastel AutoPilot M6 Calendar Integration Plugin User's Guide*. This guide explains how to deploy and use of the Calendar Plugin. Please review this guide carefully before using the product. Nastel AutoPilot M6 Calendar Integration Plugin will hereinafter be identified as the Calendar Plugin.

1.1 How this Guide is Organized

Chapter 1:   Introductory information about this document.

Chapter 2:   Provides instructions for new installations of the Calendar Plugin.

Chapter 3:   Explains how to deploy and use the Calendar Plugin.

Appendix A: Detailed list of all reference information.

Appendix B: Conventions used in Nastel documentation.

Glossary: Listing of unique and common acronyms, words and definitions.

1.2 History of this Document

Table 1-1. Document History			
Release Date	**Document Number**	**Version**	**Summary**
April 2013	M6-CAL 100.001	1.0	Initial release
August 2017	M6-CAL 100.002	1.0	Update Nastel's phone numbers and street address
May 2022	M6-CAL 100.002.1	1.0	Changed title to *AutoPilot M6 Plug-in for Calendar Integration Installation and User's Guide*

1.2.1   User Feedback

Nastel encourages all users of M6-WMQ to submit comments, suggestions, corrections and recommendations for improvement for all M6-WMQ documentation. Please send your comments to support@nastel.com. You will receive a written response, along with status of any proposed change, update, or correction.

1.3   Related Documents

Complete listings of documents related to AutoPilot M6 can be found in Appendix A.

1.4   Release Notes

See README.HTM file on installation media and root installation directory.

1.5   Intended Audience

This document is intended for personnel installing and using AutoPilot M6 Calendar Integration Plugin. The installer should be familiar with:

- AutoPilot M6 V6.5
- Target operating system environment
- The installer may need administrative privileges for the target platform.

1.6   Terms and Abbreviations

A list of terms and abbreviations used in this document is located in the Glossary.

1.7   Technical Support

If you need additional technical support, you can contact Nastel by telephone or e-mail. To contact Nastel technical support by telephone, call **800-963-9822 ext. 1**, if you are calling from outside the United States dial **001-516-801-2100.** To contact Nastel technical support by e-mail, send a message to support@nastel.com. You can also contact Nastel support via the support website. Contact your AutoPilot M6 Administrator for access information. To access the Nastel automated support system (user ID and Password are required) go to: http://support.nastel.com/. Contact your local AutoPilot M6 Administrator for further information.

1.8   Conventions

Refer to Appendix B for conventions used in this guide.

Chapter 2: Installation

This chapter provides instructions for a typical installation and setup requirements for the AutoPilot M6 Calendar Integration Plugin.

2.1   Before Installation

2.1.1   Technical Documents

Prior to installation you should review all text files and installation procedures. You should print, as needed, all of the installation-related materials to give yourself quick access to any required information during any installation or migration procedures. Additional sets of printed documents are available from your Nastel representative or Nastel Support.

2.1.2   Installation Recommendations

Nastel recommends that you observe the following update sequence when updating AutoPilot with patches, updates, and service packs:

- Machines running the Domain Server
- Machines running the managed nodes.

2.1.3   Licensing

A copy of the standard Licensing Agreement is imbedded in the installation software and is provided on the Nastel Resource Center. The formal licensing agreement has been furnished in the purchase agreement package.

2.1.4   Download

Download the Calendar Plug-in, `AP_GOOGLE_CAL-<version_number>.pkg` (for example, `AP_GOOGLE_CAL-1.0.4.pkg)`, from the Nastel Resource Center, or copy from your installation media.

2.2   Installing the Plug-in

2.2.1   Installing from a Command Prompt

1. Save your work and log off AutoPilot M6.

2. Stop the managed nodes and/or Domain Servers that will be updated as specified in the *AutoPilot M6 User's Guide*.

3. Copy **`AP_GOOGLE_CAL-<version_number>.pkg`** into the `[AUTOPILOT_HOME]\updates` `directory`.

4. Navigate to the **`[AUTOPILOT_HOME]\bin`** directory.

5. At the command prompt run:

    `[AUTOPILOT_HOME]\bin\pkgman ..\updates\ AP_GOOGLE_CAL -<version_number>.pkg`
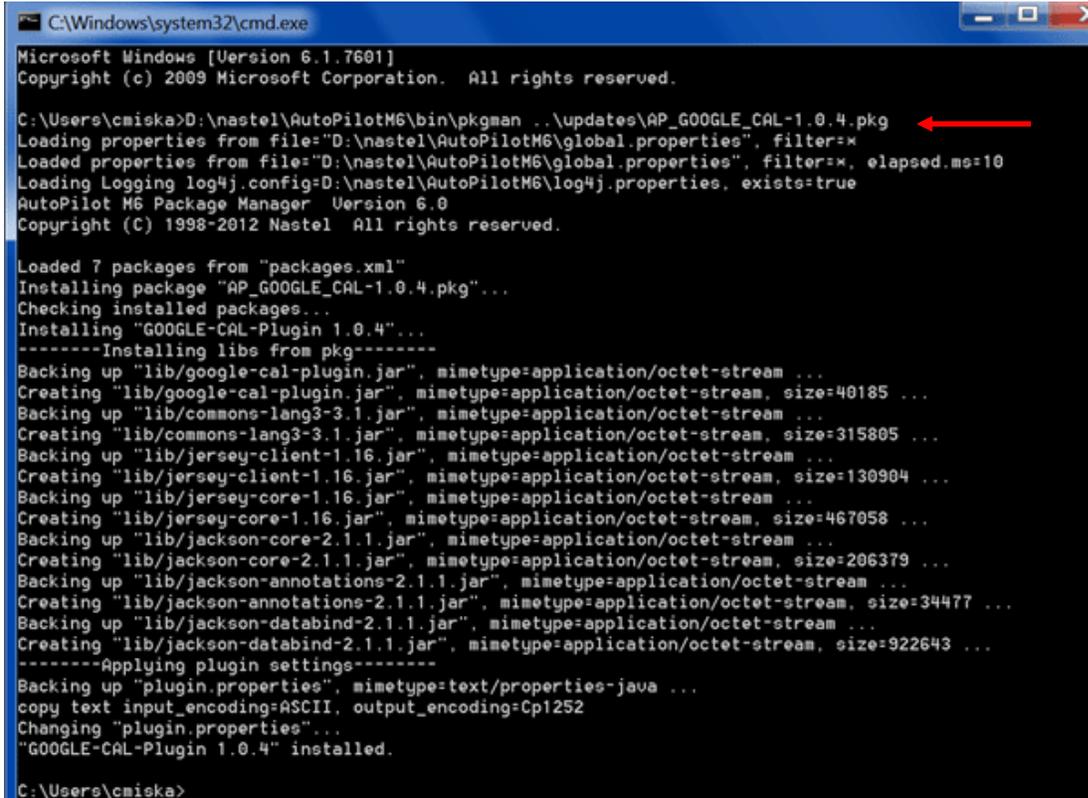
Figure 2-1. File Path Input Screen

6. Verify plug-in installation by running the command:

    `[AUTOPILOT_HOME]\bin\pkgman –info`.

    The details of the Package Manager are listed. Verify `GOOGLE-CAL-Plugin(AP_GOOGLE_CAL-<version_number>.pkg)` is listed. Make sure there are no errors posted at the bottom of the screen.
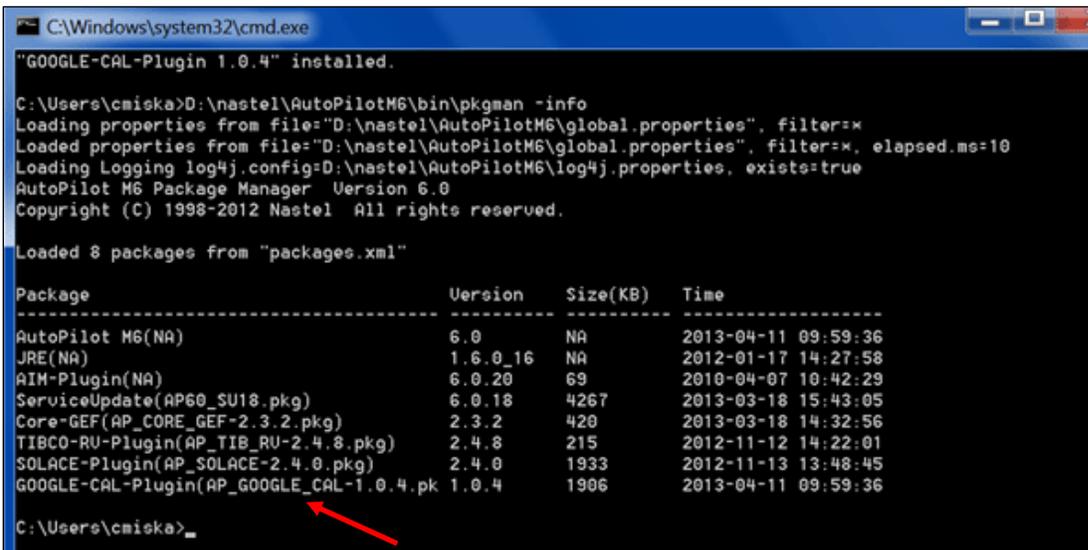
Figure 2-2. Verify Installation Screen

7. Verify that the **jar** files have been copied into the **lib** directory. In this example change directory to **lib** then run the **dir** command.

Figure 2-3. Verify JAR Files

You should see the following jar files listed.

- `commons-lang3-3.1.jar`

- `google-cal-plugin.jar`

- `jersey-core-1.16.jar`

- `jersey-client-1.16.jar`

- `jackson-databind-2.1.1.jar`

- `jackson-core-2.1.1.jar`

- `jackson-annotations-2.1.1.jar`

2.1.2 Installing from a Windows Environment

1. Save your work and log off AutoPilot M6.

2. Stop the managed nodes and/or Domain Servers that will be updated as specified in the *AutoPilot M6 User's Guide*.

3. At your PC navigate to **Start > Nastel AutoPilot M6 > M6 Product Maintenance** to display the following window.
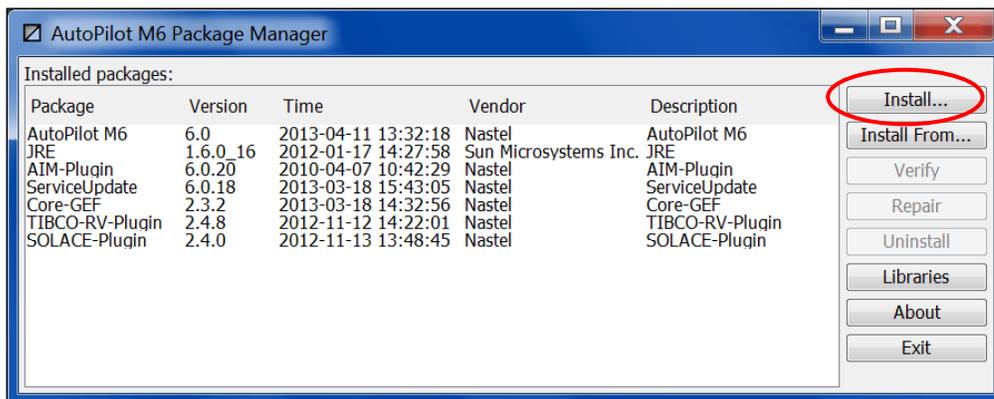
Figure 2-4. AutoPilot M6 Package Manager Window

4. Click **Install** to open the *Select Package to Install* window.
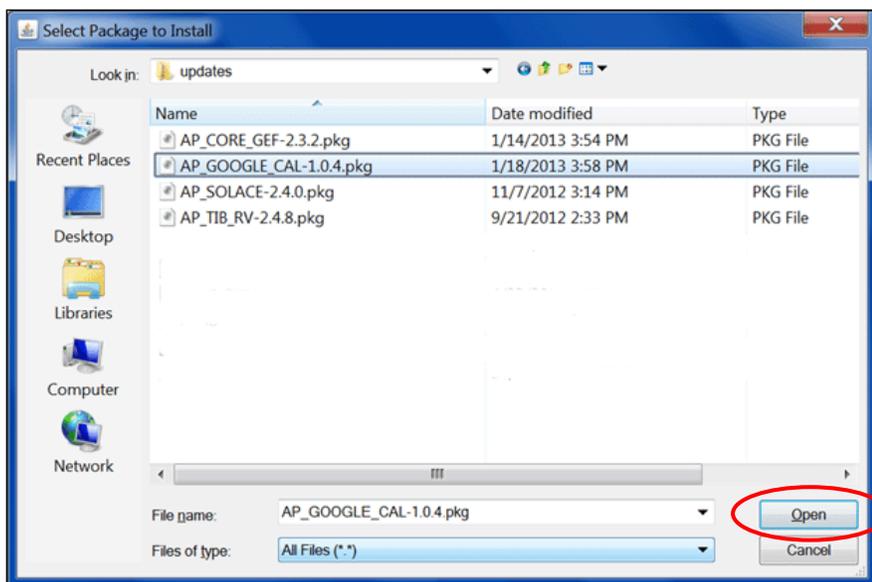
Figure 2-5. Select Package to Install Window

5. Select **AP_GOOGLE_CAL-<*version_number*>.pkg** and click **Open** to install. A confirmation message is displayed when installation is complete.
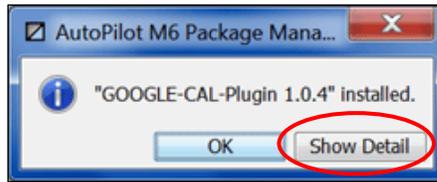
Figure 2-6. Installation Complete

6. Click **Show Detail** to verify installation and to see the following **jar** files are copied into the **lib** directory.

- commons-lang3-3.1.jar
- google-cal-plugin.jar
- jersey-core-1.16.jar
- jersey-client-1.16.jar
- jackson-databind-2.1.1.jar
- jackson-core-2.1.1.jar
- jackson-annotations-2.1.1.jar
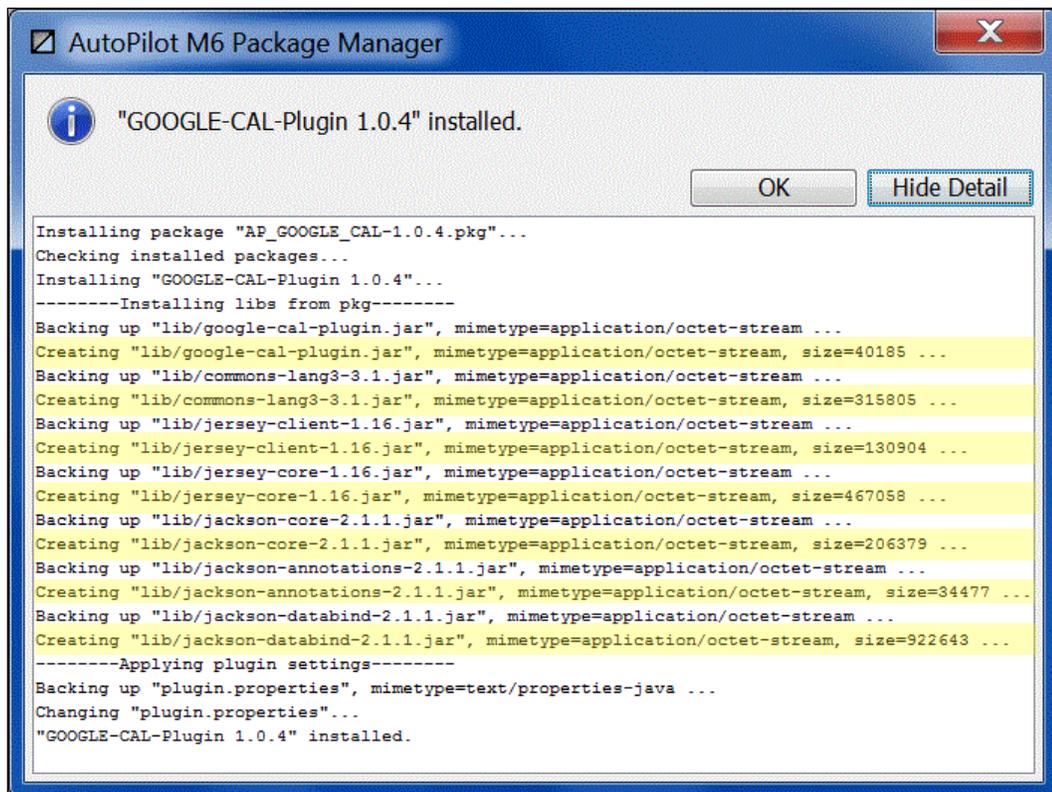
Figure 2-7. Installation Details

7. Click **OK**.

This page intentionally left blank.

Chapter 3: Using Calendar Integration Plugin

The Calendar Plugin allows AutoPilot activities to be triggered by calendar events. The expert periodically polls a specified Google calendar for currently occurring events, publishing the details of events that are currently occurring. The expert accounts for the difference in time-zones between the calendar and the CEP server.

You must have a Google account to use the Calendar Plugin. The facts published from the Google calendar are:

- Busy – true or false
- Start time
- End time
- Color ID
- Description
- Location.

To start using the Calendar Plugin do the following:

1. Create and deploy calendar expert (section 3.1).
2. Link Google calendar to the AutoPilot expert (section 3.2).

3.1   Deploying the Calendar Plugin

The following procedure is used to configure the Calendar expert within M6 managed node.

1. Open your AutoPilot Console.
2. Right-click on the managed node where you want to deploy the Calendar.
3. Click **Deploy Expert > GoogleCal > CurrentEvents**. The *Create Current Events* dialogue box (Figure 3-2) is displayed.
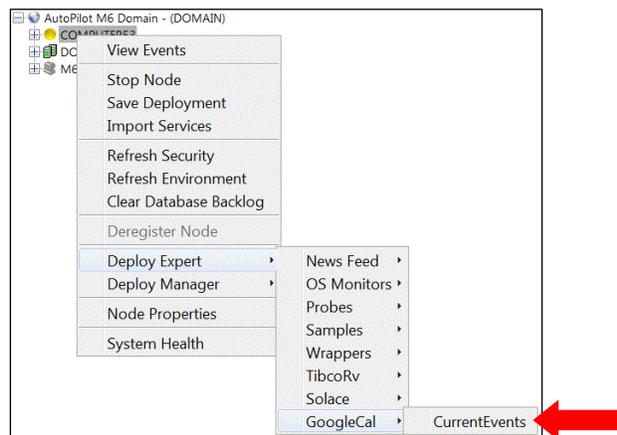
Figure 3-1. Deploy Calendar Expert

4. The *General* tab is displayed. It is recommended that you update all general properties to define your expert. At a minimum apply a definitive name to your expert.

    You can leave the defaults for the other options and skip to .
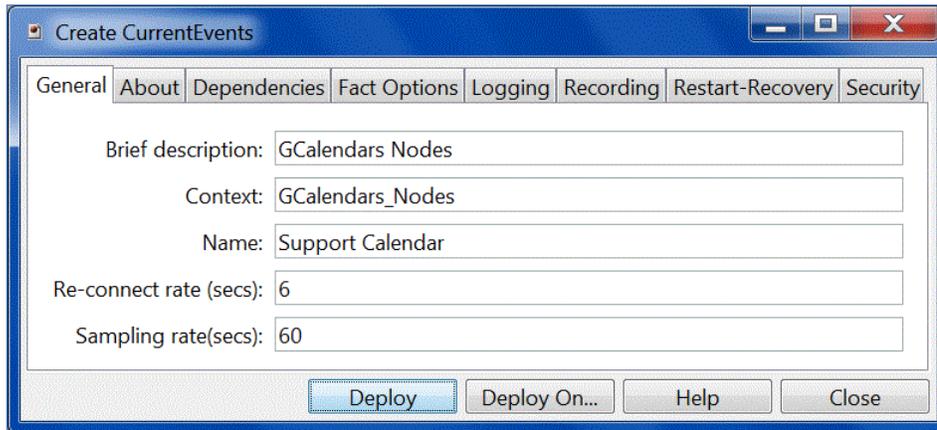
Figure 3-2. Create CurrentEvents – General

Table 3-1. Common Properties: General	
Property	**Description**
Brief description	A short, user defined description of the service. The default is the subject expert name (example: GCalendars Node).
Context	A user defined category that will be registered with the domain server. The default is: GCalendars_Node.
Name	Name that uniquely identifies the service in the domain. The default name is system assigned with the word service and twelve random digits. (Example: Service_123456789012). You can change the name to anything that suits your needs.
Re-connection rate(sec)	Rate in seconds at which the expert will attempt to reconnect to the target server, in seconds if connection failed. Default is **6**.
Sampling rate(sec)	Rate in seconds of fact samplings interval in seconds. Default is **6**.

5. Click the *About* tab. These parameters are common to all experts and cannot be edited.
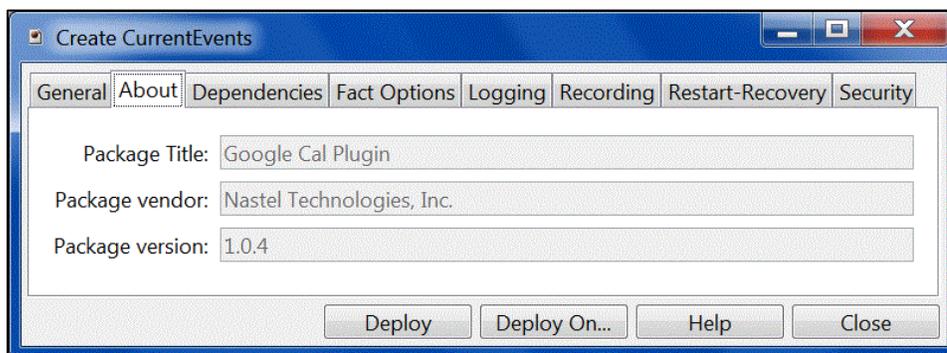
Figure 3-3. Create CurrentEvents – About

Table 3-2. Common Properties: About

Property	Description
Package Title	Implementation title of the source package.
Package vendor	Name of implementation vendor.
Package version	Package version as assigned by the vendor.

6. Click the *Dependencies* tab. Edit properties described in the table below, as required. These parameters are common to all experts.

Figure 3-4. Create CurrentEvents – Dependencies

Table 3-3. Common Properties: Dependencies

Property	Description
Platform dependencies	Dependencies on operating system platforms, which is expressed in a comma separated list.
Service dependencies	Dependencies on other services, which is expressed in a comma separated list.

7. Click the *Fact Options* tab. Edit properties described in the table below, as required. These parameters are common to all experts.

Figure 3-5. Create CurrentEvents – Fact Options

Table 3-4. Common Properties: Fact Options

Property	Description
Exclude Expire Filter (regexp)	A regular expression filter to exclude certain facts from being written to the database.
Exclude Fact Filters	Comma separated list of fact paths to exclude during publishing.
Expire facts(ms)	Automatically expires facts that have not been updated in the specified time (ms).
Fact History Size	Automatically maintains specified number of samples for each published fact in memory.
Fact History Time (ms)	Automatically maintain fact history not exceeding specified time in milliseconds.
Include Expire Filter (regexp)	A regular expression filter to include certain facts being written to the database.
Include Fact Filters	Comma separated list of fact paths to include during publishing. For example: *SYSTEM*, *FactName*
Lock Fact History	Enables/disables history collection after accumulating the first history batch up to **Fact History Time** or **Fact History Size** which ever limit is reached first. If disabled newer history samples replace older on a rolling basis.

8. Click the *Logging* tab. Edit properties described in the table below, as required. These parameters are common to all experts.
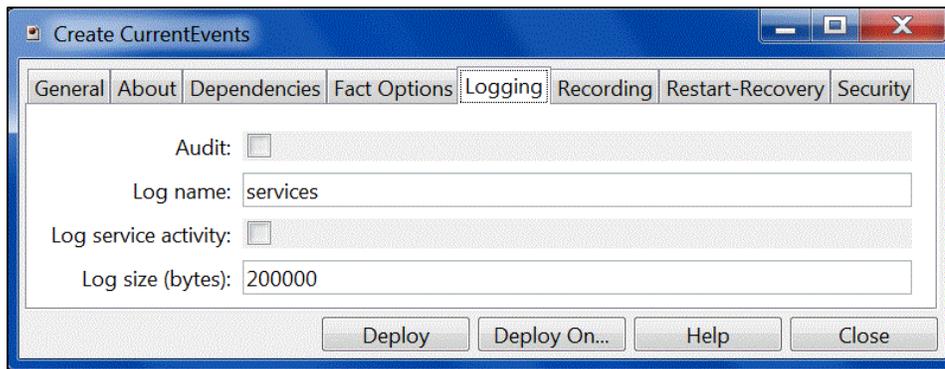
Figure 3-6. Create CurrentEvents – Logging

Table 3-5. Common Properties: Logging

Property	Description
Audit	Enable/disable service audit trace.
Log name	Log name associated with the service.
Log service activity	Enable/disable service activity trace.
Log size (bytes)	Enter log file size if the activity is enabled. Default value is 200000.

9. Click the *Recording* tab. Edit properties as defined in the table below, as required. These parameters are common to all experts.
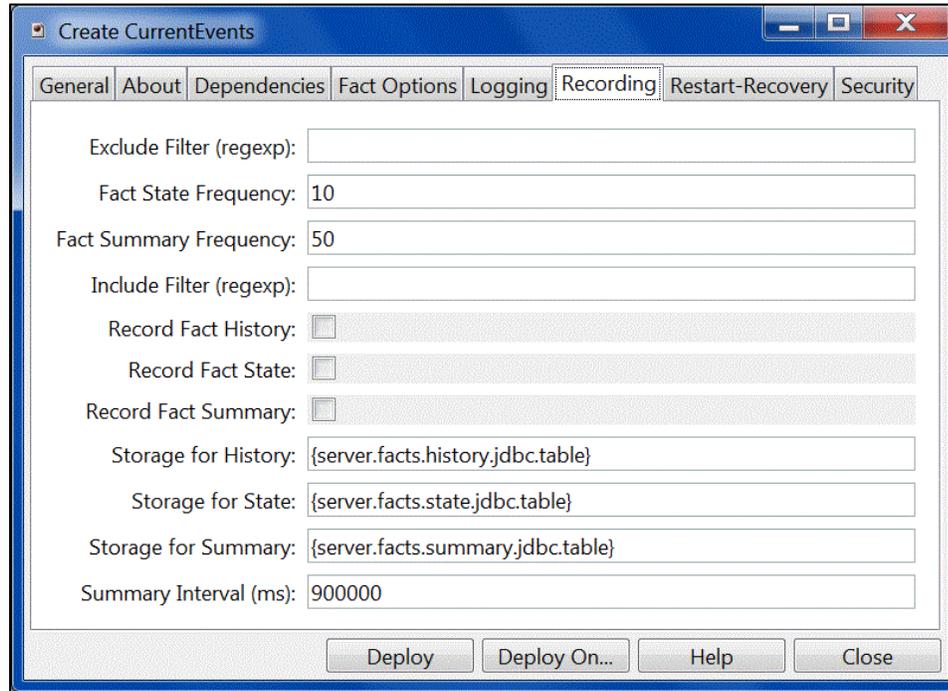
Figure 3-7. Create CurrentEvents – Recording

Table 3-6. Common Properties: Recording	
Property	**Description**
Exclude Filter (regexp)	A regular expression filter to exclude certain facts from being written to the database. Facts have the format `expert\class\instance\leaf=value` such as in the example `Servers\Linux\Serv7\processes=40`.
Fact State Frequency	If **Record Fact State** is enabled, the value entered here specifies how often the Fact State is updated.
Fact Summary Frequency	If **Record Fact Summary** is enabled, used to write an intermediate summary record every X[th] update to the fact during the Summary Interval. In this example, every 50th update to the fact an intermediate summary record is recorded. This is done to avoid waiting 15 minutes for a summary record to appear in the summary table.
Include Filter (regexp)	A regular expression filter to include certain facts being written to the database. Same format as described for the exclude filter.
Record Fact History	If enabled, records every fact change into the History database. The exclude/include filters are respected. To define database tables and set AutoPilot options, refer to *AutoPilot M6 User's Guide*, section 4.5.4.1.
Record Fact State	If enabled, records the last value published (current state) into the state database and restores that value when the CEP Server is stopped and restarted. The exclude/include filters are respected. To define database tables and set AutoPilot options, refer to *AutoPilot M6 User's Guide*, section 4.5.4.1.
Record Fact Summary	If enabled, records summary record at the interval designated in the **Summary Interval (ms)** field into the Summary database. The exclude/include filters are respected. To define database tables and set AutoPilot options, refer to *AutoPilot M6 User's Guide with Service Update 8,* section 4.5.4.1.
Storage for History	Database table where the Fact History data is stored.

Table 3-6. Common Properties: Recording

Property	Description
Storage for State	Database table where the Fact State data is stored.
Storage for Summary	Database table where the Fact Summary data is stored.
Summary Interval (ms)	If **Record Fact Summary** is enabled, designates the interval of time in ms for which baseline numbers for each numeric fact are computed. Summary Interval is only in affect when CEP instance is running in record mode (ATPNODE –record). Default 900000 is 15 minutes, which means maintain a baseline of statistics for each numeric fact for a period of 15 minutes and write a record to the database. At the end of interval fact statistics is reset and the baseline collection starts again.

10. Click the *Restart-Recovery* tab. Edit properties as defined in the table below, as required. These parameters are common to all experts.
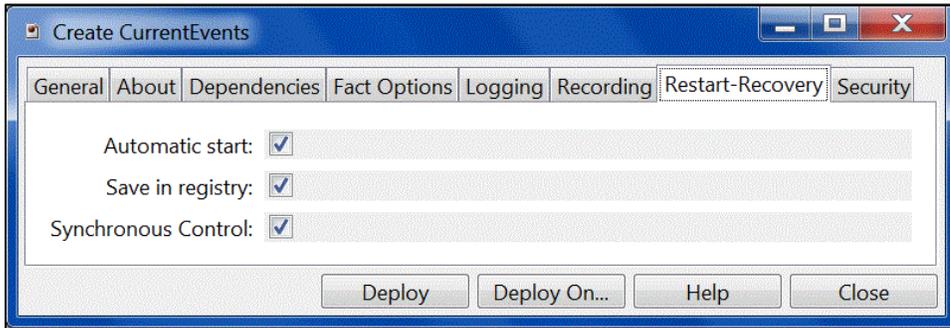
Figure 3-8. Create CurrentEvents – Restart-Recovery

Table 3-7. Common Properties: Restart-Recovery

Property	Description
Automatic start	Check to enable automatic start.
Save in registry	Check to enable saving persistent services in registry .xml file.
Synchronous Control	Check to enable synchronous service initiation.

11. Click the *Security* tab. Edit properties as defined in the table below, as required. These parameters are common to all experts.
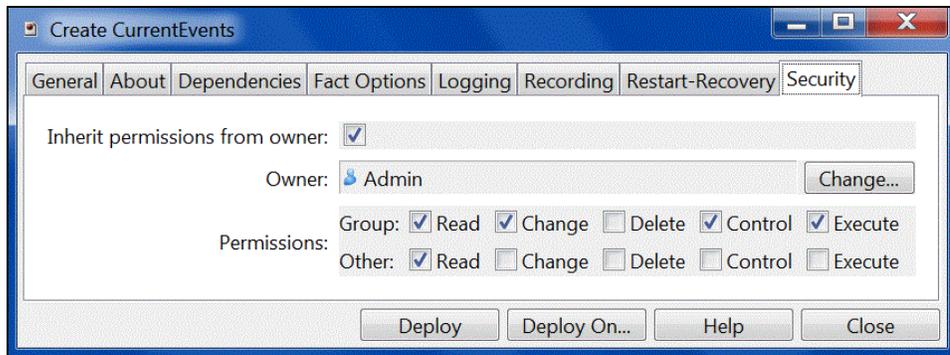
Figure 3-9. Create CurrentEvents – Security

Table 3-8. Common Properties: Security

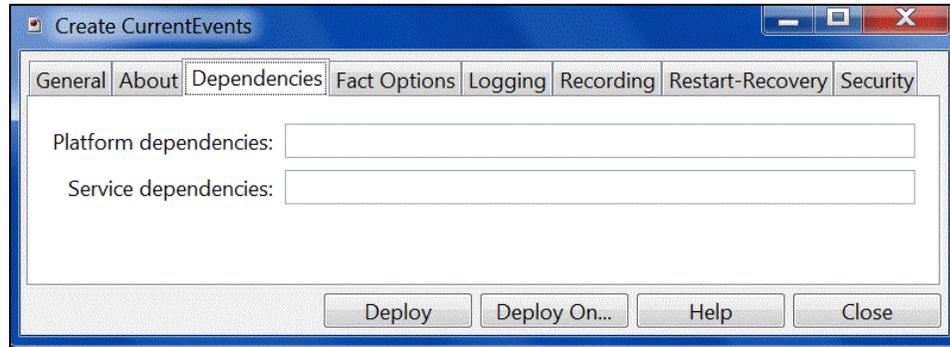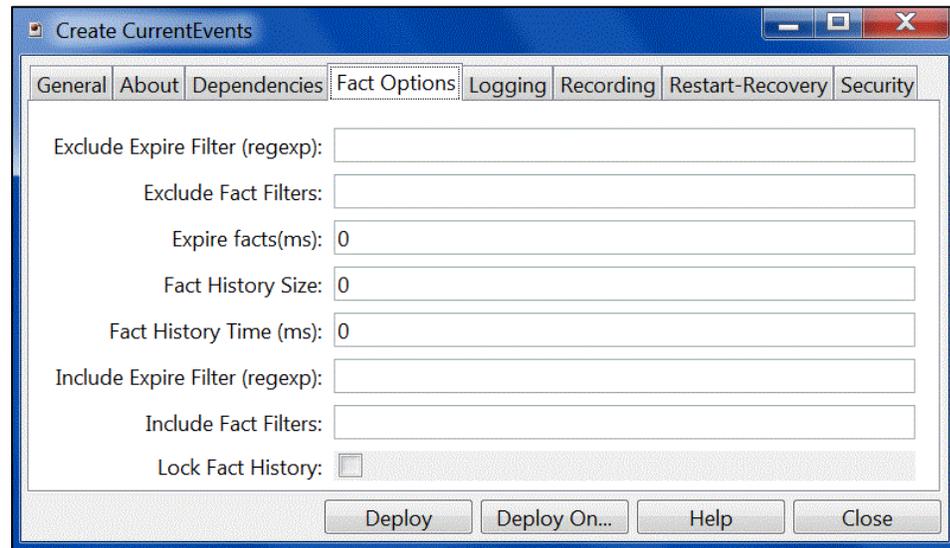| Property	Description	
Inherit permissions from owner	Enable/disable inherit permission from owner's permission masks.	
Owner	User that owns the object.	
Permissions	Permissions for users in same group and in other groups. Enable/disable as required.	
	Group	**Other**
Read	Group members may read/view attributes of an object.	Other users may read/view attributes of an object.
Change	Group members may change the attributes of an object.	Other users may change the attributes of an object.
Delete	Group members may delete the object.	Other users may delete the object.
Control	Group members may execute control actions such as start, stop, and disable.	Other users may execute control actions such as start, stop, and disable.
Execute	Group members may execute operational commands on the object.	Other users may execute operational commands on the object.

12. Click **Deploy**. A confirmation window is displayed. Click **Yes** to proceed with deployment.
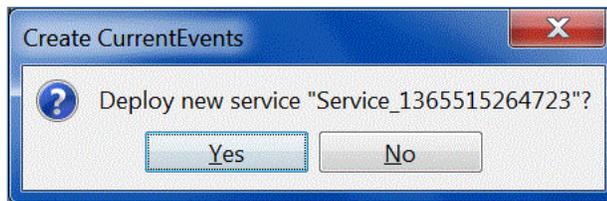
Figure 3-10. Deploy New Service

13. The deployment message will confirm the name and location of the expert. Click **OK**.
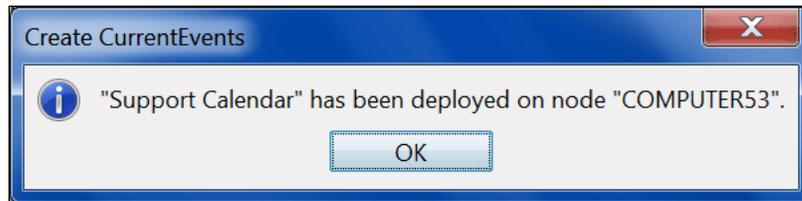
Figure 3-11. Service Deployed

14. The expert is now listed on your domain tree.
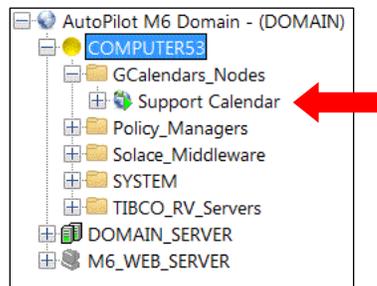
Figure 3-12. Deployed Expert Listed

3.2   Linking Your Google Calendar to AutoPilot

To link your Google calendar to the AutoPilot expert you just created, you need the following information:

- **Google Calendar ID** – This identifies the specific calendar to be tracked.
- **Authorization Code** – This is a security token representing consent by the Google user to share calendar information with the expert.

1. Log on to your Google calendar.
2. On the left side of the page, click to down-arrow next to **My calendars** to display a drop-down menu and select **Settings**.
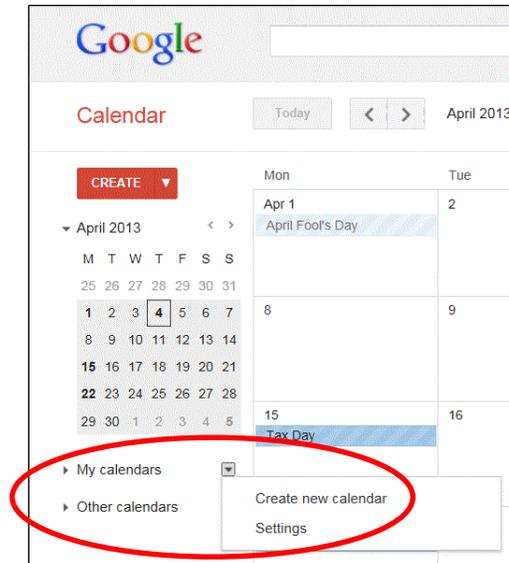
Figure 3-13. Google Calendar Page

3. If you have more than one calendar listed, click on it to display the *Calendar Details* page. Scroll down to the **Calendar Address** section and copy the **Calendar ID**.
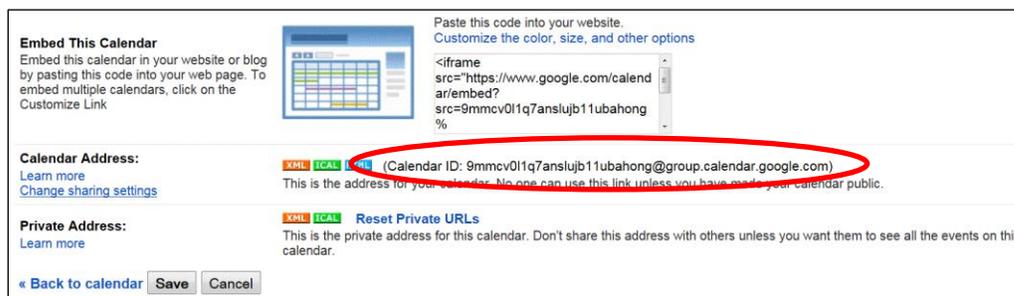
Figure 3-14. Calendar ID

4. Expand the expert, right click **GCalendars_Registry** to display a menu, and select **Add Calendar**.
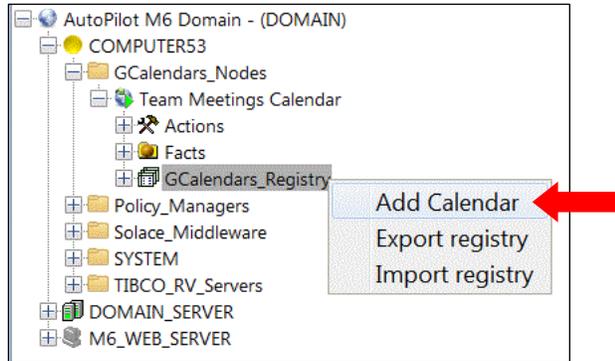
Figure 3-15. Add Calendar

5. Paste the Calendar ID you copied in Step 3 in the **Calendar ID** field.
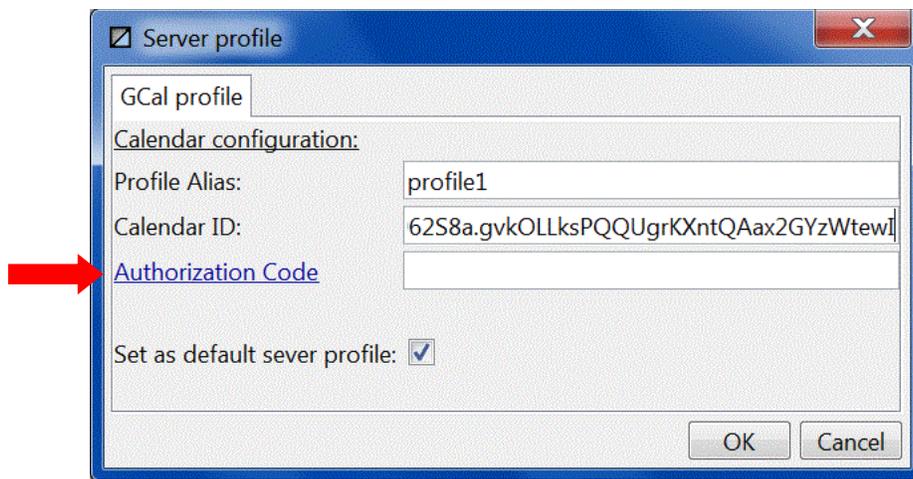
Figure 3-16. GCal Profile

6. Click **Authorization Code** to go to the Google Calendars. Click **Allow access**.
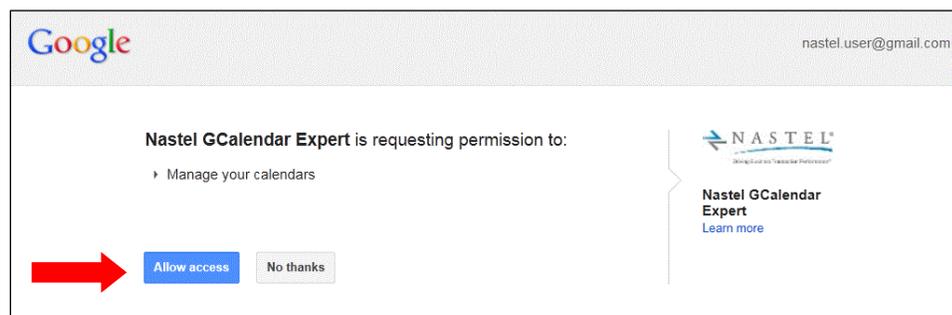
Figure 2-17. Google Calendar Permission

7. Copy the authorization code.
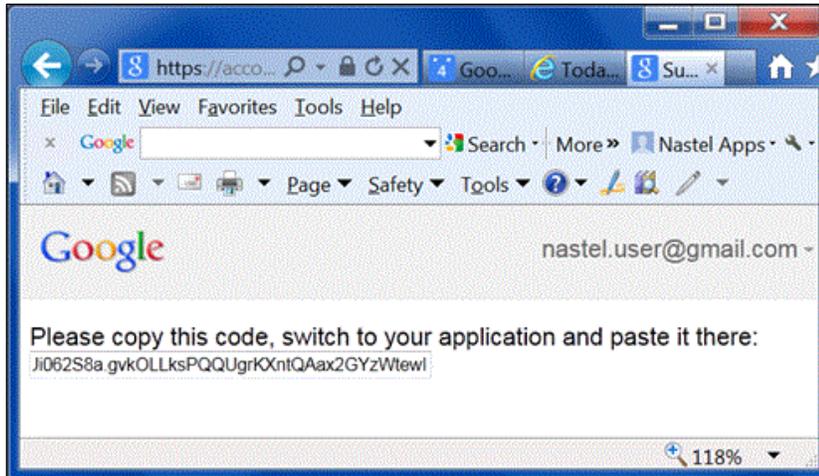
Figure 3-18. Google Calendar Authorization Code

8. Paste the authorization code in the GCal Profile screen and click **OK**. Your Google calendar is now linked to AutoPilot.
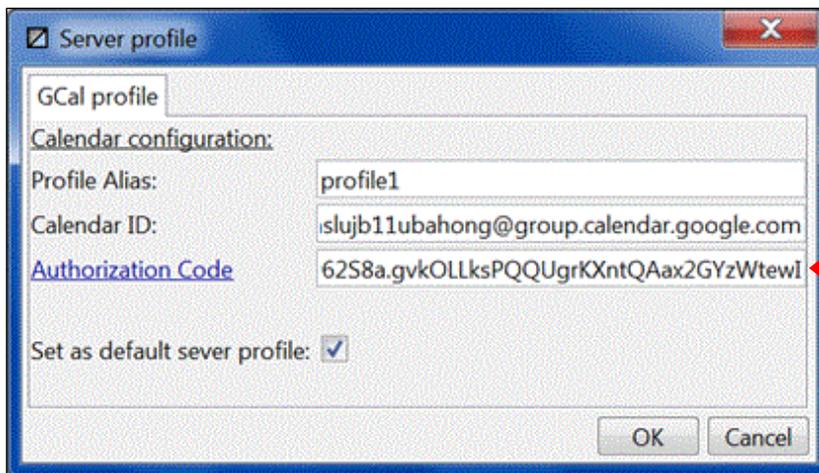
Figure 3-19. Add Authorization Code

3.3   Examples – Using the Calendar Expert

3.3.1   Facts Published to AutoPilot

After you have linked the Google Calendar to AutoPilot, facts will be published to AutoPilot. In the example below, Google Calendar has two events scheduled on April 10, 2013; "Scheduled Maintenance" and "Scheduled Maintenance Invoicing".
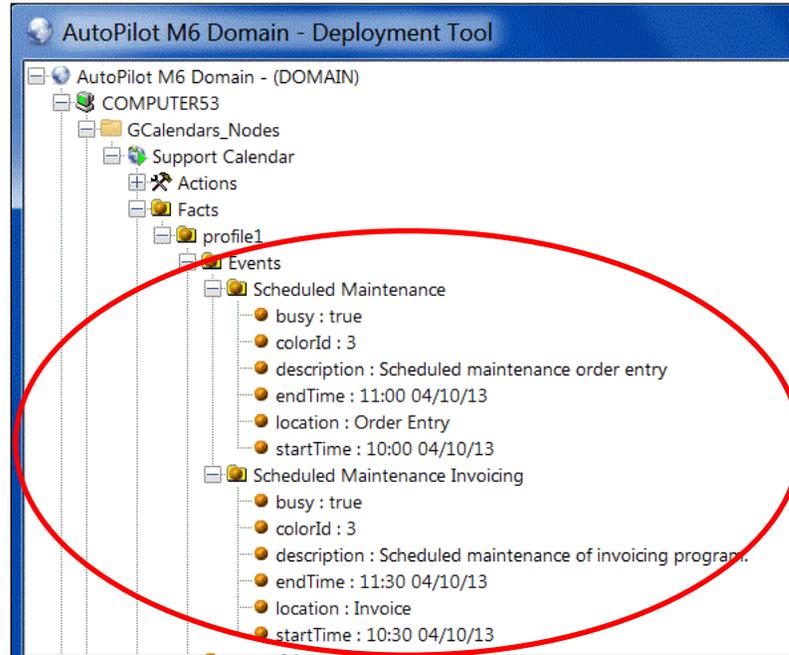
Figure 3-20. Calendar Facts

3.3.2   Business View Policy

The next step is to create a policy to trigger an alert in the Business View. This can be accomplished by using the Sensor Wizard. *(Refer the AutoPilot M6 User's Guide.)* After creating the policy, the calendar facts create schedule alerts as seen in the figure below.
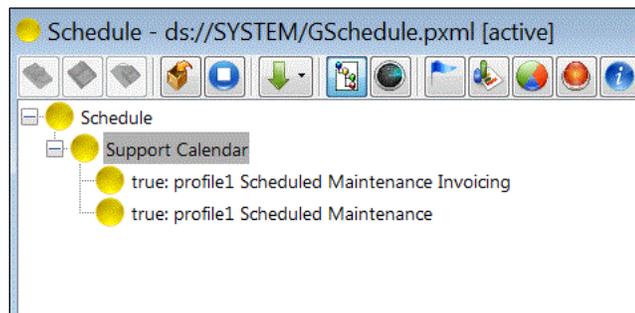
Figure 3-21. Schedule Alerts

This page intentionally left blank.

Appendix A: References

A.1   Nastel Documentation

The following table provides a list of reference information required for using the AutoPilot M6 Calendar Integration Plug-in.

Table A-1. Nastel Documentation	
Document Number (or higher)	**Title**
M6/INS 600.007	*Nastel AutoPilot M6 Installation Guide*
M6/USR 600.011	*Nastel AutoPilot M6 User's Guide*
M6/DSB 610.003	*Nastel AutoPilot M6 Business Dashboard*

This page intentionally left blank.

Appendix B: Conventions

B.1   Typographical Conventions

Table B-1. Typographical Conventions	
Convention	**Description**
Blue/Underlined	Used to identify links to referenced material or websites. Example: support@nastel.com
Bold Print	Used to identify topical headings, glossary entries, and toggles or buttons used in procedural steps. Example: Click **EXIT**.
Italic Print	Used to identify titles, menus, screen names, user inputs, or other category.
`Monospaced Bold`	Used to identify keystrokes/data entries, file names, directory names, etc.
`Monospaced Italic`	Used to identify variables in an address location. Example: `[AUTOPILOT_HOME]\documents,` where the portion of the address within the brackets [ ] are variable.
`Monospaced Text`	Used to identify addresses, commands, scripts, etc.
Normal Text	Typically used for general text throughout the document.
Table Text	Table text is generally a smaller size to conserve space.

This page intentionally left blank.

Glossary

Application: A logical collection of software components that perform a business function, running on a specific server.

AutoPilot M6: Nastel Technologies' Enterprise Application Management Platform. AutoPilot M6 is designed to monitor and control distributed IT services such as application servers, middleware, user applications, workflow engines, brokers, Service Oriented Architecture (SOA) and Enterprise Service Bus (ESB) based applications and their impact on business services.

AutoPilot M6 for WMQ: Nastel Technologies' WebSphere MQ management solution. Re-designated as M6 for WMQ with release 6.0, prior releases retain the AP-WMQ or MQControl trademark.

AutoPilot M6 Web: AutoPilot M6 Web is a browser-based interface that provides monitoring and operational control over managed resources and applications. It allows users to monitor health, recover from a failure, view historical performance graphs and visualize impacts of a failure.

AutoPilot M6/WebSphere Message Queue Integrator (AP/WMQI): Formerly AP/MQSI

AutoPilot/Message Tracking (AP/MT): Nastel's AutoPilot/Message Tracking plug-in that enables AutoPilot/IT to intercept message exits and forward the statistical data to an AutoPilot expert.

AutoPilot TransactionWorks (AP/TW): Nastel Technologies' transaction and application performance monitoring product.

AutoPilot/WebSphere (AP/WS): AutoPilot/WebSphere plug-into enables AutoPilot to monitor and manage *e*Business applications for continuous operations in addition to its standard features.

BCI: *See* Byte Code Instrumentation

Binary Large Object (BLOB): A collection of binary data stored as a single entity in a database management system. Blobs are typically images, audio or other multimedia objects, though sometimes binary executable code is stored as a blob. Database support for blobs is not universal.

BLOB: *See* Binary Large Object.

BSV: *See* Business View.

Business Transaction: A collection of related transactions that comprise a user-defined business function (e.g. purchase a book, return merchandize, purchase stock). Each of the business activities may be comprised of various workloads.

Business View (BSV): A collection of rules that define a desired state of an *e*Business environment. Business Views can be tailored to present information in the form most suited to a given user, as defined by the user.

Byte Code Instrumentation (BCI): The process of adding small portions of Java byte code around methods of a Java class. The added code performs tasks such as time spent or CPU utilization within the monitored class.

CEP (Complex Event Processing) Server: A container that can host any number of AutoPilot services such as experts, managers, policies, etc. (Called managed node prior to AutoPilot M6 Service Update 6.)

Client: Any programming component that uses the AutoPilot infrastructure; for example, the AutoPilot Console.

Common Object Request Broker Architecture (CORBA): A standard defined by the Object Management Group that enables software components written in multiple computer languages and running on multiple computers to work together. It can be invoked from a Web browser using CGI scripts or applets.

Composite Application: A collection of applications that collaborate or communicate with each other (have related sessions).

Console: The console acts as the graphical interface for AutoPilot.

Contacts: A subordinate to a given Manager or Expert.

CORBA: *See* Common Object Request Broker Architecture.

Data Source Name (DSN): The logical name that is used by Open Database Connectivity (ODBC) to refer to the drive and other information that is required to access data. The name is use by Internet Information Services (IIS) for a connection to an ODBC data source, (Example: Microsoft SQL Server database). The ODBC tool in Control Panel is used to set the DSN. When ODBC DSN entries are used to store the connection string values externally, you simplify the information that is needed in the connection string. This makes changes to the data source completely transparent to the code itself.

Data Space: A range of up to two gigabytes of contiguous virtual storage addresses that a program can directly manipulate. Unlike an address space, a data space can hold only data; it does not contain common areas or system data or programs.

Decision Support System (DSS): An AutoPilot-based service designed to monitor, store, and display any event information generated by AutoPilot enabled middleware and applications.

Deploy: To put to use, to position for use or action.

Domain Server: A specialized managed node that maintains the directory of managed nodes, experts etc. The domain server is also capable of hosting experts, managers etc

DSN: *See* Data Source Name.

DSS: *See* Decision Support System.

Event: An *Event* is something that happens to an object. Events are logged by AutoPilot and are available for use by AutoPilot Policies or the user.

EVT: Event Log file extension (for example: `sample.evt`).

Expert: Services that monitor specific applications such as an applications server, Web server or specific components within the applications (example, channels in MQSeries). Experts generate facts.

Fact: Single pieces of data that has a unique name and value. One or more facts are used to determine the health of the object, application or server.

Graphical User Interface (GUI): A type of environment that represents programs, files, and options by means of icons, menus, and dialog boxes on the screen. The user can select and activate these options by pointing and clicking with a mouse or, often, with the keyboard. Because the graphical user interface provides standard software routines to handle these elements and report the user's actions (such as a mouse click on a particular icon or at a particular location in text, or a key press); applications call these routines with specific parameters rather than attempting to reproduce them from scratch.

GUI: *See* Graphical User Interface.

HAQS: *See* High Availability Queuing Service.

Heap: In Java programming, a block of memory that the Java virtual machine uses at run time to store Java objects. Java heap memory is managed by a garbage collector, which automatically de-allocates Java objects that are no longer in use

High Availability Queuing Service (HAQS): A component of AutoPilot consisting of two policies that provide automatic queue fail-over for WebSphere MQ applications, provide high availability of WebSphere MQ resources such as queues and channels, and ensure automatic recovery of WebSphere MQ channels.

IIS: *See* Internet Information Services.

Independent Software Vendor (ISV): A business term for companies specializing in making or selling software, usually for niche markets.

Initial Program Load (IPL): The process of loading system programs and preparing a system to run applications.

Interactive System Productivity Facility (ISPF): An IBM licensed program that serves as a full-screen editor and dialog manager. Used for writing application programs. It provides a means of generating standard screen panels and interactive dialogues between the application programmer and terminal user.

Internet Information Services (IIS): Microsoft's brand of Web server software, utilizing HTTP to deliver World Wide Web documents. It incorporates various functions for security, allows CGI programs, and also provides for Gopher and FTP services.

IPL: *See* Initial Program Load.

ISPF: *See* Interactive System Productivity Facility.

ISV: *See* Independent Software Vendor.

Java: A platform-independent, object-oriented programming language developed and made available by Sun Microsystems.

Java Database Connectivity (JDBC): Provides universal data access from the Java programming language. Using the JDBC 2.0 API, you can access virtually any data source, from relational databases to spreadsheets and flat files. JDBC technology also provides a common base on which tools and alternate interfaces can be built. The JDBC *Test Tool* that was developed by Merant and Sun Microsystems may be used to test drivers, to demonstrate executing queries and getting results, and to teach programmers about the JDBC API.

Java Developer's Kit (JDK): A set of software tools developed by Sun Microsystems, Inc., for writing Java applets or applications. The kit, which is distributed free, includes a Java compiler, interpreter, debugger, viewer for applets, and documentation.

Java Management Extensions (JMX): An open technology for management and monitoring that can be deployed wherever management and monitoring are needed. By design, this standard is suitable for adapting legacy systems, implementing new management and monitoring solutions and plugging into those of the future.

Java Messaging Service (JMS): a Java Message Oriented Middleware API for sending messages between two or more clients.

Java Platform, Enterprise Edition (Java EE): The industry standard for developing portable, robust, scalable and secure server-side Java applications. Building on the solid foundation of Java SE, Java EE provides Web services, component model, management, and communications APIs that make it the industry standard for implementing enterprise class service-oriented architecture (SOA) and Web 2.0 applications.

Java Naming and Directory Interface (JNDI): Unified interface to multiple naming and directory services for applications based on Java technology.

Java Run-time Environment (JRE): The minimum core Java required to run Java programs.

Java Server Pages (JSP): Technology that enables rapid development of Web-based applications that are platform independent. Java Server Pages technology separates the user interface from content generation enabling designers to change the overall page layout without altering the underlying dynamic content. Java Server Pages technology is an extension of the Java Servlet technology.

Java Virtual Machine (JVM): The "virtual" operating system that Java-written programs run. The JVM is a hardware- and operating system-independent abstract computing machine and execution environment. Java programs execute in the JVM where they are protected from malicious programs and have a small compiled footprint.

JCL: *See* Job Control Language.

JDBC: *See* Java Database Connectivity.

JDK: *See* Java Developer's Kit.

JMS: *See* Java Messaging Service.

JMX: *See* Java Management Extensions.

JNDI: *See* Java Naming and Directory Interface.

Job Control Language (JCL): A control language that is used to identify a job to an operating system and to describe the job's requirements.

JRE: *See* Java Run-time Environment.

JSP: *See* Java Server Pages.

JVM: *See* Java Virtual Machine.

Logical Unit of Work (LUW): A collection of operations and messages within a session that should be considered to be a single unit of work (all or nothing property). These are generally delimited by BEGIN/COMMIT calls.

LUW: *See* Logical Unit of Work.

Managed Node: Containers that are capable of hosting any number of AutoPilot services, such as experts, managers, policies etc.

Manager: Managers are the home or container for policies. All business views must reside on managers, and manager must be deployed prior to deploying a business view or policy.

Message: A physical message being transported through the TPN.

Message-Oriented Middleware (MOM): A category of inter-application communication software that relies on asynchronous message passing as opposed to a request/response metaphor.

Message Queue Interface (MQI): Part of IBM's Networking Blueprint. It is a method of program-to-program communication suitable for connecting independent and potentially non-concurrent distributed applications.

MOM: *See* Message-Oriented Middleware.

MQI: *See* Message Queue Interface.

MQSeries: IBM's message queuing product. Renamed by IBM as WebSphere MQ.

Naming Service: A common server records "names" of objects and associates them with references, locations and properties.

Object Request Broker (ORB): In object-oriented programming, software that serves as an intermediary by transparently enabling objects to exchange requests and responses.

ORB: *See* Object Request Broker.

Orbix: CORBA product distributed by IONA Technologies.

Package Manager: The command line utility that allows users to list, install, uninstall, verify, and update AutoPilot installation on any Managed Node.

PKGMAN: *See* Package Manager Utility included in AutoPilot products.

Policy/Business Views: A collection of one or more sensors. Business views are used to visually present the health and status of the different systems as well as automatically issue remedial actions.

Resource: An entity on which transactions are executed or a medium of exchange. Examples include queue, DB table, file, JMS topic.

Resource Manager: An entity that is managing a collection of resources. Examples include a WMQ Queue Manager, Application Server, Database Server.

Sensor: A rule that is used to determine the health of an object or application based on one or more facts. Actions can then be issued, based on the health.

Server: A physical or virtual node within a TPN that hosts all transaction processing activity.

Service Level Agreement (SLA): A formal written agreement made between two parties: the service provider and the service recipient. The SLA itself defines the basis of understanding between the two parties for delivery of the service itself. The document can be quite complex, and sometimes underpins a formal contract. The contents will vary according to the nature of the service itself, but usually includes a number of core elements, or clauses.

Service-Oriented Architecture (SOA): An evolution of distributed computing and modular programming. SOAs build applications out of software services. Services are relatively large, intrinsically unassociated units of functionality, which have no calls to each other embedded in them. They typically implement functionalities most humans would recognize as a service, such as filling out an online application for an account, viewing an online bank statement, or placing an online book or airline ticket order. Instead of services embedding calls to each other in their source code, protocols are defined which describe how one or more services can talk to each other. This architecture then relies on a

business process expert to link and sequence services, in a process known as orchestration, to meet a new or existing business system requirement.

Session: A specific period of execution of an application. Examples include the interval during which a database or queue manager connection is active.

Simple Mail Transfer Protocol (SMTP): A TCP/IP protocol for sending messages from one computer to another on a network. This protocol is used on the Internet to route e-mail. *See also* communications protocol, TCP/IP.

SLA: *See* Service Level Agreement.

SMTP: *See* Simple Mail Transfer Protocol.

SOA: *See* Service-Oriented Architecture.

TCP/IP: *See* Transmission Control Protocol/Internet Protocol.

Time Sharing Option (TSO): An option of the MVS operating system that provides interactive time sharing from remote terminals.

TPN: *See* Transaction Processing Network.

Transaction: A group of activities targeted at achieving a common goal or a task. Collection of related sessions and LUWs.

Transmission Control Protocol/Internet Protocol (TCP/IP): A protocol developed by the Department of Defense for communications between computers. It is built into the UNIX system and has become the de facto standard for data transmission over networks, including the Internet.

Transaction Processing Network (TPN): A collection of servers engaged in transaction processing activity

TSO: *See* Time Sharing Option.

Virtual Machine: Software that mimics the performance of a hardware device, such as a program that allows applications written for an Intel processor to be run on a Motorola chip. *Also see* Java Virtual Machine.

WebLogic: A Java EE compatible application server platform which enables support for multiple programming models, which includes advanced administration tools and is the ideal foundation for Service Oriented Architecture (SOA).

WebSphere MQ: IBM's message queuing product, formally known as MQSeries.

Websphere_MQ_Manager: A specialized manager capable of hosting one or more WebSphere MQ specific policies, apart from the regular policies.

Wireless Application Protocol (WAP): An open global specification that is used by most mobile telephone manufacturers. WAP determines how wireless devices utilize Internet content and other services. WAP enables devices to link diverse systems contents and controls.

Write to Operator (WTO): An optional user-coded service that allows a message to be written to the system console operator informing the operator of errors and unusual system conditions that may need to be corrected.

WTO: *See* Write to Operator.

This page intentionally left blank.