Singularity

Admin Console Guide

Empowering People with Information that *Moves*

Tom Rose (tom.rose@i-konect.com)

Thursday, January 12, 2006

Version 1.0 - M2

TABLE OF CONTENTS

Revision History

Name	Date	Reason For Changes	Version
Tom Rose	2001-01-09	M2 Release	1.0

Introduction

The purpose of this document is to explain how to use the Admin Console packaged with the Event Process Manager (EPM).

Admin Console Overview

http://<appserver hostname>:<port>/admin (i.e. http://localhost:8080/admin/)

Default credentials are: username = admin, password = admin

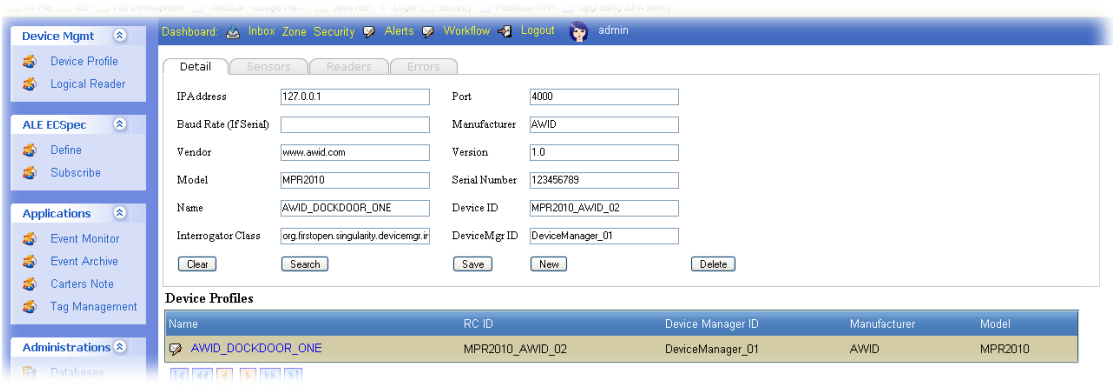
Either from the command-line, or in the service manager or Start Menu shortcuts for Windows as described in the Post Install section of the Install Guide.

The following sections describe each aspect of the console. First the Device Profiles for configuration, Interactive console for Application Level Events (ALE), and then a demo ALE client application. All are packaged in the admin console web application that is part of the EPM.

Device Profile

The profile defines a specific device such as an RFID reader or printer. The configuration of that device is captured, identifying attributes, and then what Device Manager it should be assigned to. A Device Manager may have many devices assigned to it. This example shows an AWID Reader (emulator) that is listening at 127.0.0.1 on port 4000. Vendor, Manufacturer, Firmware Version, Serial Number, ModelNumber are attributes you can assign the profile and are used for information only at this time.

The Name and ID uniquely identifies the device in the system. DeviceMgr ID is the DM this device will be assigned (this is the ID given to the DM when it was installed).



"Interrogator Class" is any class that implements the Interrogator Interface.

Possible values are:

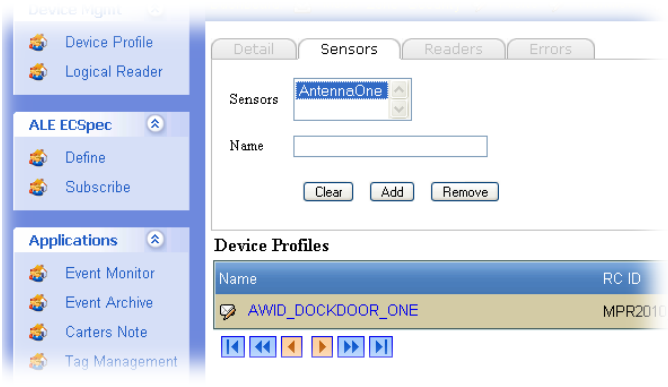
org.firstopen.singularity.devicemgr.interrogator.MPR2010_AWID_IO (default)
org.firstopen.singularity.devicemgr.interrogator.IPSADC_IPICO_IO (serial interface)
org.firstopen.singularity.devicemgr.interrogator.IPSADC_IPICO_TTO_IO (serial interface)
org.firstopen.singularity.devicemgr.interrogator.IPSADC_IPICO_TTO_TCP_IO (Socket Version of TTO)

Note: later releases will have the interrogator software as plugins that have all the device "type" information, and the system will just give a list of available interrogators. Then the instance specific information of network location can be entered.

The future vision is that devices will be packaged with UnPNP, Jini, OSGI technology, and Singularity will be aware of what devices are available on the current network. This will eliminate this kind of configuration; however we are not there yet.

Sensor

Type any name, then press "Add." Devices may have many sensors on them such as a multi-antenna reader. Currently the interrogator plug-ins are combining all events regardless of how many different physical event streams are available on the device, so this definition has no effect, but at least one sensor has to be defined. As noted above, it's intended in the future that the interrogator plug-in would define the physical event sources available on the device, instead of a manual configuration.



Reader

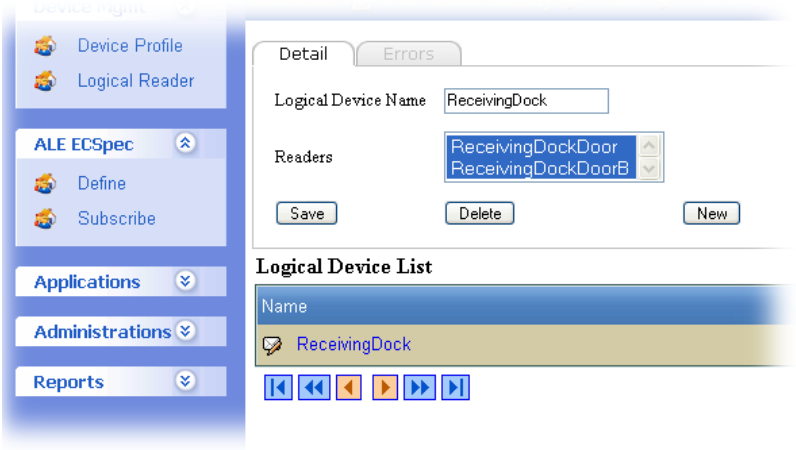
A **Reader** is an event stream and consists of one or more of the physical Sensor event streams. You could have a one to one correlation of Reader to Sensor, or combine sensor event streams. Enter a "Reader Name", select one or more Sensors, and press "Add"

Go back to the details tab and press "Save" otherwise the edits for Sensors and Readers will not be persisted and associated with the Device Profile.

Logical Reader

Now the physical device is defined, assigned a Device Manager, and physical Reader streams defined, Logical Readers can be defined. A Logical Reader can consist of any number of physical Readers, and those Readers of course span multiple Device Managers.

Press "New" to create a new Logical Reader to edit. The physical reader list will then be populated. Select one or more Readers, and enter a "Logical Device Name". Then pres "Save".



Application Level Events

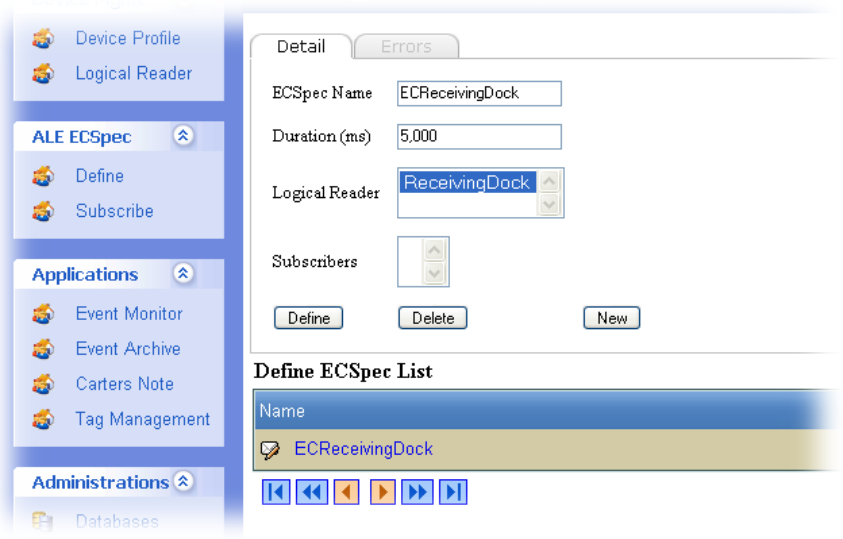
For information about ALE
http://www.epcglobalinc.org/standards_technology/specifications.html

Event Cycle Specification (ECSpec)

ECSpec defines the criteria of an Event Cycle (EC). An EC is started via a trigger from some device or elapsed time, and then is stopped by either receiving a stop trigger or a specific period of time as elapsed. When the EC is complete an ECReport is created and sent to the subscribers. See the ALE Specification to understand details about an ECSpec.

ECSpec can be defined programmatically, as well as interactively via the Admin Console. This screen provides some extremely minimal parameters to configure an ECSpec interactively; however, the full XML ECSpec interface is available.

Define ECSpec



Currently the programmatic ALE interface is provided via a XML/EJB, the interactive version defaults the rest of the values for you, and just allows for a time based event cycle trigger.

Choose a name for the ECSpec, the duration of the event cycle in milliseconds, and then select one or more of the Logical Readers in the list.

If you select a previously defined ECSpec from the list it will populate the details and show any subscribers in the subscriber list box. Also, the current state of the ECSpec is shown in each of the ECSpec List entries. This will be translated to named states, but for now “0” is Un-requested (has no subscribers), and “1” is Active (has subscribers).

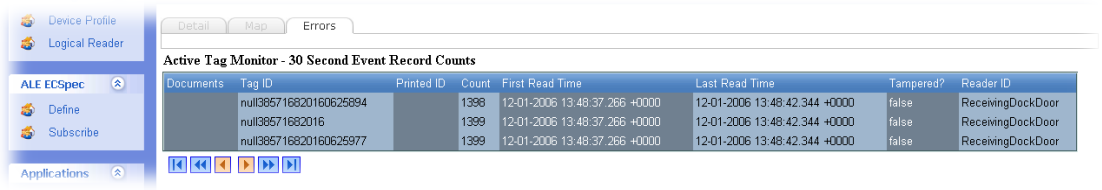
Subscribe ECSpec

Before subscribing to ECSpecs, start the Configuration and Device Managers as described in the Post Installation section of the Install manual.

Once an ECSpec is defined an application can subscribe to receive the events as each event cycle completes. Currently a URI for a JMS queue or HTTP Post is supported.
Note: The default value of Notify URI is the destination that the demo application (Event Monitor) listens on for ECReports sent from the ALE service.

Demo Application

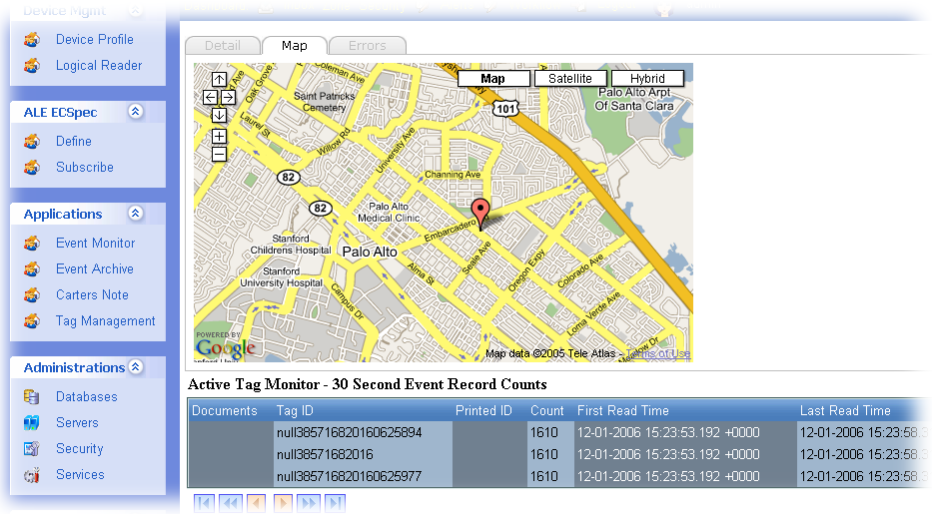
Event Monitor



Event Monitor is a ALE Client application that listens for ECReports on @ jms://localhost/CustomsTrial. The screen will refresh every 10 seconds, and populate the table with the current tags read. Read counts are accumulated for 30 seconds, then an EventRecord is archived to persist the information in the table.

Map

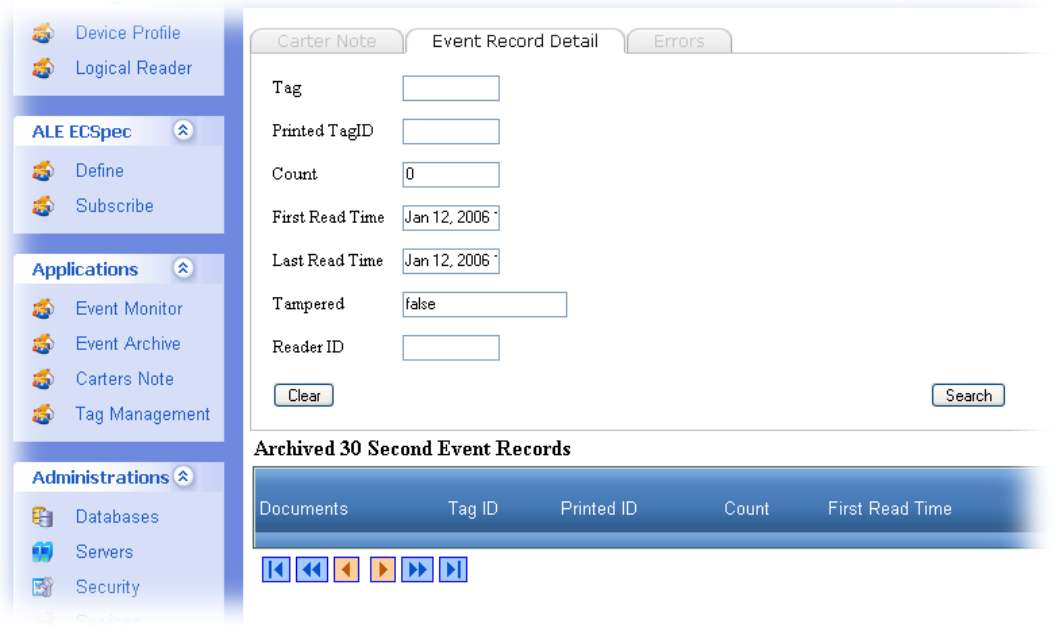
This feature will map the location where the events occurred and is integrated with Google Maps.



*Note: This feature is currently not functional.

Event Archive

Once Event Records recorded by the Event Monitor are archived they can be queried via the query-by-example form in the Viewer Tab. Blank fields are ignored for the search.



Other

The Carters Note is a business document that can be linked to a TagID via the PrintedTagID. In the Tag Management screen just enter the TagID, and PrintedID. When an Event Record is created, it will search the repository for the PrintedID, and if a Carters Note has been correlated to the PrintedID will display an icon in the Documents column of the Event Monitor or Event Archive screen. Select the icon, and the Carters Note will be displayed in the “Details Tab” of whatever screen you are on.