Building an Event Watershed

VDAB, which stands for **Visual Dataflow Application Builder**, provides a powerful framework for visually assembling dataflow components.

VDAB supports the creation of an **Event Watershed** architecture which allows for optimal handling of systems or IoT Data.

This document describes the steps required to build and handle events using the Event Watershed architecture.

Before This	• You should have reviewed the VDAB introductory tutorial and documentation to understand how to create basic flows from nodes.

Contents

Related Documentation

The following document and tutorials either are a) available or b) being developed to further support this subject.

Those available are highlighted in blue while those under development are not highlighted.

Related Documents	Details
Streaming Event Watershed	This Whitepaper introduces the Event Watershed architecture and describes how it enhances the managing and handling of large numbers of events.
Visual Event Dataflow Introduction	This Guide introduces the basic concepts of Event Dataflow programming and details how to build a simple flow.

Related Tutorial	Details

Event Watershed Overview

While effective for standalone Event Dataflow programming, VDAB containers organized as an Event Watershed provide extraordinary capabilities for managing System and IoT data throughout an organization.

The Event Watershed organizes VDAB containers in the structure of watershed and using terms related to bodies of water to help to clarify how data is handled.

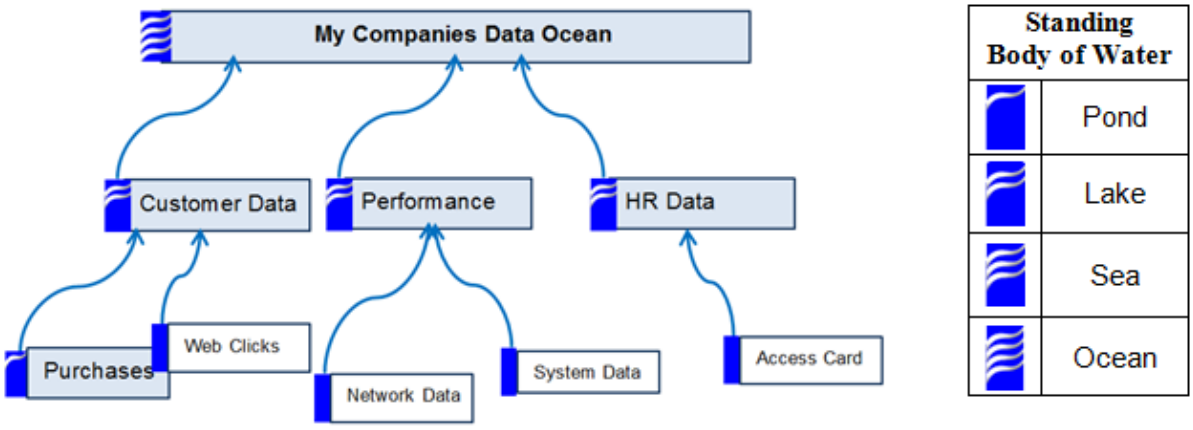
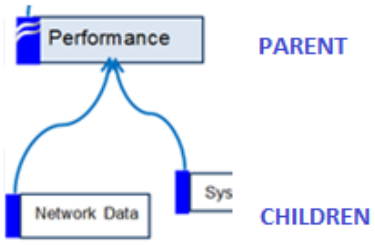


Fig1. Event Watershed

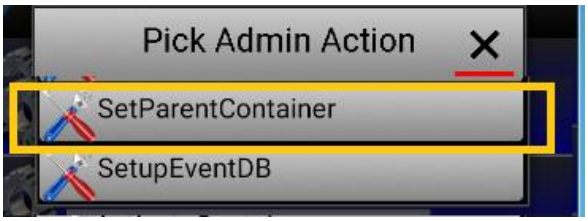
For a full background on the Event Watershed architecture refer to the whitepapers identified in the Related Documents section.

The Watershed is built by designating the Parent/Child relationship between each container. Notice that the data flows from the child to parent container.



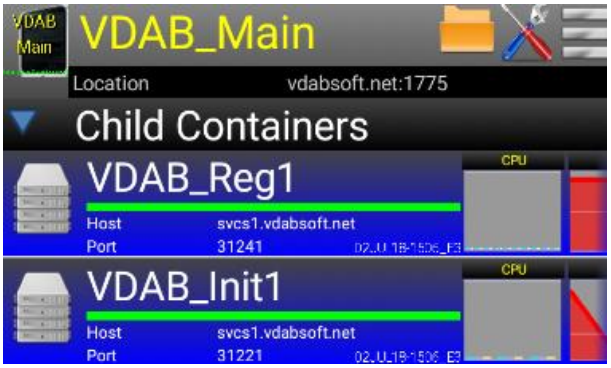
Parent Container

Every VDAB Container can designate a *Parent Container* that provides a path for propagating data *"downstream"* (actually up in Fig1).



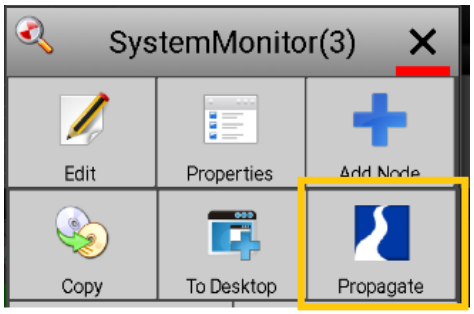
Child Containers

A Container designated as a parent will have one or more child containers.



Propagating Events to the Parent

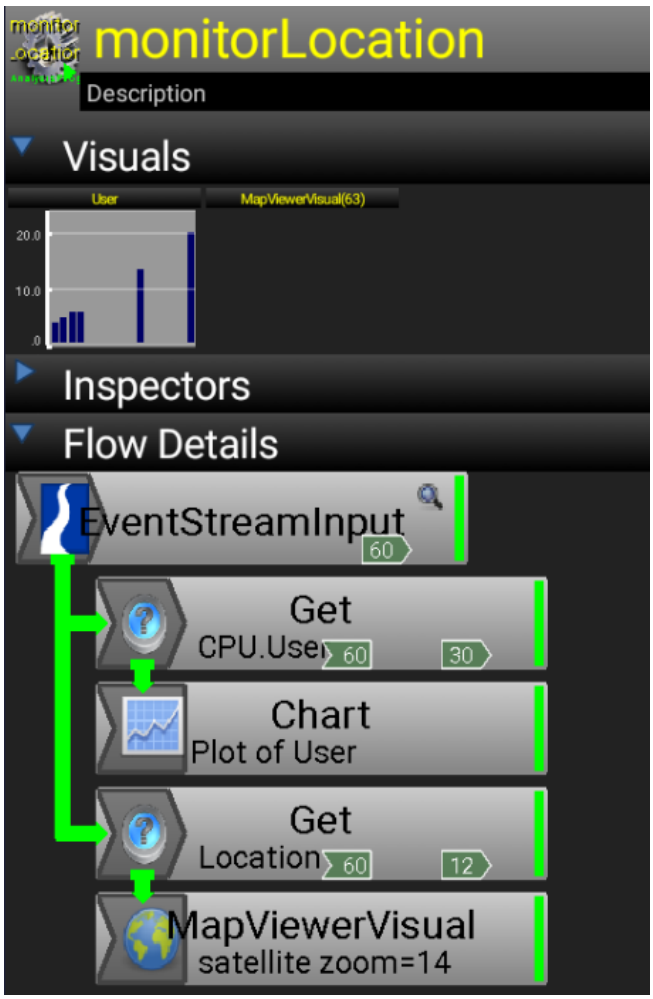
Events can be very easily pushed to the Parent container by selecting the Propagate action.



Data can pushed to the parent's parent, the parent's grandparent or any ancestor by selecting a propagation level associated with a body of water designation. (See detailed directions below.)

Sampling Events from Children

Parents can sample the events coming from any of its children by building a flow with an *EventStreamInput* Node. In the example below the system utilization and GPS location being propagated from a child container are being sampled.

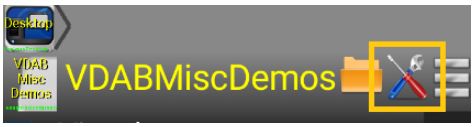


Setting the Parent Container

While VDAB processing containers can work independently, declaring parent/child relationship will allow pushing data to the parent for later processing. This is accomplished by calling *Set Parent Container* administrative action or by editing the container properties directly.

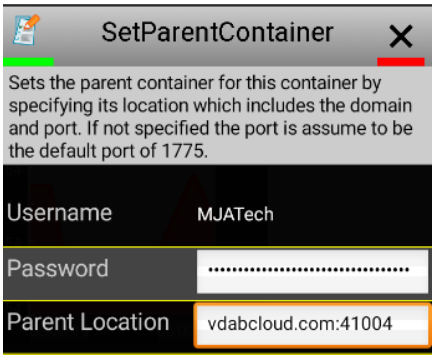
Setting the Parent Container using the Administrative Tools

1. While on the container main screen, click on the tools icon



2. Click on the *SetParentContainer* Admin Action.

3. Set the Parent Location which includes the parent's domain name or IP and port. (If the port is not designated, the standard vdab port (1775) will be used.

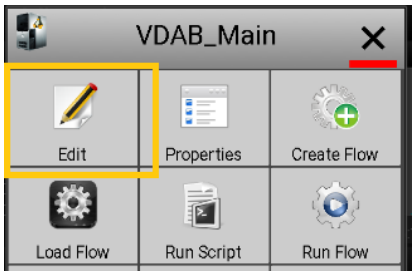


4. Click on the upper left to submit the change. The container will reinitialize and will begin sending propagated events to this parent.

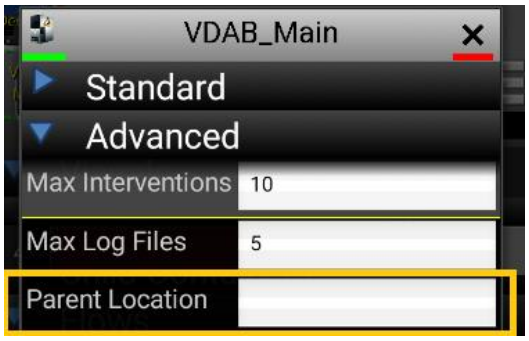
Setting the Parent Container by Editing the Container

1. Bring up the Actions Menu by clicking on the far right Icon.

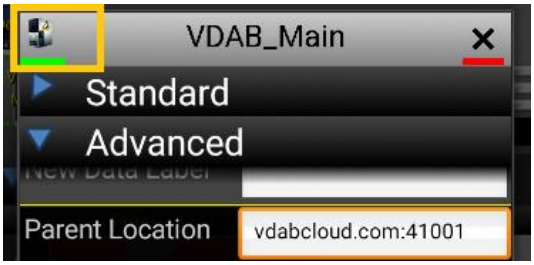
2. Select the *Edit* action.



3. Scroll down to the advanced properties and click on the *Advanced* title bar to show the advanced options. Edit the Parent Location which includes the parents domain name or IP and



4. Click on the upper left to save the edits.

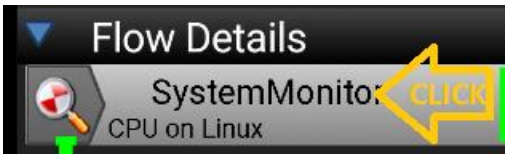


5. While events will be propagated to the parent container immediately, the container configuration must be saved to make the parent assignment permanent.

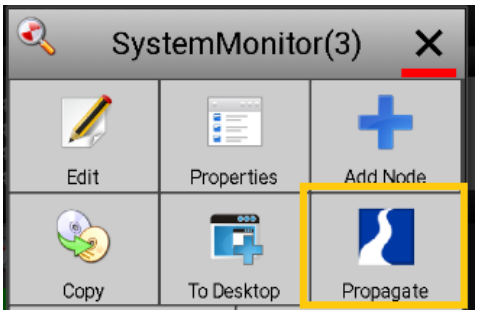
Propagating Data to the Parent and Watershed

Individual events can be sent to parent container by setting the nodes to propagate to the parent and to the entire watershed.

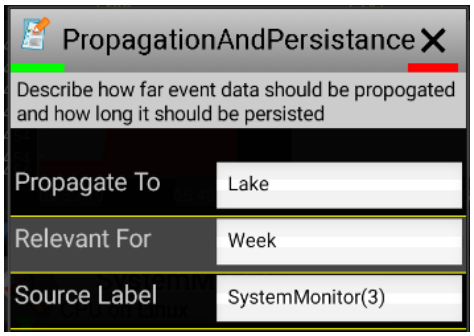
1. Click on the desired node to bring up the Actions Menu



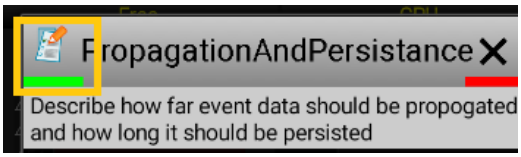
2. Select the *Propagate* action.



3. Select how far down the watershed the events should be propagated and how long the event data should be retained. If desired you can rename the data label indicating the source of the data. (This may be desirable when retrieving and using the event data.)

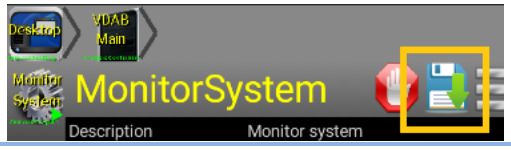


4. Click on the upper left to start event propagation.



5. The node will now include an icon indicating that data is being persisted and how far it will travel down the watershed.

6. Save the current flow to make these changes permanent.

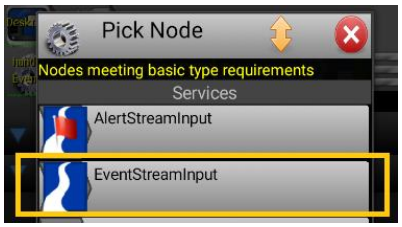


Processing Data in the Event Stream

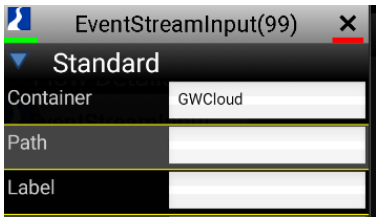
Data that is propagated to a parent or any downstream container can be sampled and processed in that container. The procedure below allows the sampling and alerting of data

1. Create a flow to handle eEvents coming from the child container. In the example below we will handle location and system data coming from an Android smartphone.

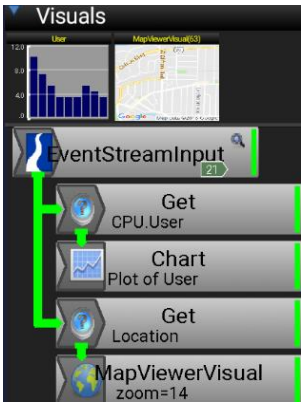
2. Add an *EventInputStream* source node to the container.



3. Edit the *EventStreamInput* Node to select the specific events you are interested in processing. Drop downs will present the options for Container, paths and data labels that have been coming into the stream. Leave blank to sample all events.



4. Add nodes to select and process the incoming data. As an example you can select a specific data element using the *GetElement* node and chart the data. You can also select location data and display a map of the location



5. A flow that monitors the event stream must be kept running to ensure it captures all events from the children. Set Auto Load and Auto Start both to true and save the flow to ensure all events are processed.