SAND2016-12875R

IDC RE-ENGINEERING REPORT

SAND2016-WXYZ
Unlimited Release
December, 2016

IDC Re-Engineering Phase 2 Iteration E1 Use Case Realizations

Version 1.2

J. Mark Harris, John F. Burns, Benjamin R. Hamlet, Mark S. Montoya, Rudy D. Sandoval

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-mission laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

SAND2016-WXYZ
Unlimited Release
December, 2016

IDC Re-Engineering Phase 2 Iteration E1 Use Case Realizations

Version 1.2

J. Mark Harris, John F. Burns, Rudy D. Sandoval
Dynamic Monitoring Software

Benjamin R. Hamlet, Mark S. Montoya
Ground System Development

Sandia National Laboratories
P.O. Box 5800
Albuquerque, New Mexico 87185-MS0401

Abstract

This document contains 4 use case realizations generated from the model contained in Rational Software Architect. These use case realizations are the current versions of the realizations originally delivered in Elaboration Iteration 1.

REVISIONS

Version	Date	Author/Team	Revision Description	Authorized by
1.0	10/12/2015	SNL IDC Re-Engineering Team	Initial Release for E1	M. Harris
1.1	12/17/2015	SNL IDC Re-Engineering Team	Release for E1	M. Harris
1.2	12/16/2016	SNL IDC Re-Engineering Team	Release for the end of the Elaboration Phase	M. Harris

TABLE OF CONTENTS

Use Case Hierarchy

The IDC Use Case Hierarchy is shown here. The use cases highlighted in yellow are the use case realizations that appear in this document.

1	**System Acquires Data**
1.1	System Receives Station Data
1.2	System Receives Bulletin Data
1.3	System Automatically Distributes Data
1.4	System Acquires Meteorological Data
1.5	System Synchronizes Acquired Station Data
1.6	System Synchronizes Processing Results
2	**System Detects Event**
2.1	System Determines Waveform Data Quality
2.2	System Enhances Signals
2.3	System Detects Events using Waveform Correlation
2.4	System Detects Signals
2.5	System Measures Signal Features
2.6	System Builds Events using Signal Detections
2.7	System Resolves Event Conflicts
2.8	System Refines Event Location
2.9	System Refines Event Magnitude
2.10	System Evaluates Moment Tensor
2.11	System Finds Similar Events
2.12	System Predicts Signal Features
3	**Analyzes Events**
3.1	Selects Data for Analysis
3.2	Refines Event
3.2.1	Determines Waveform Data Quality
3.2.2	Enhances Signals
3.2.3	Detects Signals
3.2.4	Measures Signal Features
3.2.5	Refines Event Location
3.2.6	Refines Event Magnitude
3.2.7	Evaluates Moment Tensor
3.2.8	Compares Events
3.3	Scans Waveforms and Unassociated Detections
3.4	Builds Event
3.5	Marks Processing Stage Complete
4	**N/A**
5	**Provides Data to Customers**
5.1	Requests System Data

5.2 Views System Results

6 Configures System

6.1 Controls Data Acquisition
6.2 Configures Station Usage
6.3 Defines Processing Sequence
6.4 Configures Data Acquisition
6.5 Configures Processing Components
6.6 Views System Configuration History
6.7 Configures Analysis Interfaces
6.8 Configures System Permissions

7 Monitors Performance

7.1 Analyzes Mission Performance
7.2 Monitors System Performance
7.3 Monitors Station State-of-Health
7.4 System Monitors Mission Performance
7.5 Monitors Mission Processing

8 Supports Operations

8.1 Accesses the System
8.2 Controls the System
8.3 Exports Data
8.4 Imports Data
8.5 Views Event History
8.6 Maintains Operations Log
8.7 Provides Analyst Feedback
8.8 Views Analyst Feedback
8.9 Views Analyst Performance Metrics
8.10 Views Security Status
8.11 Views Messages

9 Tests System

9.1 Performs Software Component Testing
9.2 Creates Test Data Set
9.3 Replays Test Data Set
9.4 Replays Analyst Actions

10 Maintains System

10.1 Performs System Backups
10.2 Performs System Restores
10.3 Installs Software Update
10.4 System Monitors Security

11 Performs Research

11.1 Analyzes Research Events
11.2 Develops New Algorithms and Models
11.3 Determines Optimal Processing Component Configuration

11.4 Performs Multiple Event Location

12 Performs Training

12.1 Configures Data for Training Subsystem

12.2 Trains Analysts

13 Operates Standalone Subsystem

13.1 Conducts Site Survey

13.2 Performs Standalone Analysis

14 IDC Unique

14.1 Assesses Event Consistency

14.3 System Screens Event

14.4 System Controls Stations

14.5 Performs Expert Technical Analysis

IDC Use Case Realization Report

UCR-02 System Detects Event

Use Case Description

This use case describes how the System pipeline processes the raw seismic, hydroacoustic, and infrasound waveform data from a time interval to form event hypotheses. The System first checks the quality of arriving waveform data and creates data quality control masks for waveform sections containing data that is unsuitable for processing (see 'System Determines Waveform Data Quality' UC). The System then processes waveforms to enhance signal content while reducing noise (see 'System Enhances Signals' UC).

The System detects signals (see 'System Detects Signals' UC), measures features on the signal detections (see 'System Measures Signal Features' UC), and then uses the signal detections and feature measurements to build both single station and network event hypotheses (see 'System Builds Events using Signal Detections' UC). The System uses channel based waveform correlation techniques to form single-station or network event hypotheses (see 'System Detects Events using Waveform Correlation' UC). The System measures signal features for the event hypotheses on waveform channels across the network (see 'System Measures Signal Features' UC). The System predicts signal detections and their features for events (see 'System Predicts Signal Features' UC). The System uses similarity parameters to search for historic events similar to the new event hypothesis (see 'System Finds Similar Events' UC).

After forming event hypotheses, the System resolves conflicting event hypotheses (see 'System Resolves Event Conflicts' UC) and then refines each event hypothesis' location (see 'System Refines Event Location' UC) and magnitude (see 'System Refines Event Magnitude' UC). The System evaluates the moment tensor for an event (see 'System Evaluates Moment Tensor' UC).

The System pipeline follows a sequence configured by the System Maintainer when pipeline processing raw waveform data to form event hypotheses (see 'Configures Processing Sequence' UC).

Architecture Description

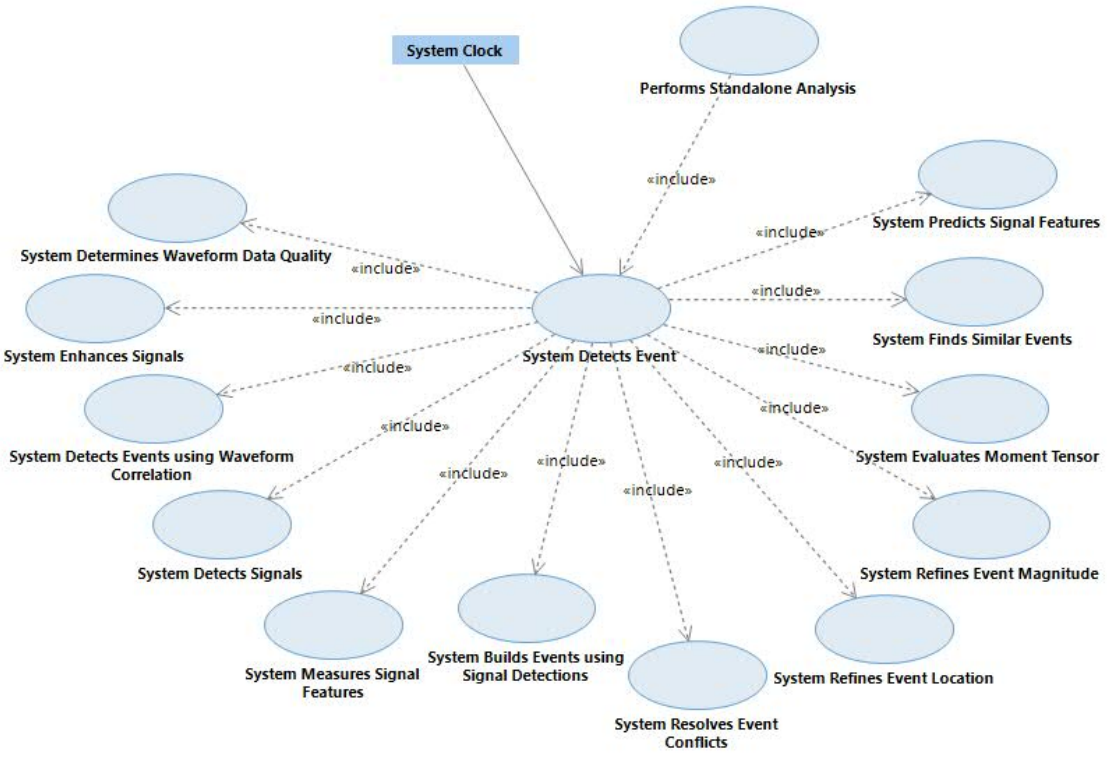
The Processing Sequence Control mechanism is responsible for executing processing sequences previously defined by the System Maintainer (see 'Defines Processing Sequence' UCR). Processing Sequence classes contain Processing Steps and Flows. These objects form a graph with Data References travelling on Flows between Processing Steps. Processing Sequence Control executes a Processing Sequence whenever a Triggering Condition initiating the Processing Sequence is satisfied. Several types of Processing Step exist in the System. A Processing Component Invocation Step invokes a control class to perform processing. Control classes invoked in this manner all realize a common interface named Processing Control IF which allows the Processing Sequence Control mechanism to invoke them. Control Flow Steps define loops around one or more Processing Steps, branches to select which of several Processing Steps to execute, and Processing Steps that Processing Sequence Control can execute in parallel. A Processing Sequence Invocation Step invokes one Processing Sequence as part of

another Processing Sequence.

The Processing Control IF realizations may store data in the OSD, causing OSD callbacks to Processing Sequence Control. Processing Sequence Control places data references to the received data on Flows, making the data available to subsequence Processing Steps.

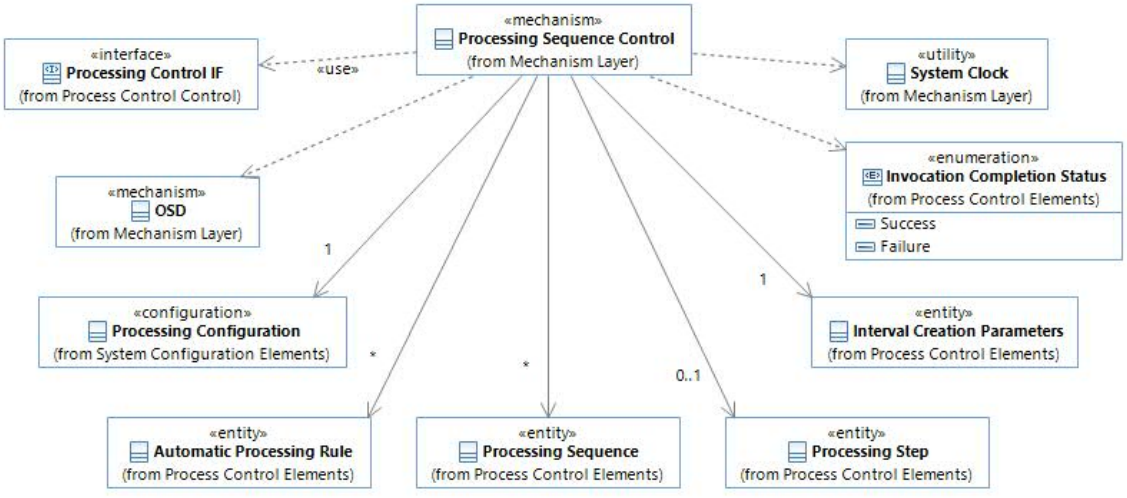
Processing Sequence Control uses Interval Creation Utility to create Intervals to track automatic and interactive processing workflow. Processing Sequence Control sets the Interval Status for automatic Processing Stage Intervals and Processing Sequence Intervals and publishes the status updates to the OSD. The Analyst views this status in 'Selects Data for Analysis' UCR and the System User views it in 'Monitors Mission Processing' UCR.

Use Case Diagram



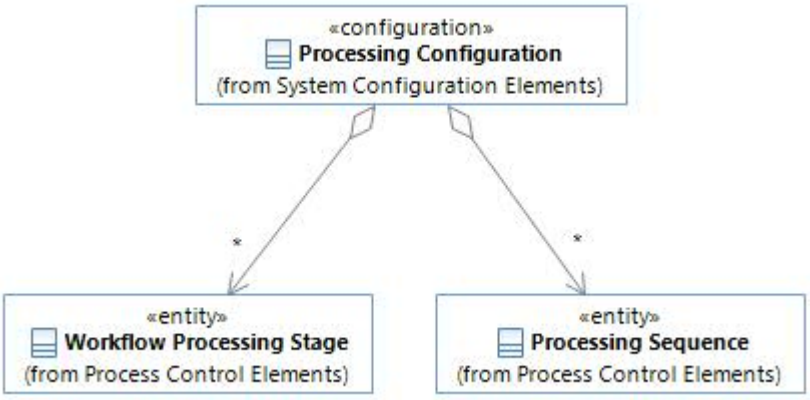
Class Diagrams

Classes - Processing Sequence Control



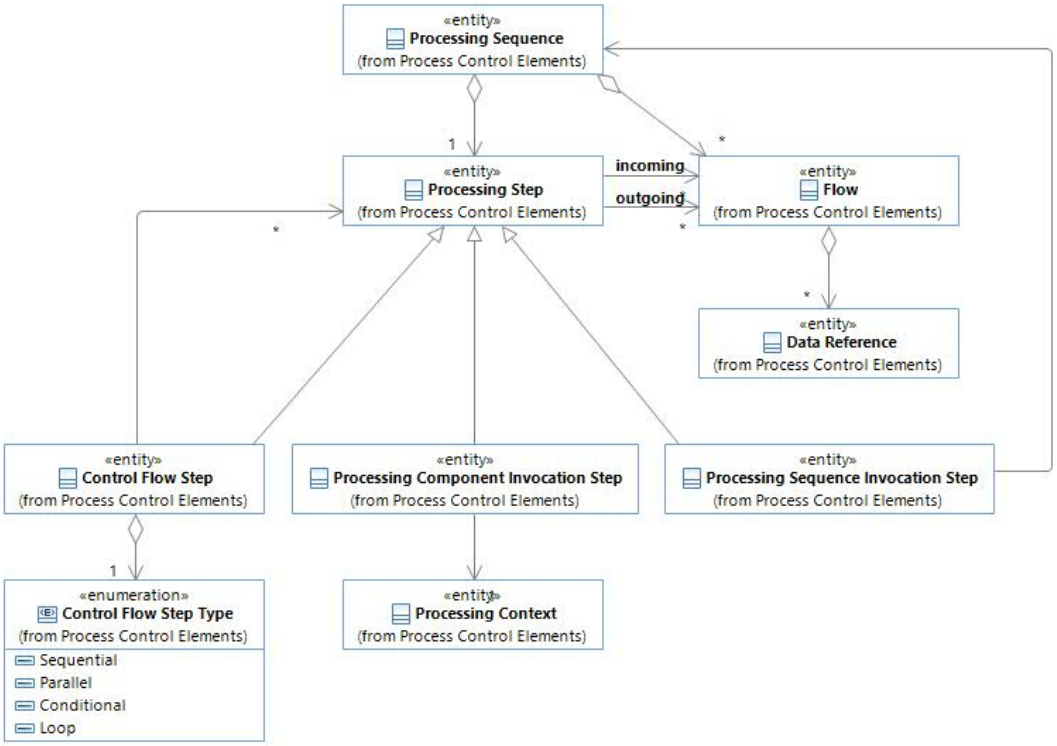
This diagram shows the Processing Sequence Control class and related classes. Processing Sequence Control periodically evaluates Automatic Processing Rules to initiate new Processing Sequences, executes the Processing Steps within a Processing Sequence, and monitors processing performed by realizations of the Processing Control IF interface.

Classes - Processing Configuration



This diagram shows the Processing Configuration class which the System Maintainer preconfigures (see 'Defines Processing Sequence' UCR) with the System's Workflow Processing Stages, Processing Sequences, and Interval creation configuration. Processing Sequence Control loads the Processing Configuration on startup (see "Alternate Flow – Processing Sequence Control - Startup").

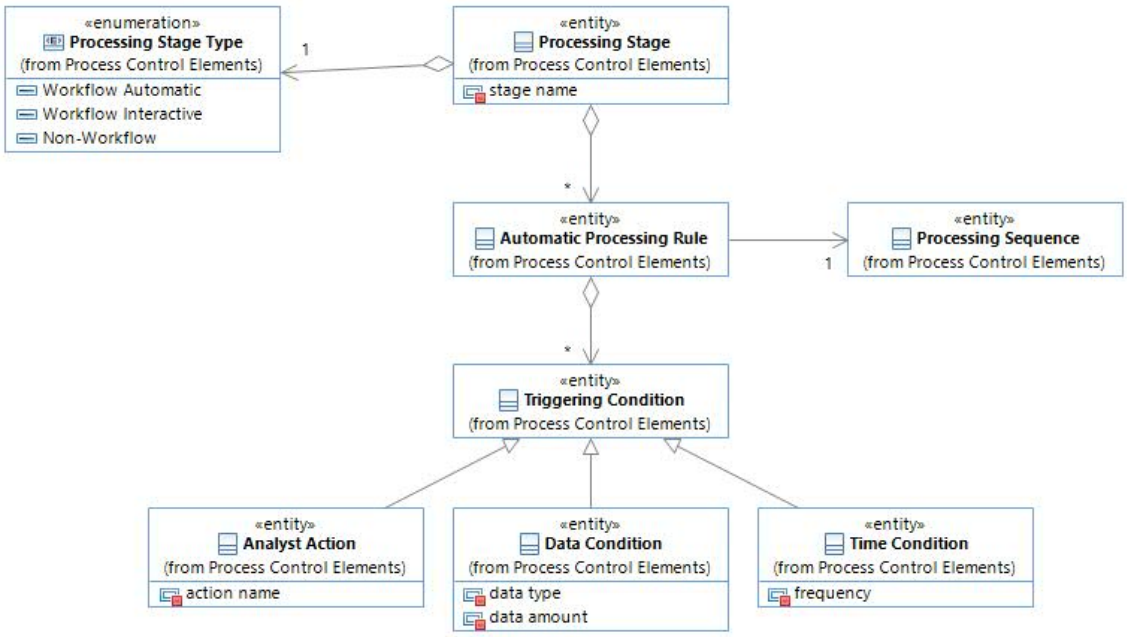
Classes - Processing Sequence



This diagram shows the structure of a Processing Sequence. A Processing Sequence consists of a graph of Processing Steps ordered based on their incoming and outgoing Flows. The System Maintainer configures Processing Sequences (see "Defines Processing Sequence" UCR) and the Processing Sequence Control mechanism is responsible for executing the steps of the graph in

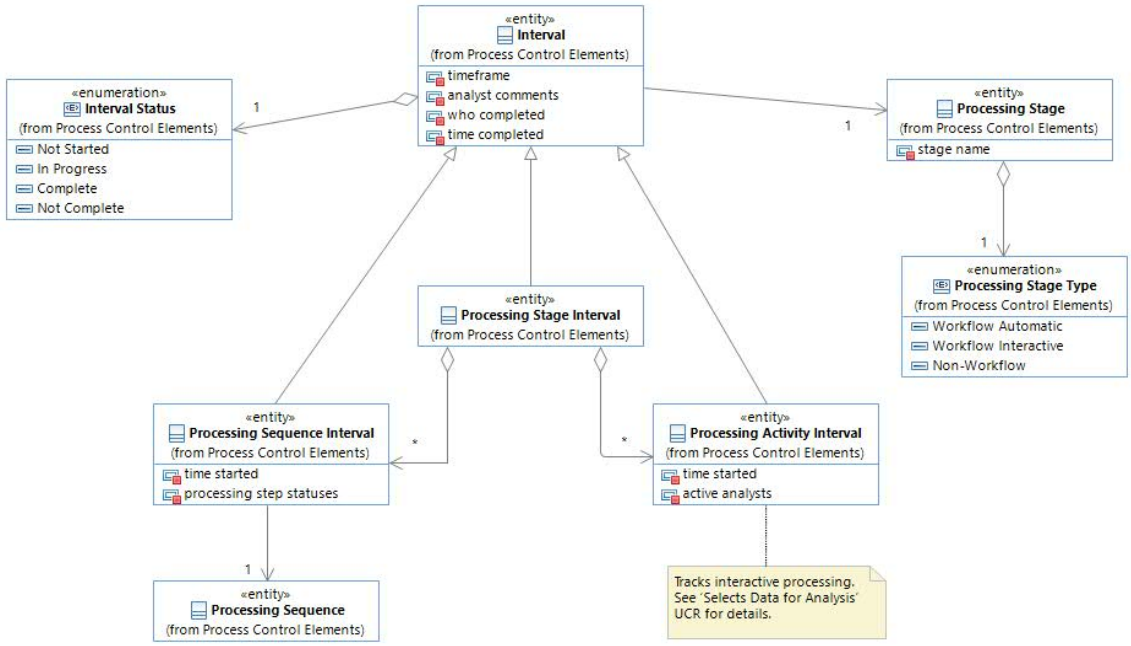
the proper order. Each step in the graph invokes a Processing Component, invokes another Processing Sequence or acts as a Control Flow Step. Control Flow Steps are the only steps that contain child steps. The child steps represent operands for the control flow. The number of children varies according the type of control flow. For example, a "Conditional" control flow step might always have two children: one for the "true" branch and one for the "false" branch. On the other hand, a "Sequential" or "Parallel" control flow step could have a variable number of children.

Classes - Automatic Processing Rule



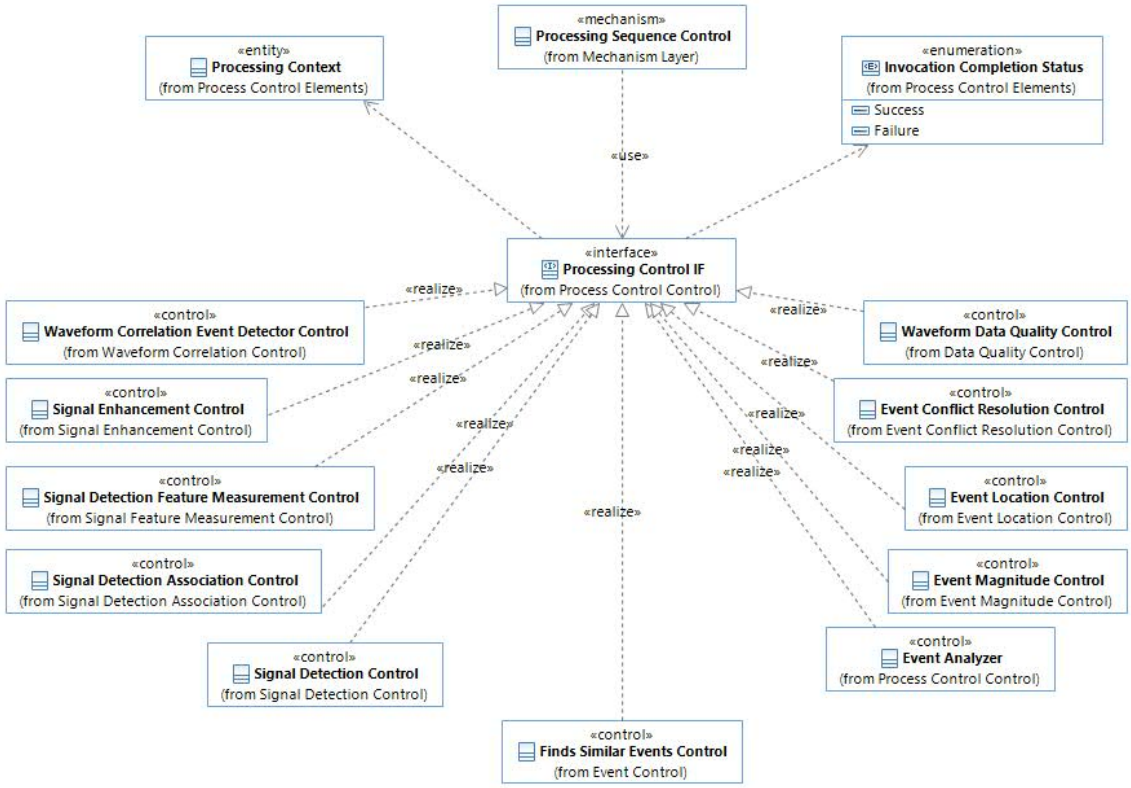
This class shows the structure of an Automatic Processing Rule. An Automatic Processing Rule combines one or more Triggering Conditions with the Processing Sequence executed by the Processing Sequence Control when the System's state satisfies those conditions. Several types of Triggering Conditions exist in the System, including Analyst Action conditions satisfied when the Analyst takes some action, Data Conditions satisfied when certain types or amounts of data are acquired or created on the System, and Time Conditions satisfied when a certain amount of time has elapsed since the conditions was previously satisfied. The System Maintainer can configure Automatic Processing Rules for each Processing Stage (see "Defines Processing Sequence" UCR).

Classes - Interval



This diagram shows the Interval class and related classes. Processing Sequence Control uses the Interval Creation Utility and Interval Creation Parameters to create Intervals at the correct times (see "Alternate Flow – Processing Sequence Control – Create Intervals"). While executing Processing Sequences, the Processing Sequence Control sets the "processing step statuses" in the Processing Sequence Interval class as well as the Interval Status for Processing Sequence Intervals and Processing Stage Intervals.

Classes - Processing Control IF



This diagram shows the Processing Control IF interface, a representative set of classes realizing its interface, and the Processing Control IF dependencies. Each instance of the Processing Component Invocation Step class (see "Classes - Processing Sequence Control") is configured to invoke one of the classes realizing the Processing Control IF through its Invoke() method. The UCRs included by System Detects Event define behaviors for these classes.

Class Descriptions

<<configuration>> Processing Configuration
Consists of workflow processing stages and processing sequences that control the automatic processing and analyst workflow performed by/on the system. Includes configuration describing which Intervals (both automatic and interactive processing) the System creates and the timeframes the System uses to create each new Interval.

<<control>> Event Analyzer
Responsible for analyzing and updating events with information indicating whether further (e.g. iterative) automated processing should be performed. Instantiations of this class implement processing sequence control logic that is unavailable in the other classes realizing the Processing Control IF.

Event Analyzer implementations have the option to update the processing configuration parameters used by subsequent Processing Steps. This is similar to how human Analysts select processing parameters for algorithms they invoke, except that Event Analyzer does not directly

invoke additional Processing Steps. Instead, Event Analyzer stores information in the OSD and then relies on callbacks to the Processing Sequence Control mechanism to initiate additional processing. Event Analyzer does this by creating a copy of a data object (e.g. a new version of an event hypothesis, etc.), updating the processing configuration parameters on that data object, and then storing the data object and its associated processing configuration parameters in the OSD. OSD callbacks may result in Automatic Processing Rules being satisfied, causing the Processing Sequence Control mechanism to invoke additional Processing Sequences using the updated processing configuration parameters. The data objects stored in the OSD are also Event Analyzer's outputs, so the Processing Sequence Control Mechanism may set data references to these objects on the outgoing Flows for the Processing Step that invoked Event Analyzer.

<<control>> Event Location Control
Responsible for controlling the event location computation. Retrieves necessary data, invokes the appropriate Event Locator Plugin to compute the new location, and stores the result.

<<control>> Finds Similar Events Control
Responsible for controlling the search for similar Events.

<<control>> Signal Detection Association Control
Control class responsible for controlling signal detection association calculations. Retrieves configuration from the OSD, invokes the appropriate Signal Detection Associator Plugin, computes quality metrics, and stores the new or modified events in the OSD.

<<control>> Signal Detection Control
Responsible for controlling automatic signal detection. Retrieves necessary data, invokes plugins to detect signals on waveform data and refine signal onset time, and stores the results.

<<control>> Signal Enhancement Control
This Control class is responsible for controlling the signal enhancement computations. It obtains the data necessary for the execution of a signal enhancement, invokes the appropriate Signal Enhancement Plugin to perform the enhancement, and stores the results via the OSD mechanism. Signal Enhancement Control is started and stopped by the System Control mechanism and is invoked by the Processing Sequence Control mechanism. When invoked, it is passed a processing context, references to data it needs for its processing, and optionally a set of processing parameters that override the default parameters. The default processing parameters are provided as part of Signal Enhancement Control's static configuration. The primary data required by the Signal Enhancement Control is a set of Waveforms that needs potential signals within it enhanced.

<<control>> Waveform Correlation Event Detector Control
Responsible for controlling waveform correlation event detection computations. Retrieves necessary data, invokes the appropriate Waveform Correlation Event Detector Plugin to detect new events, and stores the results.

<<control>> Waveform Data Quality Control
Waveform Data Quality Control class is responsible for controlling the waveform data quality

computations that create and update Waveform QC Masks. It retrieves the configuration, parameters, and other data necessary for its execution, invokes the appropriate Waveform Data Quality Plugin to compute Waveform QC Masks, and stores the masks in the OSD. The System Control mechanism starts and stops Waveform Data Quality Control. The Processing Sequence Control mechanism invokes Waveform Data Quality Control when executing preconfigured Processing Sequences (see 'System Detects Event' UCR).

<<entity>> Analyst Action
Special type of Triggering Condition that is based on an action performed by an Analyst. The set of available actions is predefined by the system.

<<entity>> Automatic Processing Rule
Represents a Processing Sequence and the set of Triggering Conditions for initiating it.

<<entity>> Control Flow Step
A specialized kind of Processing Step that is used to represent control flow between other Processing Steps. The Control Flow Step is represented by the type of control flow operation (e.g. Parallel, Conditional, etc.) and operands to which it applies (i.e. other Processing Steps).

<<entity>> Data Condition
Special type of Triggering Condition that is based on the availability of data (e.g. 100 signal detections, 15% of all waveforms for the interval).

<<entity>> Data Reference
Represents a reference to data that passes between Processing Steps on a Flow.

<<entity>> Flow
Represents control and data flow between two Processing Steps.

<<entity>> Interval
Class for tracking the status of interactive or automatic processing on a specific timeframe of data. Specialized intervals exist for Processing Stage, Processing Activity, and Processing Sequence.

<<entity>> Interval Creation Parameters
Represents the parameters used by Interval Creation Utility to create new Intervals. This includes which Intervals to create at which times. May also define when a certain amount of data (e.g. acquired waveforms) must exist before the Interval Creation Utility creates a new Interval.

<<entity>> Processing Component Invocation Step
A specialized kind of Processing Step that represents an invocation of a specific Processing Component.

<<entity>> Processing Context
Represents the context in which data is being stored and/or processed. This includes the

Processing Stage (either automatic or interactive) and Interval performing the processing session (e.g. processed by Analyst vs. processed by System). For Analyst processing, may identify the Analyst work session. For System processing, may identify the Processing Sequence and/or Processing Step being executed (including a way to identify a particular Processing Sequence and Processing Step among the many possible instantiations), the visibility for the results (private vs. global), and the lifespan of the data (transient vs. persistent). This information is needed by the Processing Sequence Control to manage the execution of Processing Sequences, which may execute in the context of an Analyst refining an Event or in the context of the system initiating automatic processing. It is also needed by the Object Storage and Distribution (OSD) mechanism to determine how to store and distribute the data.

<<entity>> Processing Sequence
A user-configurable set of Processing Steps to be executed by the Processing Sequence Control mechanism. Each Processing Step may invoke a Processing Component or another Processing Sequence. Special steps are used to specify control flow (e.g. conditional logic, parallelism, etc.).

<<entity>> Processing Sequence Invocation Step
A specialized kind of Processing Step that represents an invocation of a specific Processing Sequence.

<<entity>> Processing Stage
Represents a named stage of data processing, which may be part of the System Maintainer-defined workflow or an Analyst-defined stage outside the workflow. All Processing Results are associated to a Processing Stage.

<<entity>> Processing Step
Represents a single step within a Processing Sequence. A Processing Step may invoke a Processing Component or invoke a Processing Sequence, and may optionally specify parameter overrides for the invoked component/sequence. Special kinds of Processing Steps known as Control Flow Steps are used to specify control flow between Processing Steps.

<<entity>> Time Condition
Special type of Triggering Condition that is based on time (e.g. every 5 minutes).

<<entity>> Triggering Condition
Represents a condition which must be satisfied in order to trigger a Processing Sequence.

<<entity>> Workflow Processing Stage
Represents a Processing Stage that is part of the System Maintainer-defined Analyst workflow of automatic processing stages and Analyst review stages.

<<enumeration>> Interval Status
Represents the current status of a Processing Sequence Interval or a Processing Activity Interval.

<<interface>> Processing Control IF

Defines the interface implemented by all <<control>> classes in the system that are controlled by the Processing Sequence Control <<mechanism>>. <<control>> classes realize this common interface to support configurable processing sequence definition and execution. Processing Sequence Control uses the Invoke() operation declared in Processing Control IF to call <<control>> classes while executing processing sequences. When called in this way the <<control>> classes operate on the provided data (e.g. event hypotheses, signal detections, etc.) using either default parameters configured by the System Maintainer and loaded by the <<control>> class on startup or override parameters provided to the Invoke() operation.

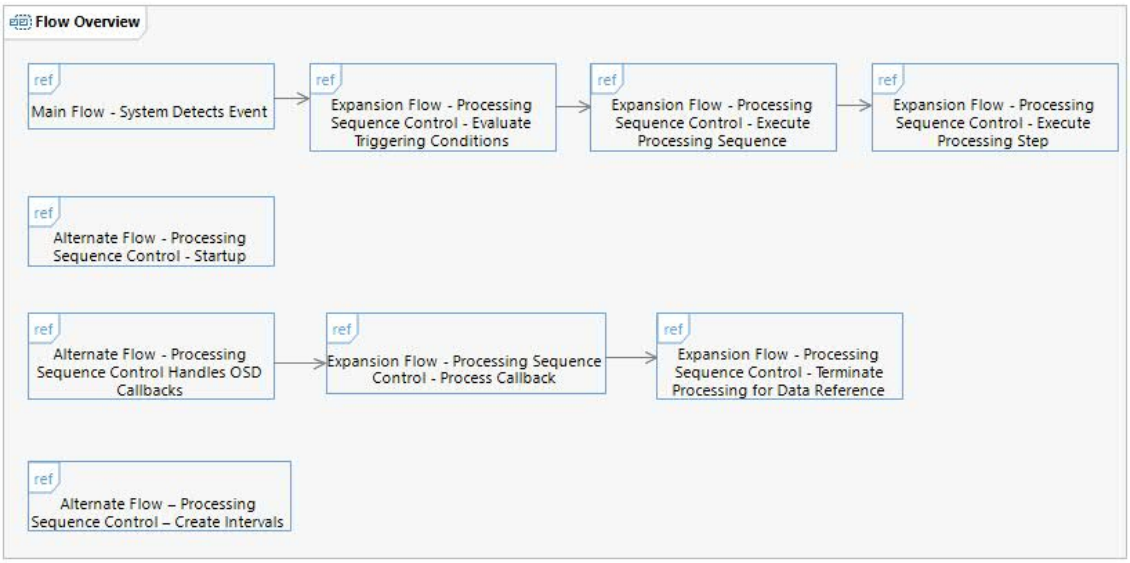
<<mechanism>> OSD
Represents the Object Storage and Distribution mechanism for storing and distributing data objects internally within the system.

<<mechanism>> Processing Sequence Control
Mechanism for executing and controlling processing sequences configured by the System Maintainer.

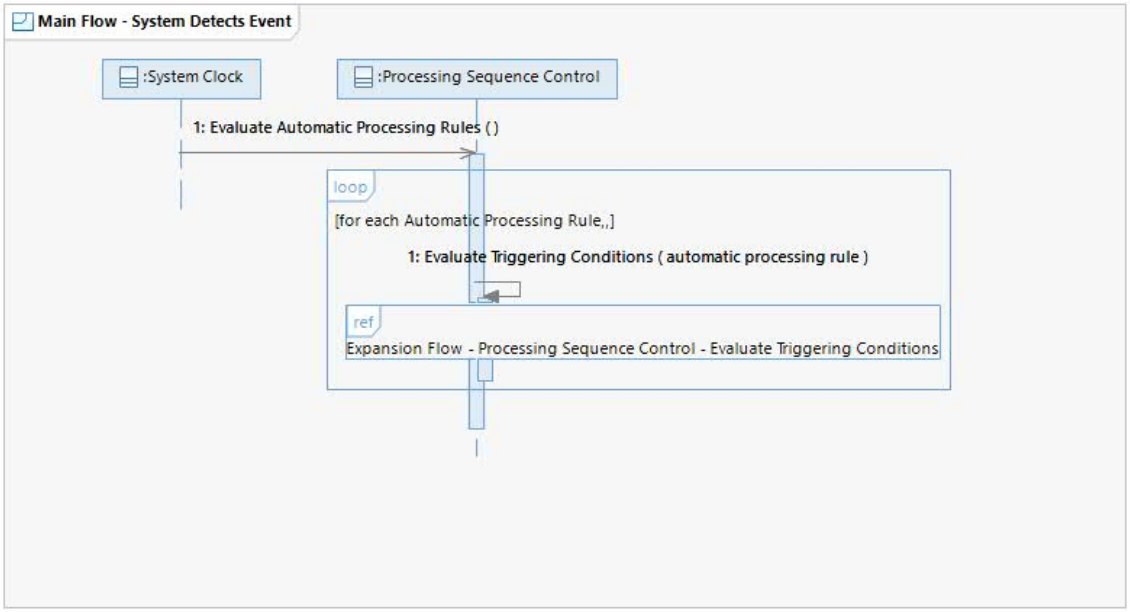
<<utility>> System Clock
Represents the mechanism to schedule, reschedule, and cancel callbacks.

Sequence Diagrams

Flow Overview



Main Flow - System Detects Event

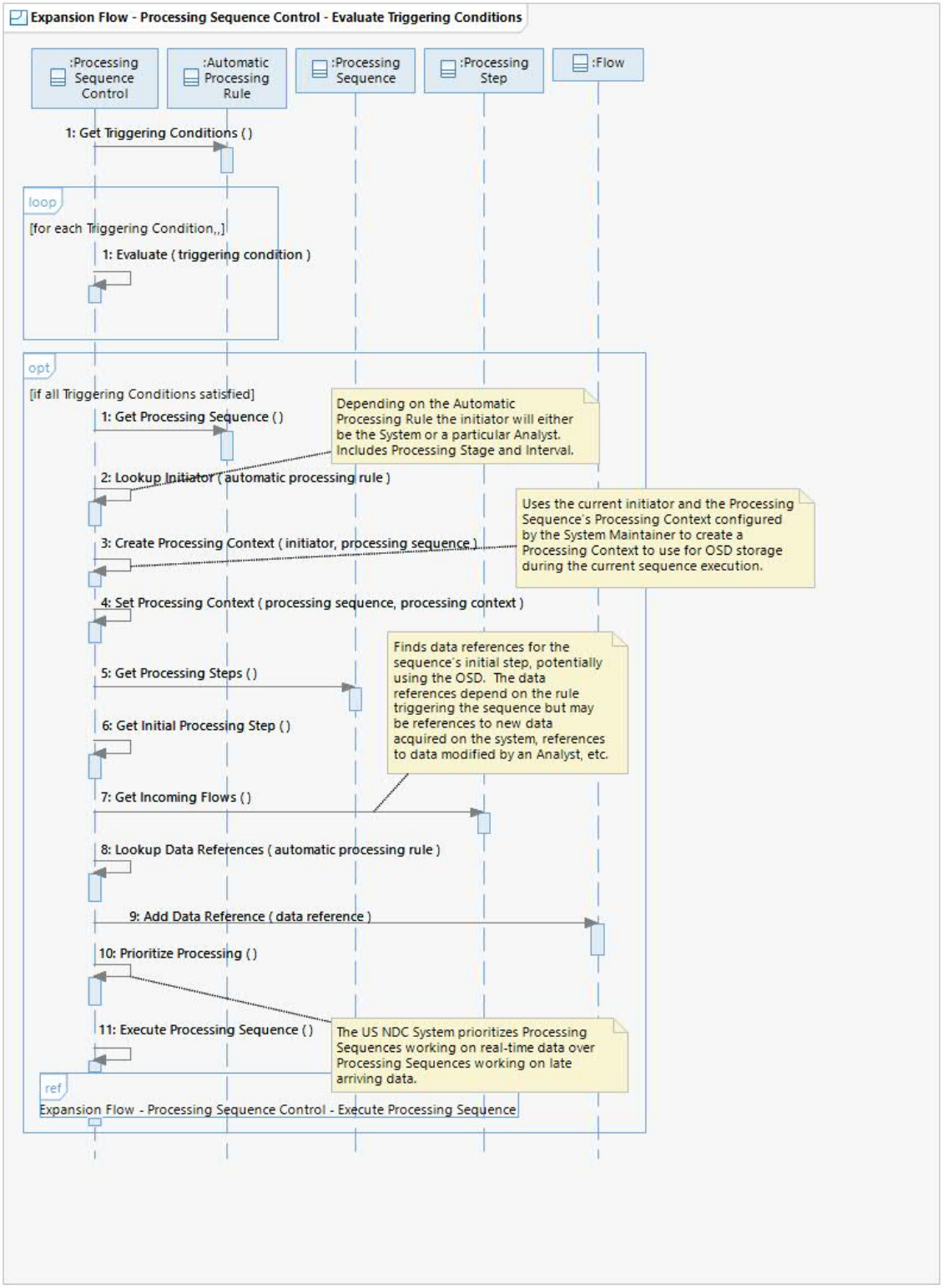


This flow shows how Processing Sequence Control periodically evaluates the Automatic Processing Rules to trigger Processing Sequence execution.

Operation Descriptions

None

Expansion Flow - Processing Sequence Control - Evaluate Triggering Conditions



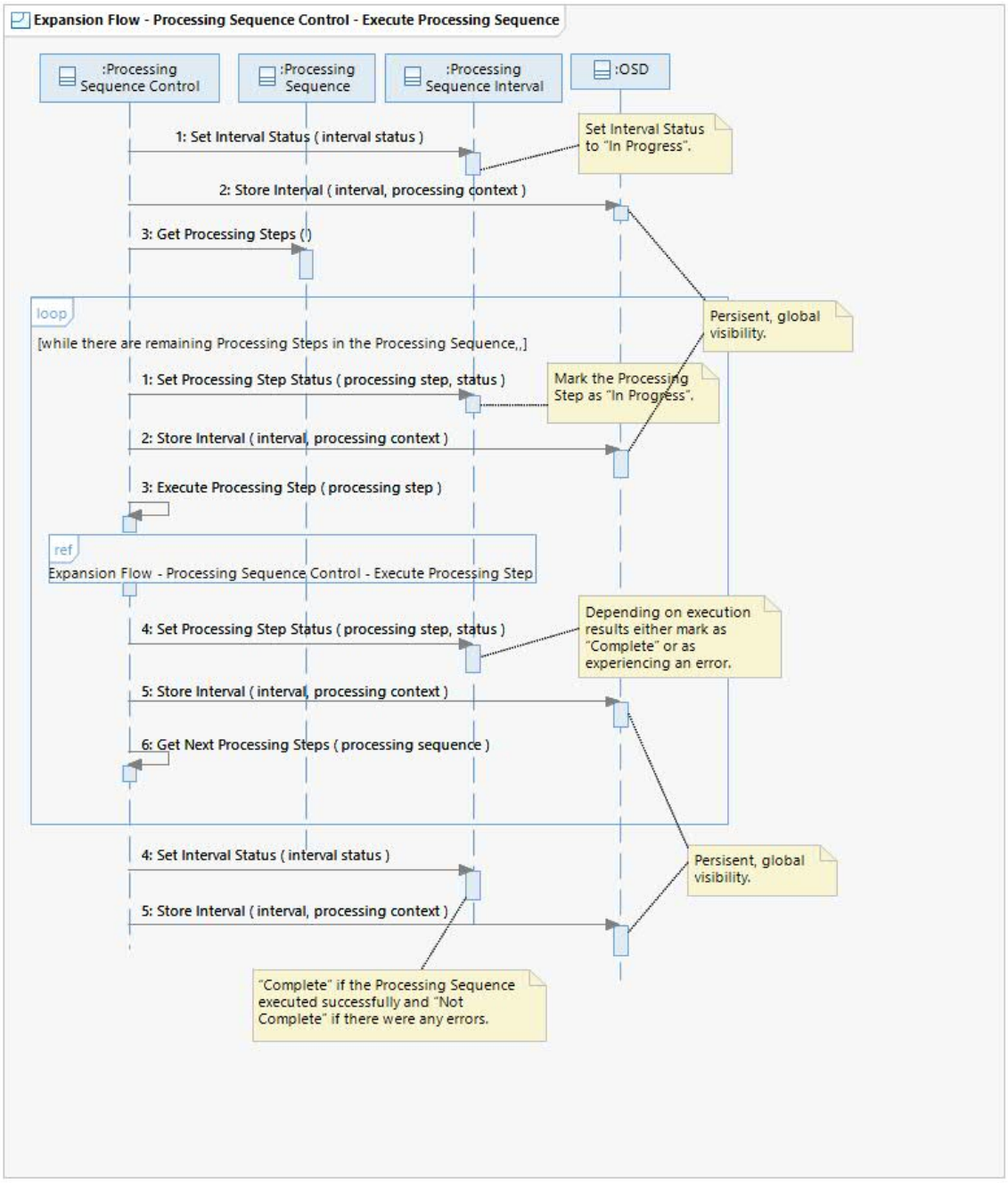
This flow shows Processing Sequence Control evaluating the Triggering Conditions for an Automatic Processing Rule. If all of the Triggering Conditions are satisfied then Processing

Sequence Control executes the Processing Sequence associated to the Automatic Processing Rule.

Operation Descriptions

None

Expansion Flow - Processing Sequence Control - Execute Processing Sequence



This flow shows how Processing Sequence Control executes a processing sequence by iteratively executing the Processing Steps in the sequence.

Operation Descriptions

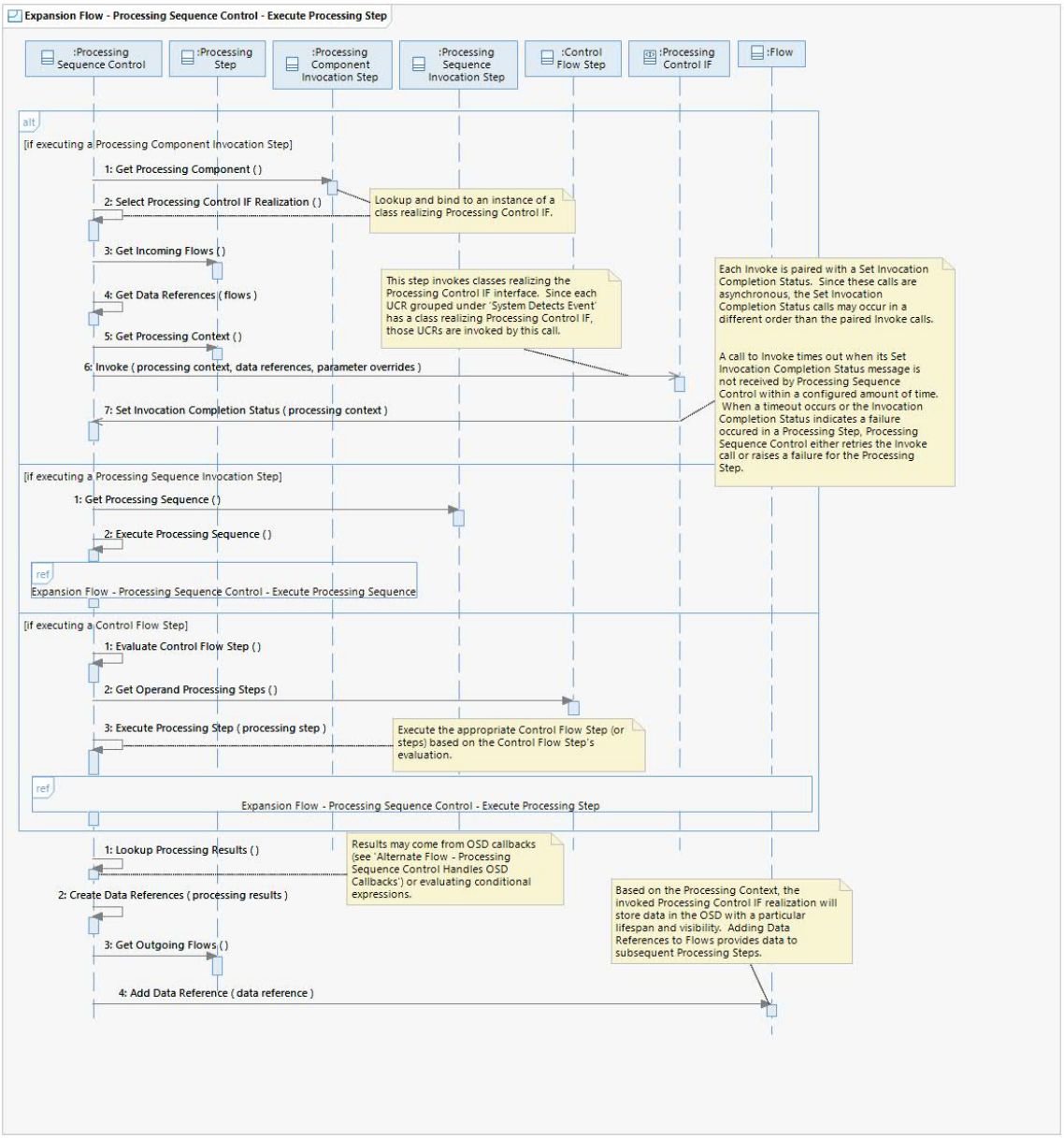
Operation: OSD::Store Interval()
Store the given interval with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Operation: OSD::Store Interval()
Store the given interval with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Operation: OSD::Store Interval()
Store the given interval with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Operation: OSD::Store Interval()
Store the given interval with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Expansion Flow - Processing Sequence Control - Execute Processing Step

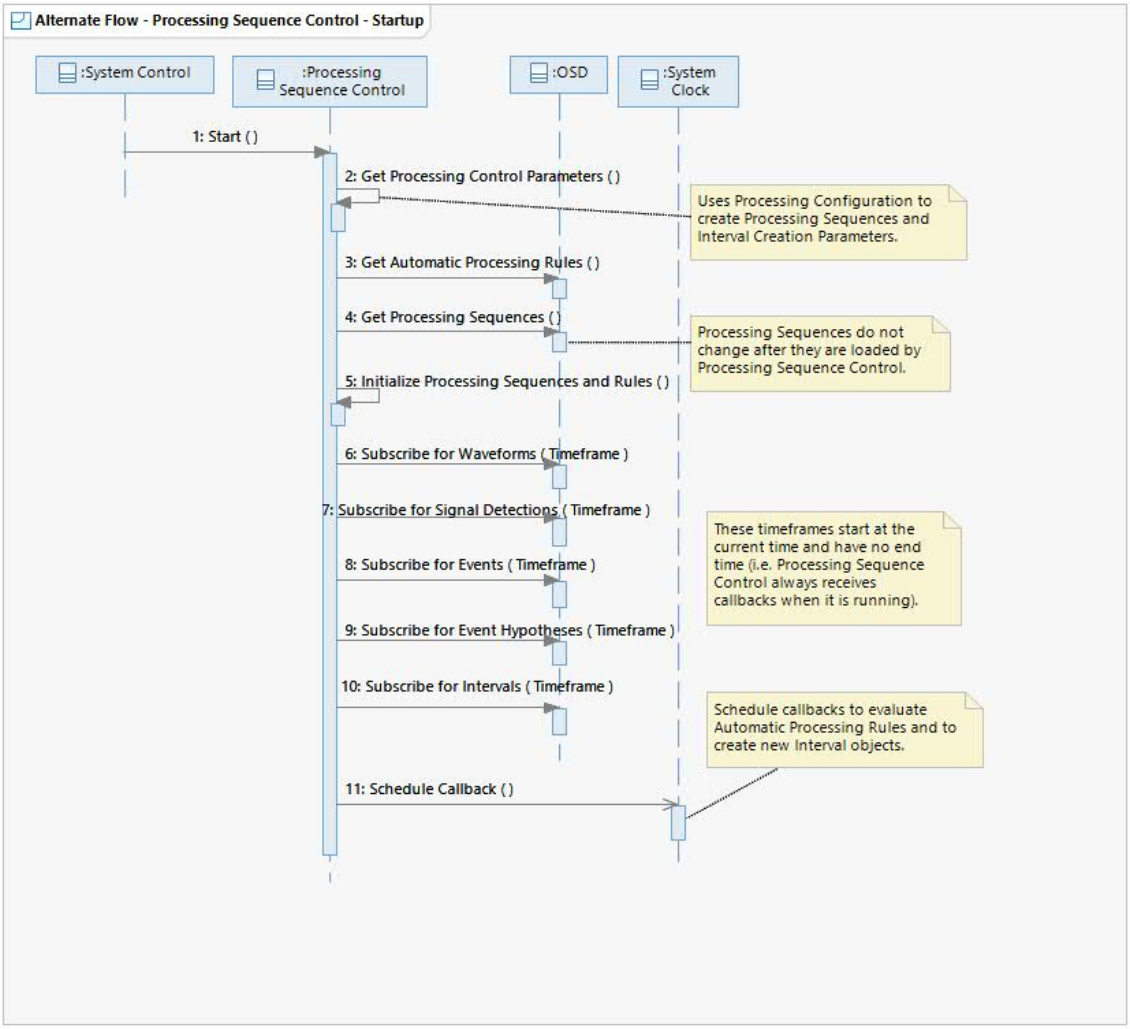


This flow shows how Processing Sequence Control executes a Processing Step. Processing Sequence Control executes a Processing Component Invocation Step by binding to the appropriate realization of the Processing Control IF interface, getting the Data References for the Processing Step, and invoking the Processing Control IF with the Data References. Processing Sequence Control monitors Invocation Completion Status to determine when the Processing Control IF invocation completes. Processing Sequence Control executes a Processing Sequence Invocation Step by initiating execution of that sequence. Processing Sequence Control executes a Control Flow Step by evaluating the Control Flow Step's conditions to determine which of the operand Processing Steps need to be executed and then executing those Processing Steps. Regardless of Processing Step type, after Processing Sequence Control executes the Processing Step it sets Data References to the Processing Step's results on the step's outgoing Flows.

Operation Descriptions

None

Alternate Flow - Processing Sequence Control - Startup



The flow shows how System Control starts Processing Sequence Control. Processing Sequence Control loads each Processing Sequence and subscribes for data updates from the OSD. Processing Sequence Control subscribes for updates to either provide data to Processing Sequences (see 'Alternate Flow - Processing Sequence Control Handles OSD Callbacks' or to remove data from Processing Sequences (see 'Alternate Flow - Processing Sequence Control Terminates Processing for Data Reference'). Processing Sequence Control schedules regular callbacks from the System Clock. The first set of callbacks trigger Processing Sequence Control to evaluate Automatic Processing Rules to determine which Processing Sequences to execute (see “Main Flow”). The second set of callbacks trigger Processing Sequence Control to create new Interval objects (see “Alternate Flow – Processing Sequence Control – Create Intervals”).

Operation Descriptions

Operation: OSD::Subscribe for Signal Detections()
Subscribe for updates regarding Signal Detection creations, modifications, and associations occurring within the specified timeframe. This includes updates for new or modified unassociated Signal Detections.

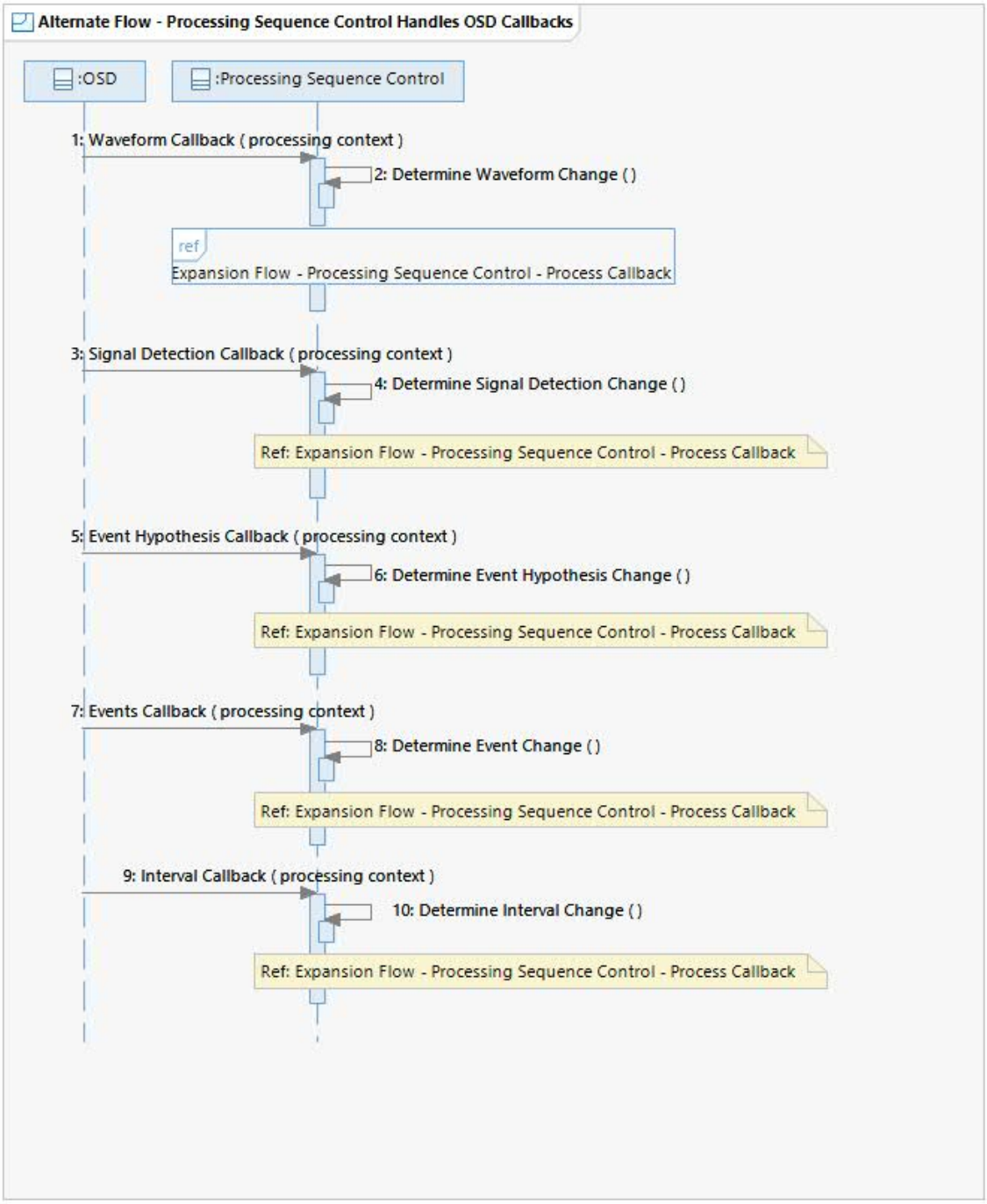
Operation: OSD::Subscribe for Waveforms()
Subscribe for updates regarding raw and derived waveforms occurring within a specified timeframe. This includes information about what waveforms have been acquired by the System as well as what derived waveforms have been formed, but does not include the actual waveform data.

Operation: OSD::Subscribe for Events()
Subscribe for changes to Event objects within the given timeframe. Callbacks are invoked on subscribers any time an Event within the timeframe is added or modified.

Operation: Processing Sequence Control::Initialize Processing Sequences and Rules()
Initializes the Automatic Processing Rules and the associated Processing Sequences so that Processing Sequence Control can evaluate the rules and initiate the sequences when the System state satisfies those rules.

Operation: OSD::Subscribe for Intervals()
Subscribe for changes to Interval objects that overlap with the given timeframe. Interval objects track the active analysts and completion status of intervals corresponding to processing stages and processing activities within processing stages. Callbacks are invoked on subscribers any time the set of active analysts or completion status for an Interval changes.

Alternate Flow - Processing Sequence Control Handles OSD Callbacks

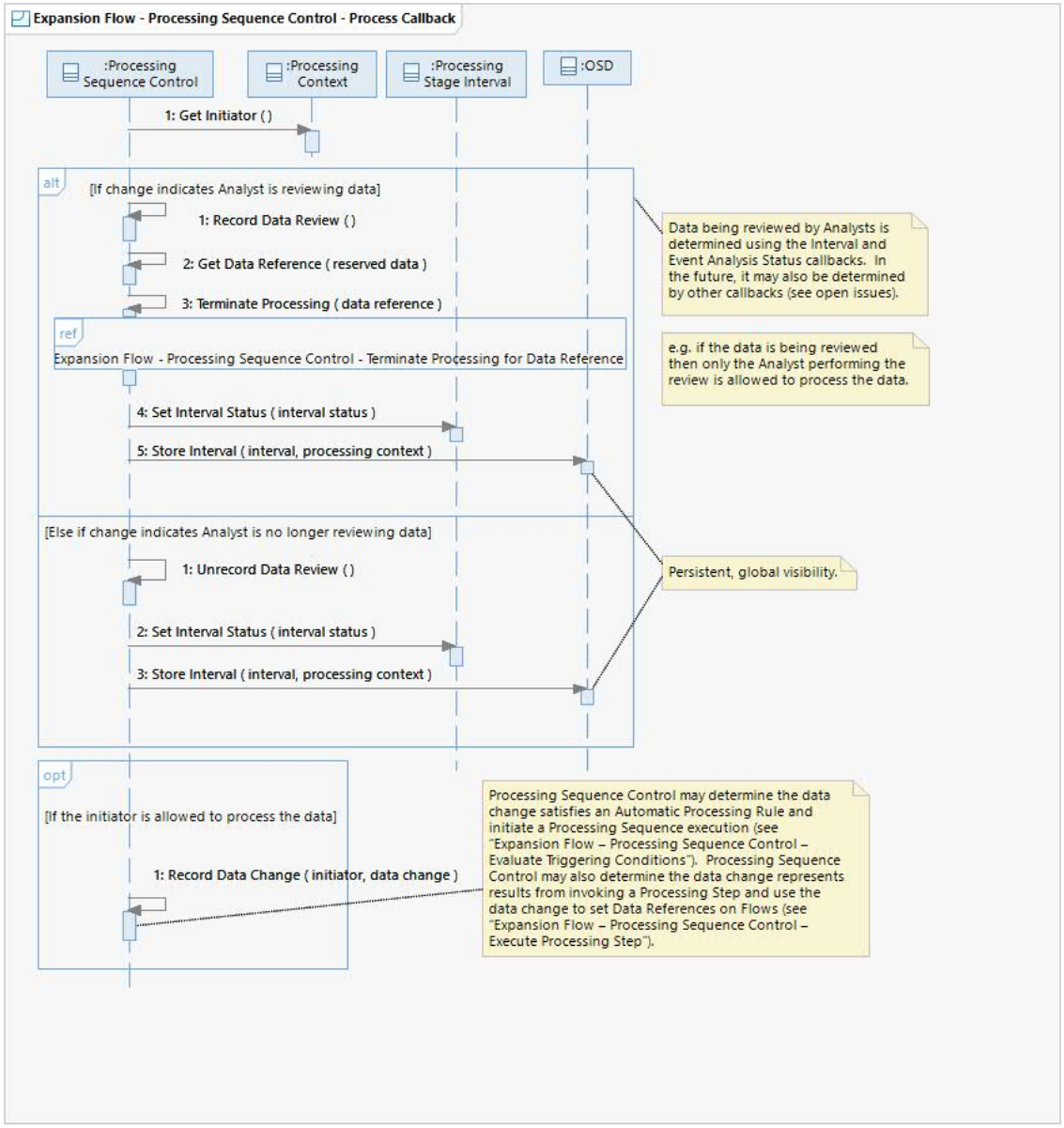


This flow shows how Processing Sequence Control processes OSD callbacks. OSD callbacks occur when data is stored in the OSD with any lifespan and visibility settings. In addition to supporting pipeline processing, this allows the initiation and running of Processing Sequences as a result of Analyst interactions with data that the Analyst has not selected to store permanently and/or make globally visible.

Operation Descriptions

None

Expansion Flow - Processing Sequence Control - Process Callback



This flow shows how Processing Sequence Control processes data changes from OSD callbacks. Processing Sequence Control monitors data changes to determine which Processing Sequences are allowed to process which types of (e.g. Processing Sequences for System processing can only use data that is not open for Analyst review). Processing Sequence Control records data changes and uses those changes to initiate Processing Sequence execution (see "Expansion Flow – Processing Sequence Control – Evaluate Triggering Conditions") and to pass data between the Processing Steps in a Processing Sequence (see "Expansion Flow – Processing Sequence Control – Execute Processing Step").

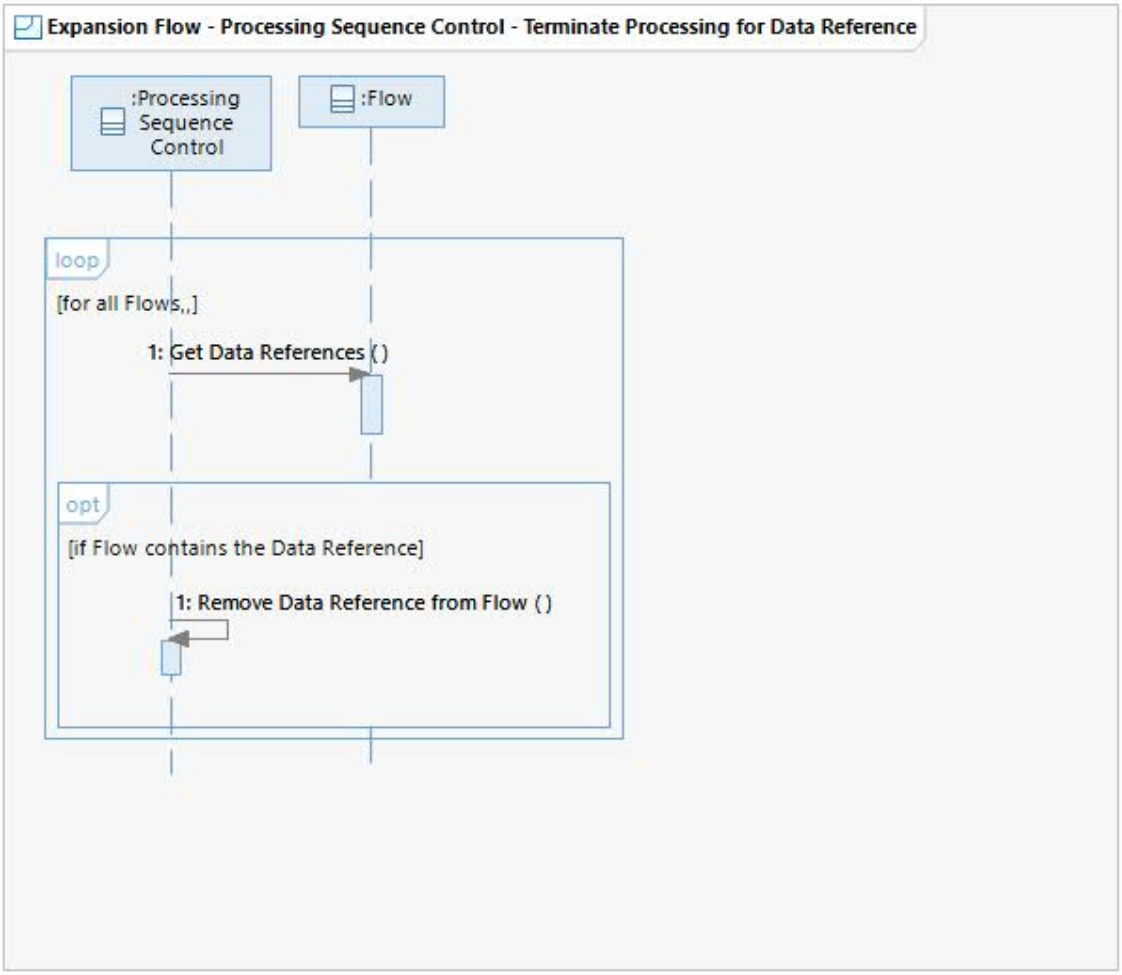
Processing Sequence Control sets the automatic Processing Stage Interval's status based on OSD callbacks. If a callback indicates an Analyst has selected to review the data in an interactive Processing Stage and automatic processing for the stage is complete then the Interval is set to "Complete" (which occurs when all of the associated Processing Sequence Intervals are also "Complete") or "Not Complete" (when one or more of the associated Processing Sequence Intervals are not "Complete"). If a callback indicate an Analyst has completed review for an Interactive Processing Stage and a new automatic Processing Stage is initiated then the Processing Stage Interval is set to "In Progress".

Operation Descriptions

Operation: OSD::Store Interval()
Store the given interval with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Operation: OSD::Store Interval()
Store the given interval with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Expansion Flow - Processing Sequence Control - Terminate Processing for Data Reference

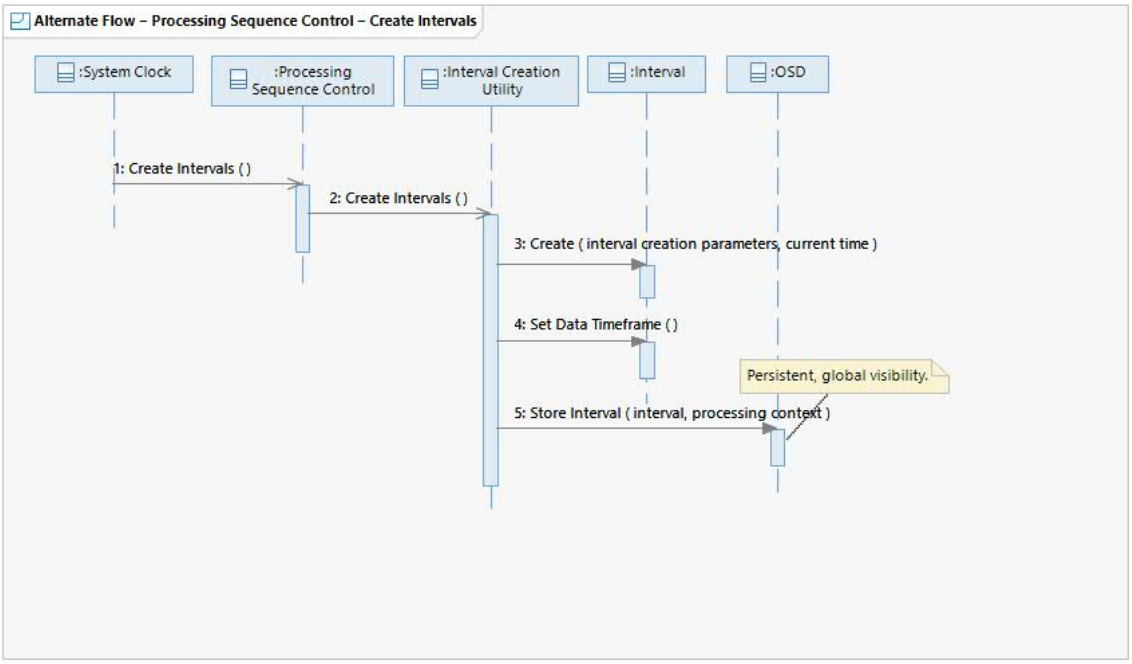


This flow shows how Processing Sequence Control terminates processing for data that has been opened for Analyst review (e.g. the Analyst reserved an event for analysis (see "Refines Event" UCR), the Analyst opened a time interval to scan waveforms and unassociated signal detections (see "Scans Waveforms and Unassociated Detections" UCR) by removing all Data References to the opened data. This flow is invoked when Processing Sequence Control receives a callback from the OSD indicating the event or time interval is open for Analyst review (see "Alternate Flow - Processing Sequence Control Handles OSD Callbacks"). This flow does not terminate or interrupt any Processing Steps that are processing the data when this flow is invoked. Those Processing Steps continue processing, and Processing Sequence Control removes the data from further processing when the Processing Steps complete their processing. The results of running these Processing Steps have no effect on the data opened for analysis.

Operation Descriptions

None

Alternate Flow – Processing Sequence Control – Create Intervals



Operation Descriptions

Operation: OSD::Store Interval()
Store the given interval with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

State Machine Diagrams

None

SSD Mappings

General:

S-1296: [*Threshold*] The System shall store the processing time period(s) during which each Waveform QC Mask was applied to the underlying waveform data.

S-1297: [*Threshold*] The System shall store the Waveform QC Masks applied to the waveform data used for each waveform processing operation.

S-1298: [*Threshold*] The System shall store the channel masked by each Waveform QC Mask.

S-1299: [*Threshold*] The System shall store the identity of the user or processing stage creating

each Waveform QC Mask.

S-1300: [*Threshold*] The System shall store the identity of the user or processing stage modifying each Waveform QC Mask.

S-1301: [*Threshold*] The System shall store the identity of the user or processing stage removing each Waveform QC Mask.

S-1302: [*Threshold*] The System shall store the time of each Waveform QC Mask creation.

S-1303: [*Threshold*] The System shall store the time of each Waveform QC Mask removal.

S-1304: [*Threshold*] The System shall store the time of each Waveform QC Mask modification.

S-1305: [*Threshold*] The System shall store the type of error being masked for each automatically created Waveform QC Mask.

S-1386: [*Threshold*] The System shall store the beam definition parameters for all beams.

S-1387: [*Threshold*] The System shall store continuous beams for virtual event hypotheses for predefined locations.

S-1393: [*Threshold*] The System shall store all derived channels related to one or more signal detections.

S-1394: [*Threshold*] The System shall store derived waveform data with no related signal detections for the Operational Processing Time Period.

S-1405: [*Threshold*] The System shall create an origin beam steered to an event hypothesis' hypocenter and a seismic array station's predicted first P arrival time whenever a seismic array station lacks a first P signal detection association.

S-1421: [*Threshold*] The System shall store all signal detections.

S-1438: [*Threshold*] The System shall store time domain measurements.

S-1450: [*Threshold*] The System shall store polarization feature measurements.

S-1465: [*Threshold*] The System shall store frequency domain waveform measurements.

S-1486: [*Threshold*] The System shall store fk spectra measurements.

S-1549: [*Threshold*] The System shall perform late network signal association during the operational processing time period.

S-1556: [*Threshold*] The System shall store all event hypotheses formed by the System.

S-1557: [*Threshold*] The System shall store all signal detection associations for each event hypothesis stored by the System.

S-1576: [*Threshold*] The System shall store the station quality metrics for all stations for each event hypothesis.

S-1580: [*Threshold*] The System shall recompute the event hypothesis quality metric for an event hypothesis when any of the event hypothesis quality statistics used to calculate the event hypothesis quality metric are updated.

S-1588: [*Threshold*] The System shall store the event quality metric for each event hypothesis.

S-1599: [*Threshold*] The System shall compute a new event hypothesis relocation when an automatic process modifies any event hypothesis relocation parameter contributing to that event hypothesis' location.

S-1618: [*Threshold*] The System shall store up to 300 unique event hypotheses for each event.

S-1619: [*Threshold*] The System shall store the confidence level of each computed event hypothesis location uncertainty bound.

S-1620: [*Threshold*] The System shall store the type (i.e., confidence, coverage, or k-weighted with the associated weights) of each location uncertainty bound.

S-1621: [*Threshold*] The System shall store modeling uncertainties for model based predictions of signal detection measurements.

S-1622: [*Threshold*] The System shall store uncertainties for observed signal detection measurements.

S-1623: [*Threshold*] The System shall store the sum squared weighted residual for each event hypothesis location.

S-1624: [*Threshold*] The System shall store the defining/non-defining state for each signal detection measurement associated to a stored event hypothesis.

S-1625: [*Threshold*] The System shall store a preferred event hypothesis for each event for each processing stage.

S-1626: [*Threshold*] The System shall store the processing stage during which each event hypothesis location was created.

S-1627: [*Threshold*] The System shall store the processing stage during which an event hypothesis is modified.

S-1628: [*Threshold*] The System shall store the processing stage that rejected an event.

S-1663: [*Threshold*] The System shall store uncertainties for all event hypothesis magnitude estimates.

S-1664: [*Threshold*] The System shall store each single station magnitude estimate for each event hypothesis.

S-1665: [*Threshold*] The System shall store each network magnitude estimate for each event hypothesis.

S-1666: [*Threshold*] The System shall store the defining/non-defining state for each station magnitude associated to a stored event hypothesis.

S-1816: [*Threshold*] The System shall store the earth model and version used to compute an earth model prediction.

S-1817: [*Threshold*] The System shall store the corrections applied to earth model predictions.

S-1818: [*Threshold*] The System shall store the correction surface used to correct an earth model prediction.

S-1819: [*Threshold*] The System shall store the predicted slowness computed from a basemodel.

S-1820: [*Threshold*] The System shall store the uncertainties of a predicted slowness computed using a basemodel.

S-1821: [*Threshold*] The System shall store the predicted azimuths computed using a phase-specific basemodel.

S-1822: [*Threshold*] The System shall store the uncertainties of predicted azimuths computed using a basemodel.

S-1823: [*Threshold*] The System shall store the predicted travel-times computed from a basemodel.

S-1824: [*Threshold*] The System shall store the uncertainties of predicted travel-times computed using a basemodel.

S-1842: [*Threshold*] The System shall store predicted amplitude attenuation.

S-1843: [*Threshold*] The System shall store predicted amplitude attenuation uncertainties.

S-1860: [*Threshold*] The System shall process waveform data within a configurable processing time interval when a configurable percentage of data is available.

S-1861: [*Threshold*] The System shall process all available alphanumeric data within a configurable processing time interval.

S-1862: [*Threshold*] The System shall run a previously configured group of operations whenever the triggering event for that group of operations occurs.

S-1872: [*Threshold*] The System shall provide the Analyst the capability to interrupt automated event hypothesis processing to analyze data if configured.

S-1967: [*Threshold*] The System shall store results from all stages of data processing.

S-2042: [*Threshold*] The System shall store automatic and interactive processing parameters in the database.

S-2043: [*Threshold*] The System shall store automatic and interactive processing results.

S-2044: [*Threshold*] The System shall store the relation of processing results to processing parameters in the database.

S-2166: [*Threshold*] The System shall automatically process late-arriving waveform data within one (1) minute of receipt by the Data Processing Partition.

S-2171: [*Threshold*] The System shall prioritize the processing of real time data over the processing of late arriving data.

S-2172: [*Threshold*] The System shall automatically initiate data processing within 5 minutes of data acquisition on the Data Processing Partition.

S-2173: [*Threshold*] The System shall automatically execute processing of waveform data (i.e., data acquisition, data processing, and data storage).

S-2175: [*Threshold*] The System shall process up to 2000 seismic event hypotheses per day without disruption of the Data Processing Partition.

S-2177: [*Threshold*] The System shall produce an automated event bulletin in near real-time during normal conditions without disrupting operations.

S-2178: [*Threshold*] The System shall produce an automated event bulletin in near real-time during swarm conditions without disrupting operations.

S-2223: [*Threshold*] The System shall store all data and derived processing results to persistent storage as soon as the data and/or derived processing results are available.

S-2417: [*Threshold*] The System shall store hydroacoustic signal detection groups

S-5610: [*Threshold*] The Data Processing Partition shall access and process all waveform data

stored on the system.

S-5715: [*Threshold*] The System shall store wind velocity (including uncertainty) computed from meteorological models.

S-5716: [*Threshold*] The System shall store temperature (including uncertainty) computed from meteorological models.

S-5717: [*Extensibility*] The System shall store gravity wave corrections to temperature predictions.

S-5720: [*Threshold*] The System shall store spectrograms.

S-5722: [*Threshold*] The System shall store power spectral density.

S-6469: [*Threshold*] The System shall store detection feature maps.

S-6521: [*Threshold*] The System shall store seed events.

S-6522: [*Threshold*] The System shall store seed event quality.

IDC Specific:

S-5795: [*Threshold*] The System shall compute Event Consistency checks when an event hypothesis is saved.

Notes

General:
1. UCR 'Monitors System Performance' describes how system processes are monitored.

2. The Event Analyzer control class appears in this UCR as a control class that can be used to implement processing sequence control logic that is both too specific for the Processing Sequence Control mechanism to implement and which is not available in other control classes. Since the details of this additional logic do not appear in any UC, the analysis model will not further describe Event Analyzer.

IDC Specific:
1. Event Screening Control also realizes Processing Control IF, but is not shown in the Processing Control IF Class Diagram.

2. The System Maintainer uses Defines Processing Sequence Display (see 'Defines Processing Sequence' UCR) to configure a processing sequence to perform event consistency calculations after an event is saved (see 'System Accesses Event Consistency' UCR).

3. A Control class or Plugin class may request an auxiliary seismic Waveform by querying the OSD for the Waveform. If the Waveform is not found, the OSD uses Station Data Receiver Control to request the waveform segment from the Station (see 'System Receives Station Data' UCR).

This page intentionally left blank.

IDC Use Case Realization Report

UCR-02.08 System Refines Event Location

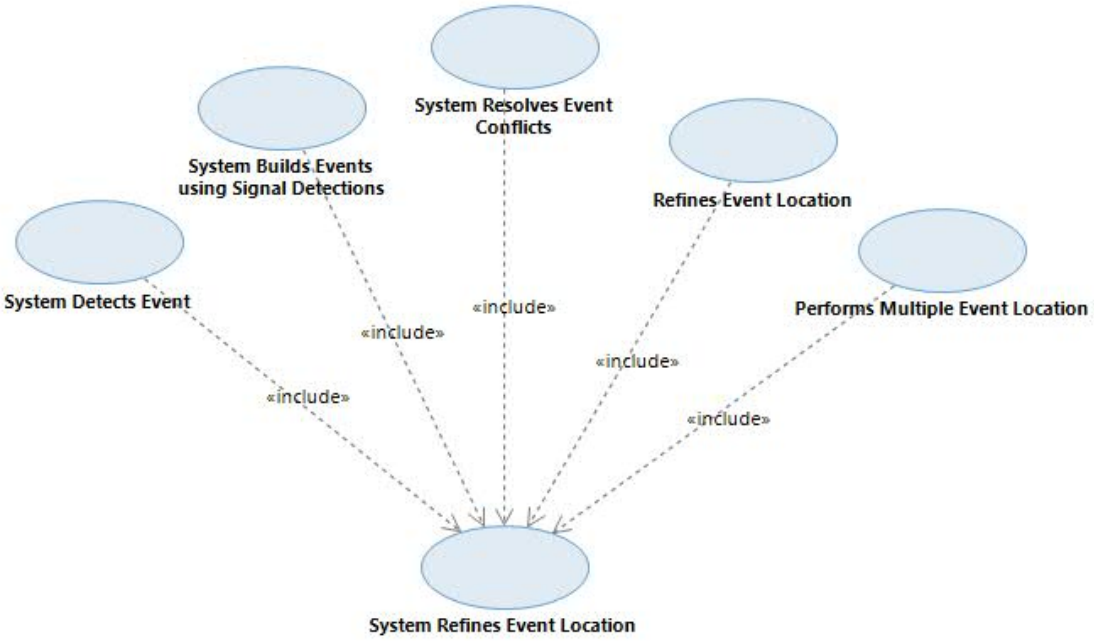
Use Case Description

This use case describes how the System refines event hypothesis location solutions using single event or multiple event algorithms. Event locations can be absolute or relative. The System locates events by finding the event location minimizing the difference between signal detection feature measurements and signal detection feature predictions (see 'System Measures Signal Features' UC). The System references both empirical knowledge from past events and geophysical models to form the signal detection feature predictions (see 'System Predicts Signal Features' UC). The System also computes an uncertainty bound for each event hypothesis location solution describing a region bounding the event hypothesis' hypocenter and origin time at a particular confidence level. The System creates a variety of location solutions for each event hypothesis. These location solutions vary from one another in either the input parameters the System uses or in the location solution components the System restrains to fixed values (e.g., depth) during event location calculations. The System computes location solutions using input parameters configured by the System Maintainer (see 'Configures Processing Components' UC). The Analyst has the option to override input parameters originally configured by the System Maintainer (see 'Refines Event Location' UC).

Architecture Description

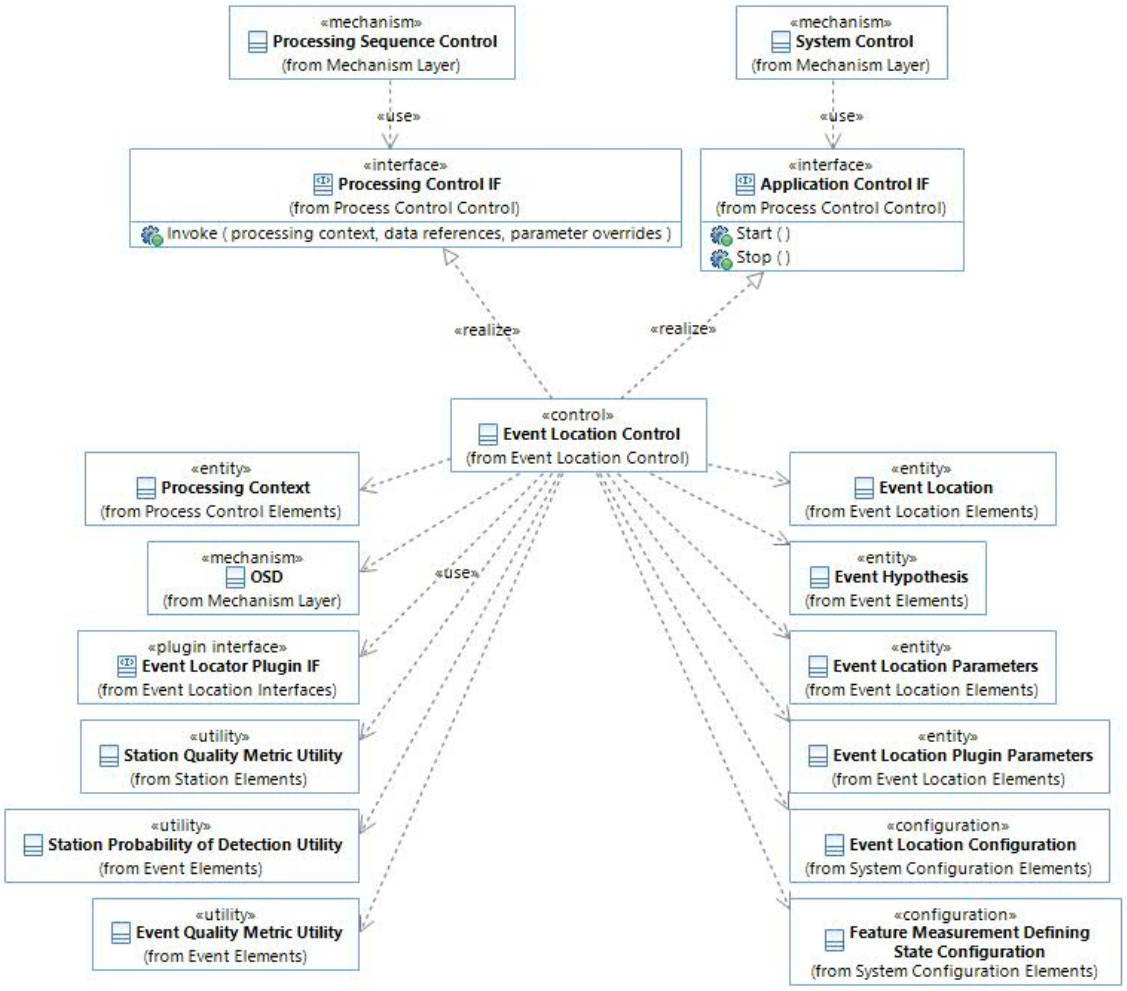
The Event Location Control class is responsible for controlling event location computations. Event Location Control may be invoked by Processing Sequence Control as part of executing a step in a Processing Sequence (see 'System Detects Event' UCR), manually invoked by an Analyst as part of refining an event (see 'Refines Event Location' UCR), or manually invoked by a Researcher (see 'Performs Multiple Event Location' UCR). Following the plugin and parameter pattern described in the Architecture Document, Event Location Control uses an Event Locator Plugin to perform the event location calculations. Each of the Event Locator Plugin implementations in the System realize a common plugin interface. The specific Event Locator Plugin used varies dynamically at runtime based on the Event Location Parameters. When invoked from Processing Sequence Control, Event Location Control builds up the Event Location Plugin Parameters to be passed to the Event Locator Plugin, selects and invokes the appropriate Event Locator Plugin based on those parameters, updates the Event Hypotheses with the results, and stores the Event Hypotheses via the OSD mechanism.

Use Case Diagram



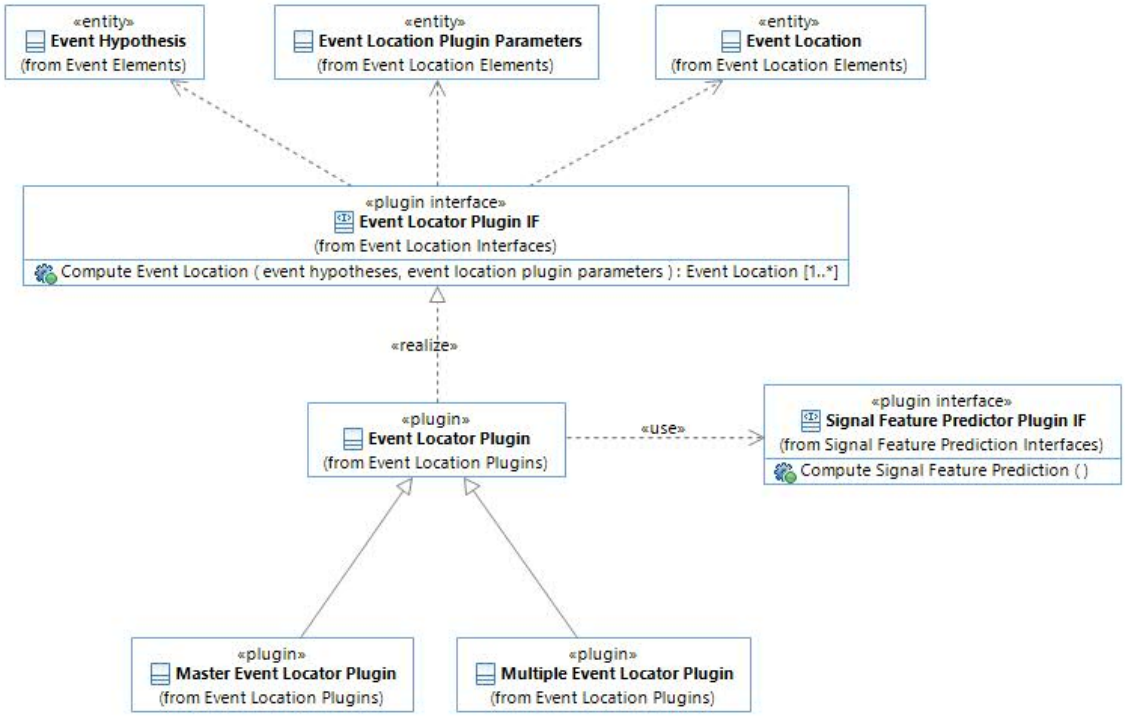
Class Diagrams

Classes - Event Location Control



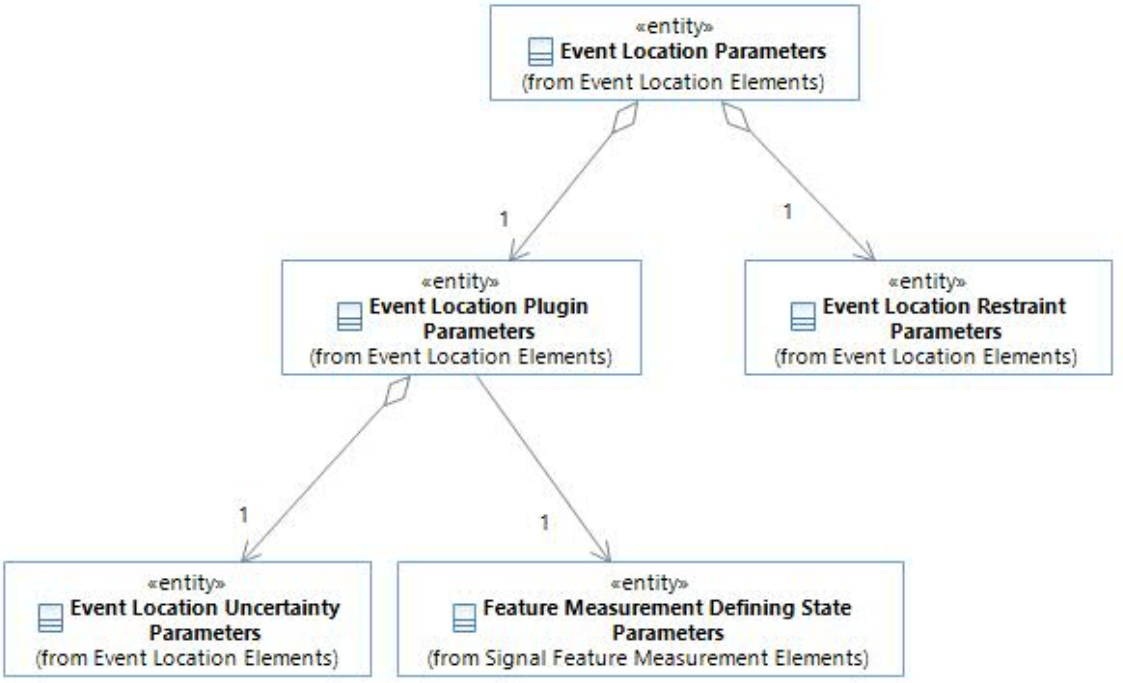
This diagram shows the Event Location Control class and related classes. Event Location Control implements the Processing Control and Application Control interfaces so that it can be started and stopped by System Control and used as part of a processing sequence. Event Location Control depends on entity classes relevant to Events, such as Event Hypothesis and Event Location.

Classes - Event Locator Plugin IF



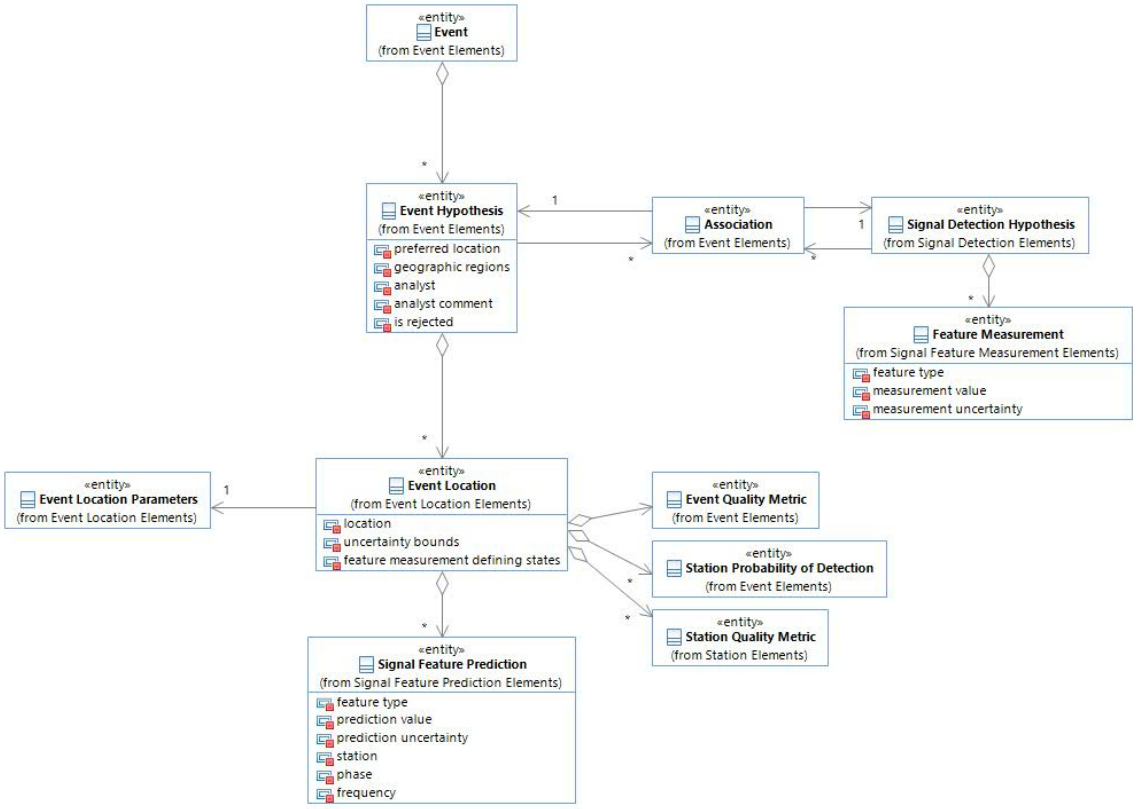
This diagram shows the Event Locator Plugin IF interface, which defines the common interface that all event locator plugins must realize. The Master Event Locator Plugin and Multiple Event Locator Plugin represent two plugins implementation, but more implementations may exist. An Event Locator Plugin may access plugins implementing the Signal Feature Predictor Plugin IF. For example, a locator may use a Signal Feature Predictor Plugin when calculating feature measurement residuals. See 'System Predicts Signal Features' UCR for details about Signal Feature Predictor Plugin.

Classes - Event Location Parameters



This diagram shows details of the Event Location Parameters class. The Event Location Parameters class is used by Event Location Control to determine general behavior of location calculations and the Event Location Plugin Parameters class is provided as input to the Event Locator Plugin to control specific algorithm behavior. Event Location Control creates the parameters from the Event Location Configuration preconfigured by the System Maintainer (see "Configures Processing Components" UCR). The Analyst may override the parameters (see 'Refines Event Location' UCR).

Classes - Event Hypothesis



This diagram shows the portions of an Event Hypothesis that are used to compute Event Location as well as the portions that are computed by the algorithm. The event location algorithm analyzes Feature Measurements of Signal Detections Hypotheses associated to one or more Event Hypotheses. The algorithm uses the station location associated with each Signal Detection Hypothesis as well as features of the detection to compute the Event Location, which is the primary output of the event location algorithm. Multiple different Event Locations (each with a different set of Event Location Restraints, as stored in the Event Location Parameters associated to that Event Location) may be computed for each Event Hypothesis. The Event Location Parameters class captures the parameters that were provided as input to the event location algorithm, enabling subsequent Analysts to recompute the location with the same parameters.

Class Descriptions

<<configuration>> Event Location Configuration
Default event location configuration as configured by the System Maintainer. Contains configuration about which Event Locator Plugins the System should invoke, as well as the types of location uncertainty bounds and restrained locations the System should compute. The Analyst may override the Event Location Parameters computed from this configuration.

<<configuration>> Feature Measurement Defining State Configuration
Represents all signal detection feature measurement defining state configuration in the system. This includes all configurations used to determine which signal detection feature measurements

are by default defining and non-defining for various types of system calculations.

<<control>> Event Location Control
Responsible for controlling the event location computation. Retrieves necessary data, invokes the appropriate Event Locator Plugin to compute the new location, and stores the result.

<<entity>> Association
Represents an association between a Signal Detection Hypothesis and an Event Hypothesis.

<<entity>> Event
Represents information about an Event. Keeps track of all the Event Hypotheses for the Event, which Event Hypothesis is the preferred one for each processing stage, the active analysts for the Event (i.e. whether the Event is under "active review"), whether the Event is "complete" for each processing stage, and other Event-related information.

<<entity>> Event Hypothesis
Represents geophysical information about an Event as determined by an Analyst or through pipeline processing. There can be multiple Event Hypotheses for the same Event (e.g. different associated Signal Detection Hypotheses, different location solutions).

<<entity>> Event Location
Represents a computed location for an event.

<<entity>> Event Location Parameters
Represents the parameters that are used by Event Location Control. This includes which Event Locator Plugin to invoke as well as the types of restrained event locations that Event Location Control will invoke the plugin to compute. Initially set by Event Location Control based on the Event Location Configuration defined by the System Maintainer, but the Analyst may select to override parameter values when refining events.

<<entity>> Event Location Plugin Parameters
Represents all parameters passed to an Event Locator Plugin. This includes parameters describing default feature measurement defining states and the types of uncertainty bounds the plugin should compute. May also include parameters specific to the plugin being invoked.

<<entity>> Event Location Restraint Parameters
Represents restraints on the location coordinate spaces (lat, lon, depth or time) for the event location computation. Restraints indicate which coordinate spaces are restrained and the associated restrained value (or value range) to be used for that coordinate.

<<entity>> Event Location Uncertainty Parameters
Represents the type of uncertainty bound (Confidence, Coverage or K-Weighted), confidence level and scaling factor for the locator to use when computing event location uncertainty.

<<entity>> Event Quality Metric
Represents the quality of a single event hypothesis. Event quality is based on a variety of event

quality statistics, possibly including the Station Quality Metric and Station Probability of Detection for Stations with Signal Detections associated to the Event Hypothesis as well as for Stations that did not detect the Event Hypothesis, the algorithms used to detect the event, event location and event location uncertainty, etc.

<<entity>> Feature Measurement
Represents the value and uncertainty of a measured feature of a signal detection.

<<entity>> Feature Measurement Defining State Parameters
Represents defining state parameters for feature measurements. The parameters include the following for each feature measurement for each type of calculation (e.g. location, magnitude, etc.)
- Whether the feature measurement is initially defining or non-defining
- Whether an algorithm is free to toggle the defining state

The analyst can override the defining/non-defining state for these parameters, unless prohibited by the default defining/non-defining state:
- Time, azimuth, or slowness measurement of a signal detection for event hypothesis relocation
- Signal Detection measurements for event hypothesis relocation based on channel
- Signal Detection measurements for event hypothesis relocation based on signal detection phase assignment

<<entity>> Processing Context
Represents the context in which data is being stored and/or processed. This includes the Processing Stage (either automatic or interactive) and Interval performing the processing session (e.g. processed by Analyst vs. processed by System). For Analyst processing, may identify the Analyst work session. For System processing, may identify the Processing Sequence and/or Processing Step being executed (including a way to identify a particular Processing Sequence and Processing Step among the many possible instantiations), the visibility for the results (private vs. global), and the lifespan of the data (transient vs. persistent). This information is needed by the Processing Sequence Control to manage the execution of Processing Sequences, which may execute in the context of an Analyst refining an Event or in the context of the system initiating automatic processing. It is also needed by the Object Storage and Distribution (OSD) mechanism to determine how to store and distribute the data.

<<entity>> Signal Detection Hypothesis
Represents geophysical information about a Signal Detection as determined by an Analyst or through pipeline processing. There can be multiple Signal Detection Hypotheses for the same Signal Detection (e.g. different onset times, different phase labels).

<<entity>> Signal Feature Prediction
Represents a predicted signal feature (e.g., travel time, azimuth, slowness, amplitude, probability of detection) and the associated uncertainties.

<<entity>> Station Probability of Detection
Represents a Station's probability of detecting an Event Hypothesis.

<<entity>> Station Quality Metric
Represents a Station's quality for a particular time. Separate station quality metrics can be computed for a Station with each metric based on different selections of the Station's raw and derived waveforms (e.g. Station Quality Metric could be computed using Waveforms from a Station's raw Channels and a separate quality metric could be computed for a beam created from the Station's waveforms).

<<interface>> Application Control IF
Defines the interface implemented by all <<control>> classes in the system that are controlled by System Control.

<<interface>> Processing Control IF
Defines the interface implemented by all <<control>> classes in the system that are controlled by the Processing Sequence Control <<mechanism>>. <<control>> classes realize this common interface to support configurable processing sequence definition and execution. Processing Sequence Control uses the Invoke() operation declared in Processing Control IF to call <<control>> classes while executing processing sequences. When called in this way the <<control>> classes operate on the provided data (e.g. event hypotheses, signal detections, etc.) using either default parameters configured by the System Maintainer and loaded by the <<control>> class on startup or override parameters provided to the Invoke() operation.

<<mechanism>> OSD
Represents the Object Storage and Distribution mechanism for storing and distributing data objects internally within the system.

<<mechanism>> Processing Sequence Control
Mechanism for executing and controlling processing sequences configured by the System Maintainer.

<<plugin interface>> Event Locator Plugin IF
Standard interface for all Event Locator plugins. All Event Locator plugins in the system realize this interface.

<<plugin interface>> Signal Feature Predictor Plugin IF
Standard interface for all Signal Feature Predictor plugins. All Signal Feature Predictor plugins in the system realize this interface. Plugins that implement Signal Feature Predictor IF may predict different types of signal features, such as travel time, azimuth, slowness, amplitude, and probability of detection.

<<plugin>> Event Locator Plugin
A nominal class representing Event Locator Plugin implementations that may be plugged in to the system behind the Event Locator IF plugin interface. Event Locators are responsible for calculating event locations. Configuration for specific plugin implementations include the settings for controlling their behavior (e.g. settings for max number of iterations, which Signal Feature Predictor Plugin and Earth Model Plugin to use, etc.).

<<plugin>> Master Event Locator Plugin
Specialization of Event Locator Plugin that locates an event based on a specified "master event" (i.e. Master Event Relocation). Configuration for this plugin includes information required to select which master events to use (potentially based on geographic region), which signal detections and feature measurements to use, etc. Parameters for this plugin may include a particular master event, the location and location uncertainty of the master event, and feature measurements and associated uncertainties pertinent to location computation for signal detections associated to the master event that also exists on the event under refinement (i.e. signal detections with the same channel and phase).

<<plugin>> Multiple Event Locator Plugin
A specialization of Event Locator Plugin that simultaneously locates a set of two or more event hypotheses relative to each other (i.e. Multiple Event Relocation). Configuration for this plugin includes information required to select which events to use (potentially based on geographic region), which signal detections and feature measurements to use, etc. Parameters for this plugin may include the set of event hypotheses to relocate, feature measurements (including associated uncertainties) pertinent to the location computation, and a ground truth or fixed location used as a starting location when calculating absolute locations for each relocated event hypothesis.

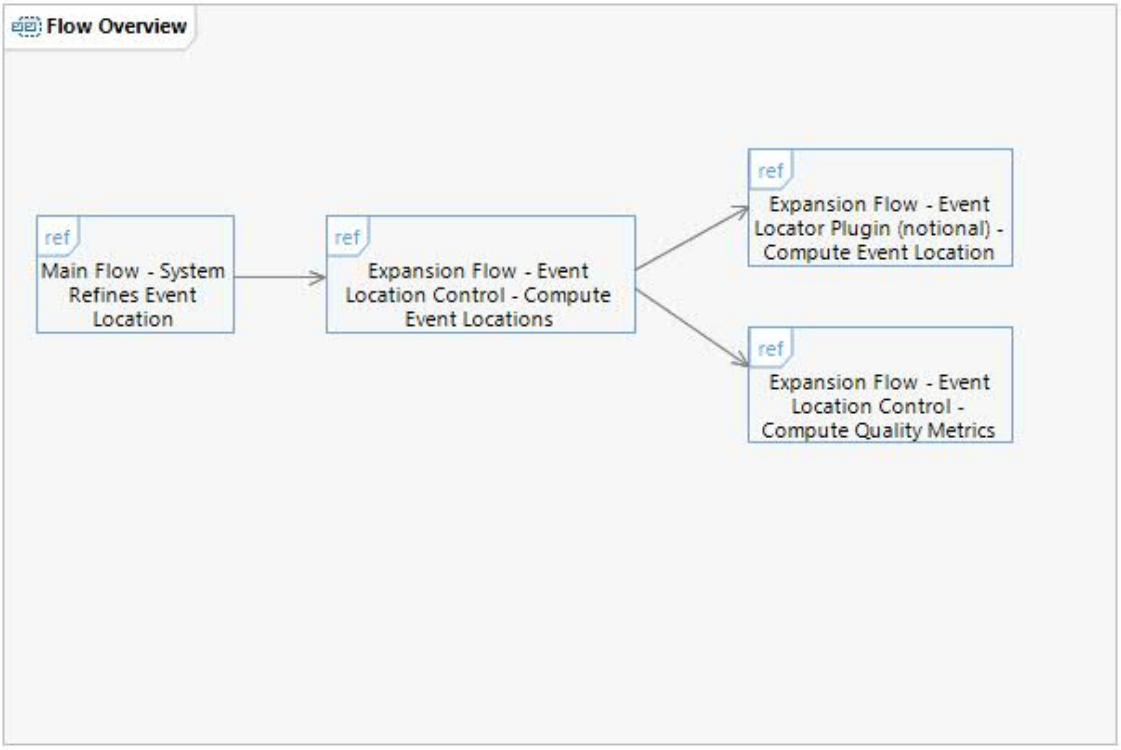
<<utility>> Event Quality Metric Utility
Utility class that computes the event quality metric. The quality metric may be based on how well an event hypothesis meets the event definition criteria, the Event Hypothesis Creation Method, etc.

<<utility>> Station Probability of Detection Utility
Utility that computes station probability of detecting events. The utility may base this probability on either modeled noise or on actual noise recorded at a station.

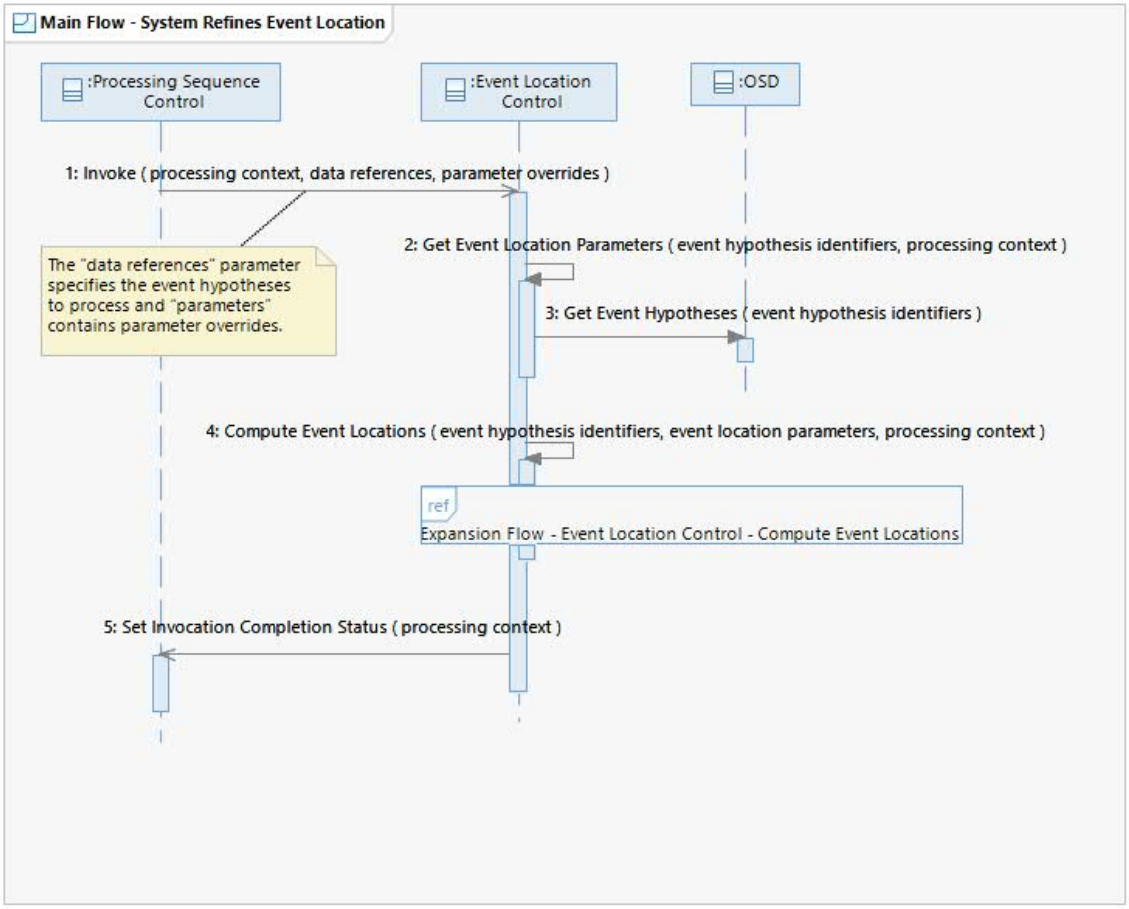
<<utility>> Station Quality Metric Utility
Utility class that computes the station quality metric. Computes the metric for stations using waveforms, SOH information, Waveform QC Masks, etc. and also computes the metric for configured derived channels (e.g. detection beams). Can compute a continuous station quality metric for use in performance monitoring and a different but related station quality metric based on an event hypothesis.

Sequence Diagrams

Flow Overview



Main Flow - System Refines Event Location

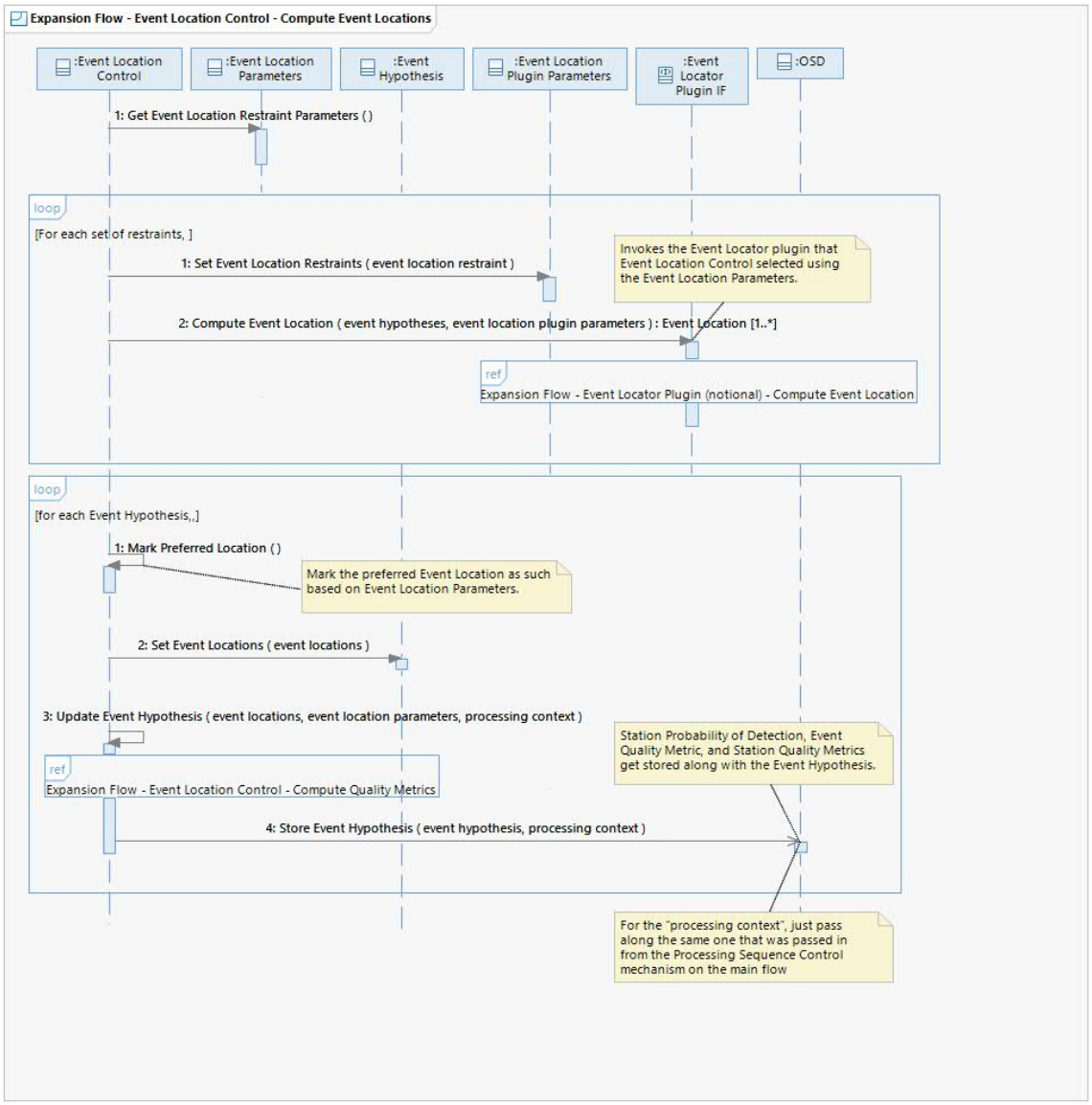


This flow shows how the system refines event location. This flow is stimulated by the Processing Sequence Control mechanism as part of executing an automatic processing sequence. The precise triggering conditions for such sequences are configured by the System Maintainer (see "Defines Processing Sequence" UCR).

Operation Descriptions

Operation: Event Location Control::Get Event Location Parameters()
Creates the Event Location Parameters for each Event Hypothesis based on defaults configured by the System Maintainer (represented by the Event Location Configuration class) and the Event Location Parameters most recently used by either the System or an Analyst to locate the Event Hypothesis' preferred location. Default parameters may augment parameters used during the most recent location calculation if the event has changed (e.g. signal detection associations have been added to or removed from the hypothesis since the last time its location was computed).

Expansion Flow - Event Location Control - Compute Event Locations



This flow shows how the Event Location Control invokes the Event Locator Plugin to compute locations for each provided event hypothesis. The control class selects which Event Locator Plugin to invoke based on the event locator type specified in the Event Location Parameters and then invokes the Event Locator Plugin through the Event Locator Plugin IF interface. The Event Locator Plugin returns Event Locations. Event Location Control updates the Event Hypotheses with the Event Locations. Note that this flow may also be invoked directly from the Refines Event Location Display (see 'Refines Event Location' UCR).

Operation Descriptions

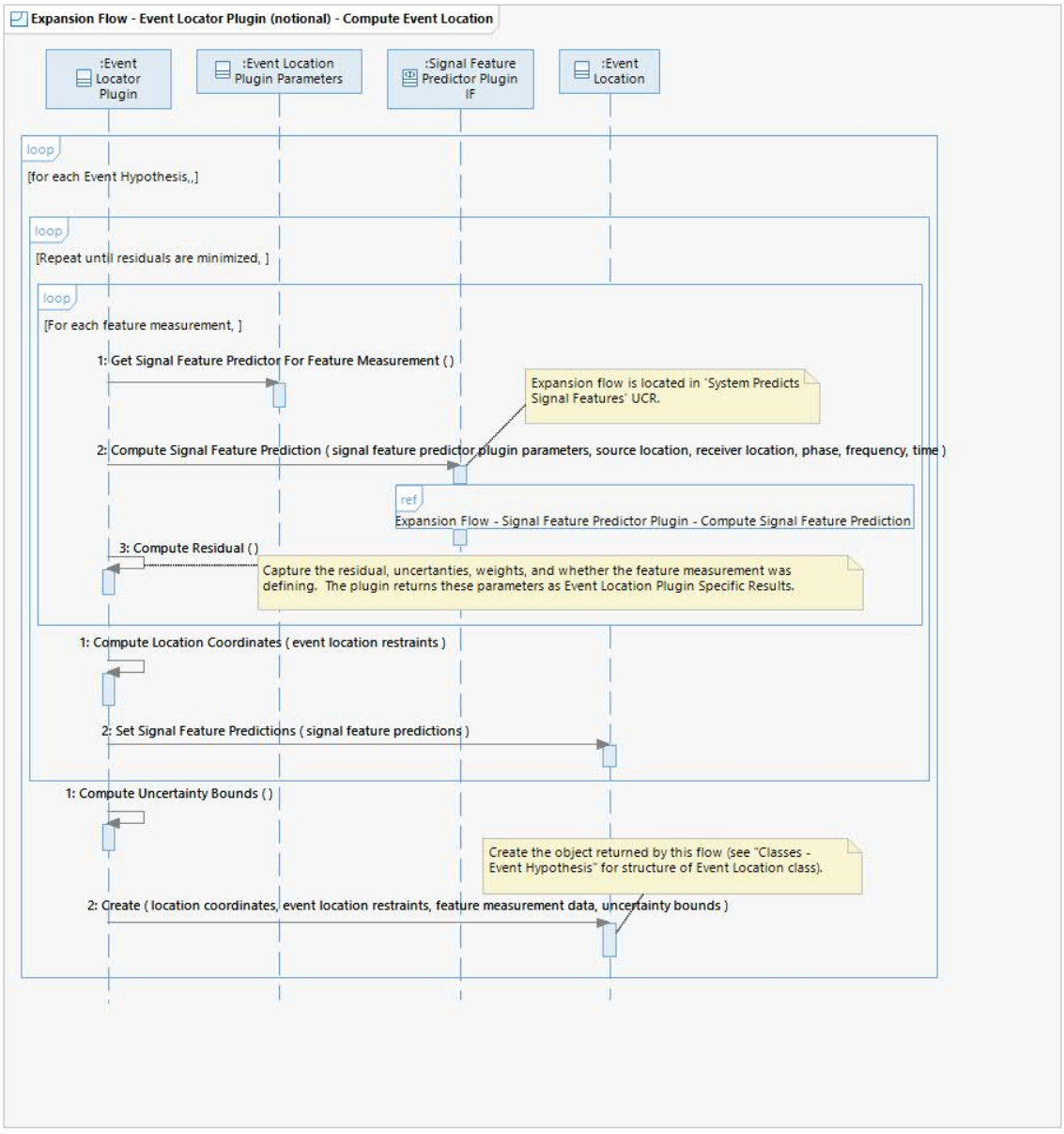
Operation: Event Locator Plugin IF::Compute Event Location()

The interface method for an Event Location Plugin to compute an event location. The plugin may compute the location using teleseismic and regional seismic signal detections.

Operation: OSD::Store Event Hypothesis()

Store the given Event Hypothesis with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Expansion Flow - Event Locator Plugin (notional) - Compute Event Location



This flow notionally shows how a particular Event Locator plugin might compute a location for a batch of event hypotheses. The flow shown here may not apply to all Event Locator Plugins. In this example, the Event Locator Plugin iteratively computes residuals between observed vs. predicted feature measurements and updates the location coordinates until the residuals no longer improve. To compute predicted feature measurements, the Event Locator uses a Signal Feature Predictor plugin. The specific predictor used may vary for each prediction based on parameters

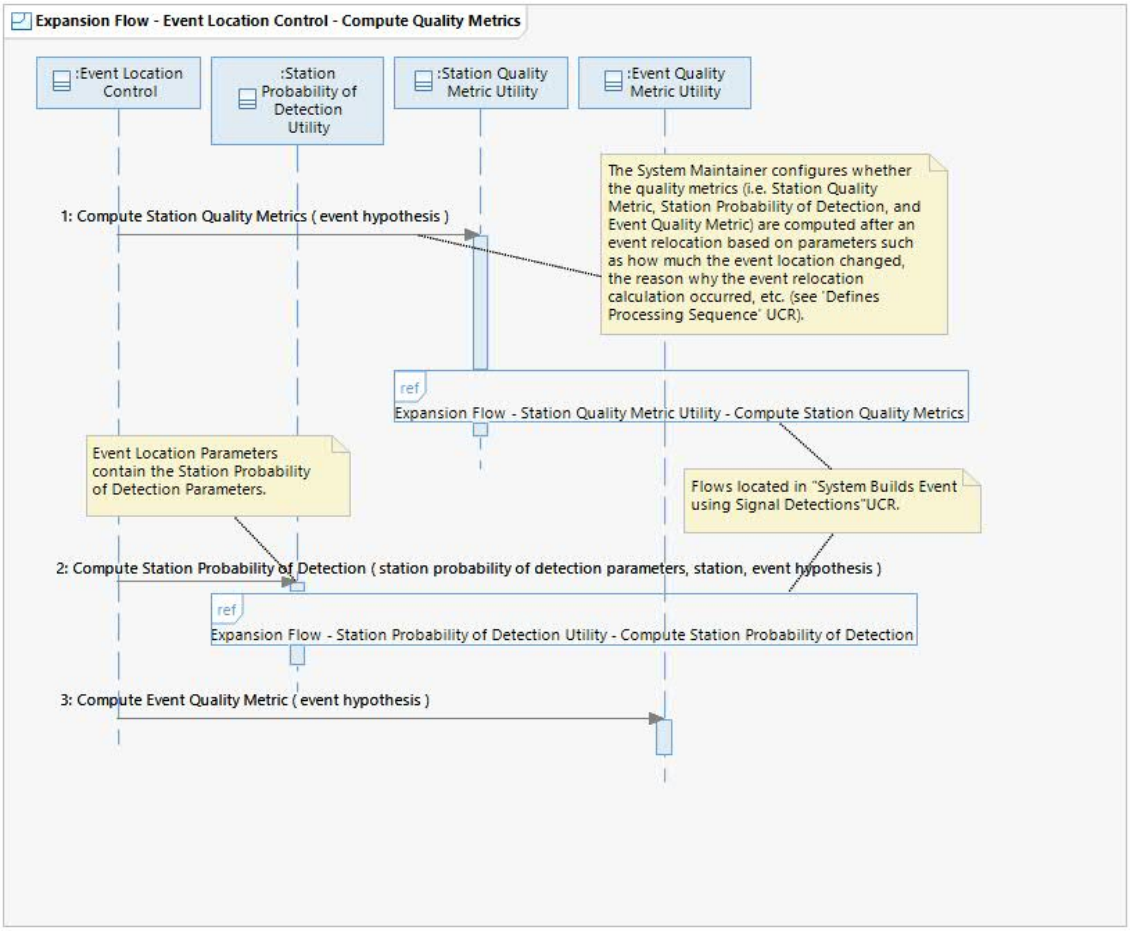
specified in the Event Location Plugin Parameters class. The Event Locator Plugin can set the defining/non-defining state for each signal detection measurement while computing the event location.

As a possible variation of this flow for performing Master Event Location, the flow might look essentially the same except that instead of using a Signal Feature Predictor to get the predicted feature measurements the Event Locator might use the feature measurements associated with a designated "master event" (obtained from the Event Location Plugin Parameters class). A variation of this flow to simultaneously relocate multiple event hypotheses relative to each other might use feature measurement differences between all of the input event hypotheses to simultaneously find locations for each event hypothesis. The algorithm might iterate this process until the entire set of locations for the event hypotheses no longer improves (i.e. the locations are globally optimal).

Operation Descriptions

Operation: Event Locator Plugin::Compute Residual()
Compute the difference between the signal feature measurement and the signal feature prediction, as well as the residual uncertainty.

Expansion Flow - Event Location Control - Compute Quality Metrics



This flow shows how the Event Location Control class recomputes quality metrics based on the Event Hypothesis' updated event locations.

Operation Descriptions

None

State Machine Diagrams

None

SSD Mappings

General:

S-1563: [*Threshold*] The System shall locate event hypotheses found using waveform correlation processing using the same location algorithms as events found using other types of event processing.

S-1572: [*Threshold*] The System shall compute the station quality metric for all events.

S-1576: [*Threshold*] The System shall store the station quality metrics for all stations for each event hypothesis.

S-1580: [*Threshold*] The System shall recompute the event hypothesis quality metric for an event hypothesis when any of the event hypothesis quality statistics used to calculate the event hypothesis quality metric are updated.

S-1588: [*Threshold*] The System shall store the event quality metric for each event hypothesis.

S-1592: [*Threshold*] The System shall compute event hypothesis relocations using seismic, hydroacoustic, and infrasound signal detection feature measurements.

S-1593: [*Threshold*] The System shall compute event hypothesis relocations using the signal detection feature measurements from a single station.

S-1594: [*Threshold*] The System shall compute event hypothesis relocations using the signal detection feature measurements from multiple stations.

S-1595: [*Threshold*] The System shall compute event hypothesis relocation uncertainty bounds.

S-1596: [*Threshold*] The System shall compute the uncertainty coverage ellipse for each event hypothesis relocation.

S-1600: [*Threshold*] The System shall set the defining/non-defining state for signal detection measurements during event hypothesis relocation processing.

S-1619: [*Threshold*] The System shall store the confidence level of each computed event hypothesis location uncertainty bound.

S-1620: [*Threshold*] The System shall store the type (i.e., confidence, coverage, or k-weighted with the associated weights) of each location uncertainty bound.

S-1623: [*Threshold*] The System shall store the sum squared weighted residual for each event hypothesis location.

S-1624: [*Threshold*] The System shall store the defining/non-defining state for each signal detection measurement associated to a stored event hypothesis.

S-1631: [*Threshold*] The System shall compute event hypothesis relocations using teleseismic and regional seismic signal detections.

S-1640: [*Threshold*] The System shall perform master event relocation using travel time differences.

S-1653: [*Threshold*] The System shall compute new event hypothesis magnitude estimates when

a new event hypothesis location is computed.

S-2036: [*Threshold*] The System shall use configured default defining/non-defining state settings and precedence rules to determine the initial defining/non-defining state for each parameter.

S-2043: [*Threshold*] The System shall store automatic and interactive processing results.

S-2223: [*Threshold*] The System shall store all data and derived processing results to persistent storage as soon as the data and/or derived processing results are available.

S-6290: [*Threshold*] The System shall perform multiple event relocation using differences in signal detection feature measurements.

Notes

General:
1. Multiple Event Relocation (such as Joint Hypocenter Determination) is considered to be a specialized research activity and is thus invoked by the Researcher in 'Performs Multiple Event Location' UCR instead of by the Analyst from 'Refines Event Location' UCR.

2. Computation of the event quality metric after an event is relocated is configurable (see 'Defines Processing Sequence' UCR). Configuration could include selection of a minimum change in location that would result in recomputing the event quality metric or selection of when to recompute the event quality metric based on the cause for the event relocation.

IDC Use Case Realization Report

UCR-03.02 Refines Event

Use Case Description

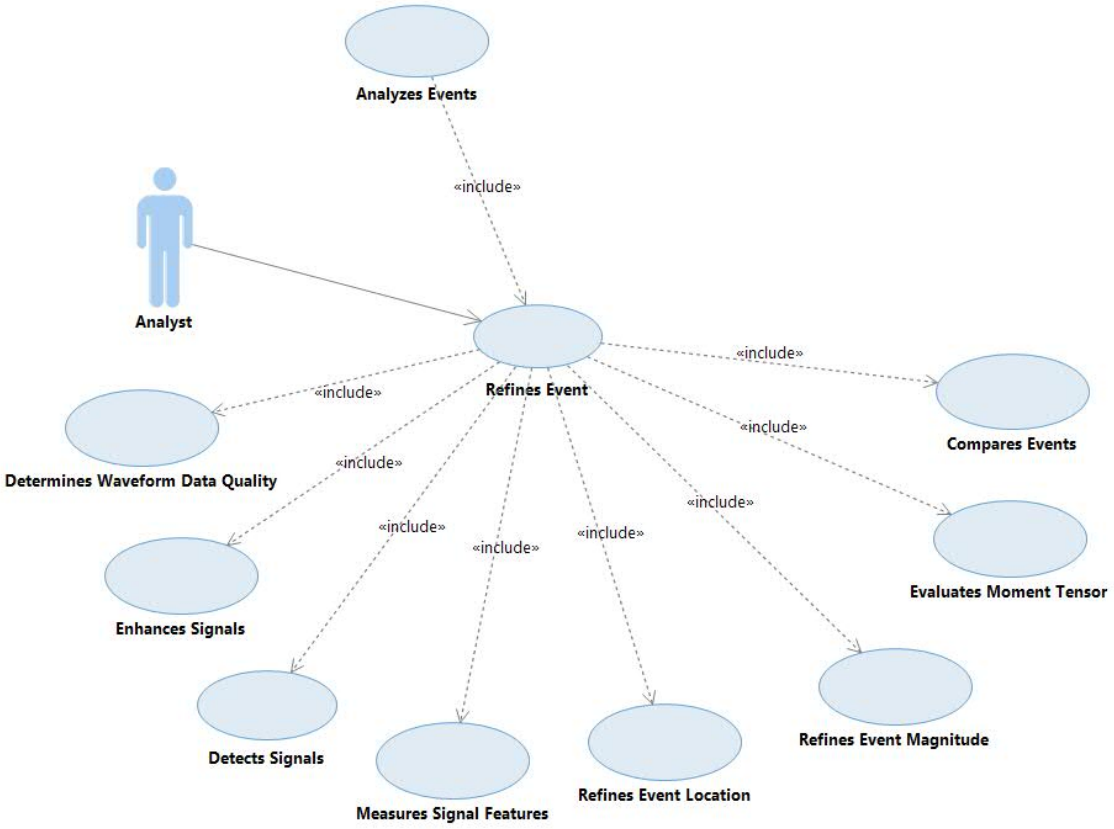
This architecturally significant use case describes how the Analyst refines an event hypothesis. The Analyst checks waveform quality (see 'Determines Waveform Data Quality' UC). For waveforms of sufficient quality, the Analyst enhances signals and suppresses noise on waveforms for relevant stations (see 'Enhances Signals' UC), adds and associates missing detections, and modifies or unassociates detections already associated with the event hypothesis (see 'Detects Signals' UC). The Analyst rejects event hypotheses that are invalid. For valid event hypotheses, the Analyst measures signal features associated with the detections (see 'Measures Signal Features' UC) and evaluates the moment tensor ('Evaluates Moment Tensor' UC). The Analyst uses these signal features to refine the location (see 'Refines Event Location' UC) and magnitude (see 'Refines Event Magnitude' UC) of the event hypothesis. The Analyst compares events to determine how similar events were constructed (see 'Compares Events' UC). The Analyst repeats these steps until satisfied with the results.

This use case is architecturally significant because it encompasses interaction between a large number of capabilities available to Analysis, including synchronized interaction among those capabilities, the Analyst ability to initiate automatic processing algorithms with overridden System parameters, and capture and display of provenance for Analyst actions.

Architecture Description

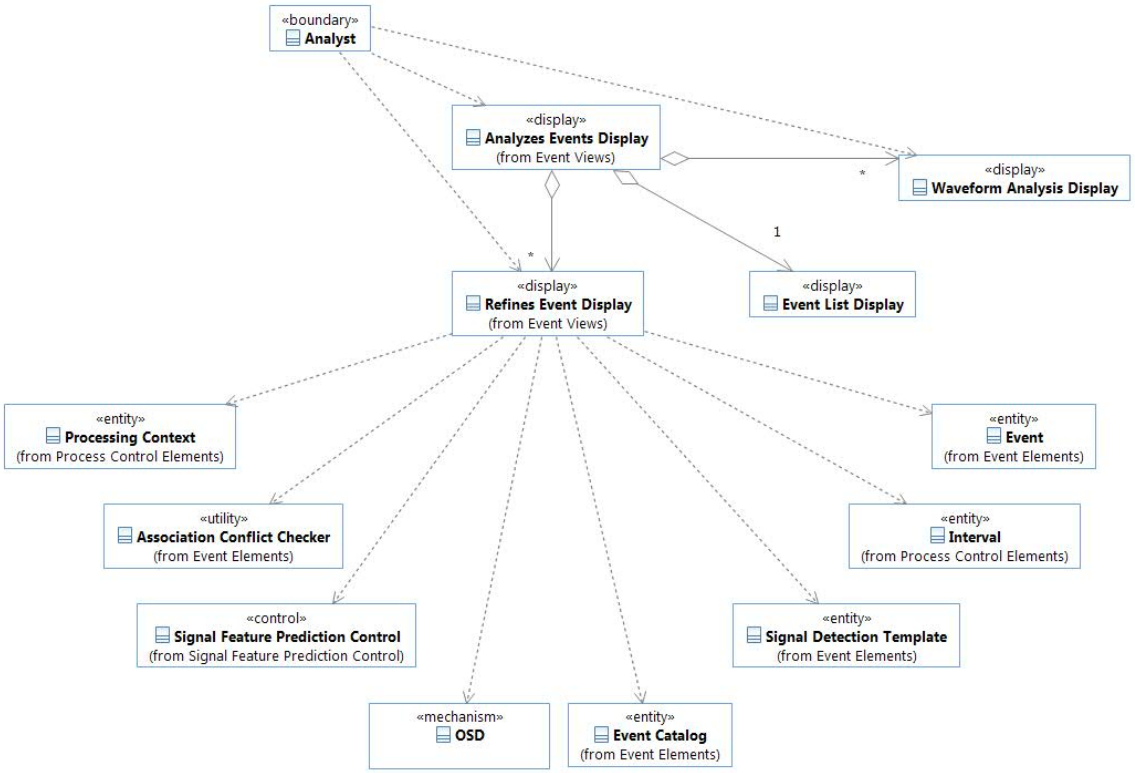
The Analyst refines an Event by selecting an Event on the Event List Display to open the Refines Event Display for the Event. The Refines Event Display retrieves the current preferred Event Hypothesis for the Event to use as a starting point, creates a local copy of it for the current processing stage, and provides the Analyst with the ability to refine it (depicted in included use cases). As the Analyst refines the Event the Event Hypothesis is updated and stored transiently in a private context via the OSD mechanism to make it available to the Processing Sequence Control mechanism for further automatic processing (the Processing Sequence Control mechanism is described in "System Detects Event" UCR). To save their changes, the Analyst selects to save the Event Hypothesis, which the display handles by storing the Event Hypothesis in a global context to persistent storage via the OSD.

Use Case Diagram



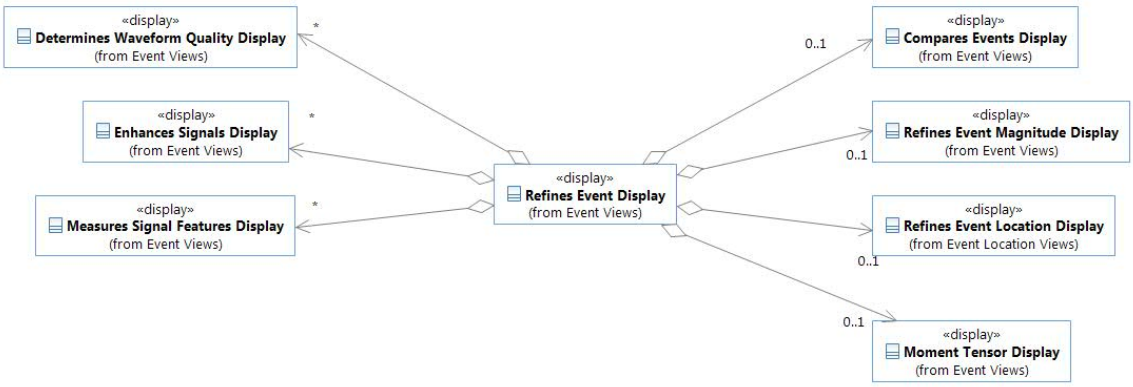
Class Diagrams

Classes - Refines Event Display



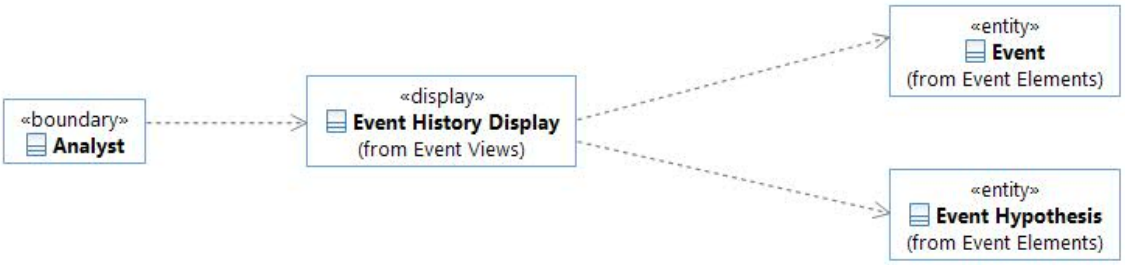
This diagram shows the Refines Event Display and related classes pertaining to this realization. The Analyst opens the Refines Event Display from the Analyzes Events Display. This display subscribes for the Event being refined in order to warn the Analyst if the Event is under active review by another Analyst in the same interval. The display subscribes for Intervals in order to warn the Analyst if the interval containing the Event is under active review by another Analyst (see "Scans Waveforms and Unassociated Detections" UCR). The Refines Event Display uses the Association Conflict Checker class to check for association conflicts with other Events whenever the current Event is modified or another Event in the interval is saved. The display also provides the Analyst with the ability to create Signal Detection Templates, which it stores in the OSD. Refines Event Display stores the refined Event in the OSD.

Classes - Refines Event Display - Sub-displays



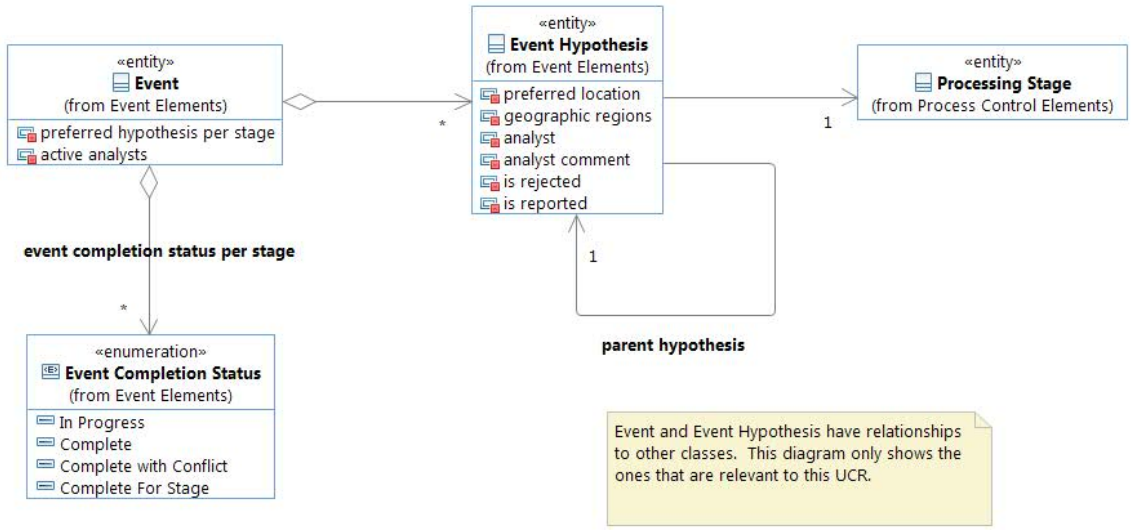
This class shows sub-displays of the Refines Event Display. The Refines Event Display creates and manages these sub-displays based on Analyst actions. Analyst interactions with these sub-displays are described in the corresponding UCRs; however, in general, the OSD mechanism is used to synchronize information between the displays. When the Analyst first opens the Event, the Refines Event Display creates a new Event Hypothesis and stores it in the OSD (in a private context, not visible to other Analysts). The Refines Event Display then subscribes for changes to this Event Hypothesis via the OSD. As the Analyst interacts with the various sub-displays, those Analyst actions may trigger processing on the privately stored Event Hypothesis; however, the sub-displays do not have knowledge of which processing steps will be performed since the processing sequences to be executed in response to Analyst actions are configurable (see "Defines Processing Sequence" UCR) and known only by the Processing Sequence Control mechanism. The Refines Event Display is informed of any processing performed on the Event Hypothesis via OSD callbacks.

Classes - Event History Display



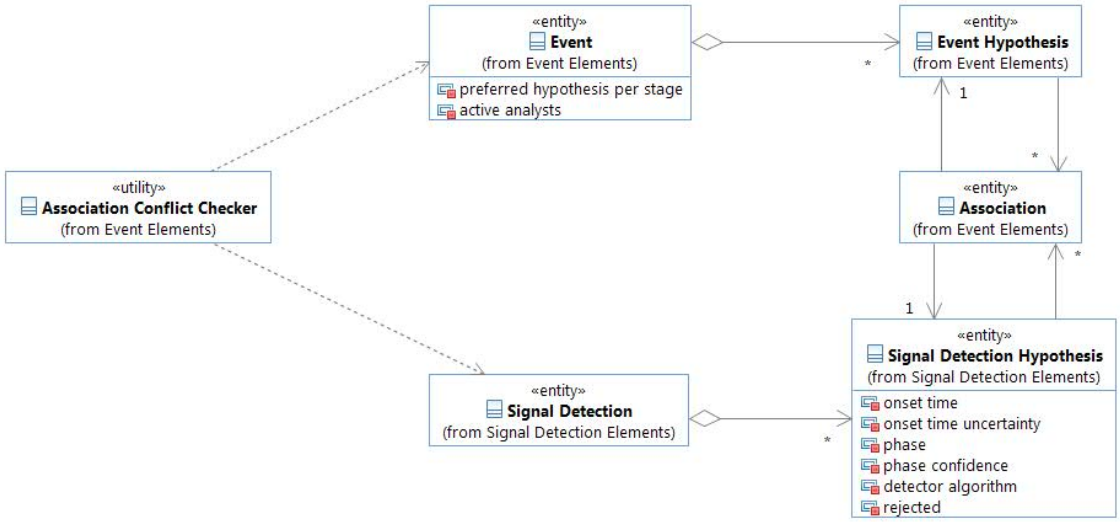
This diagram shows the Event History Display and related classes. The Analyst may use the Event History Display to select to begin their refinement with any hypothesis in an Event's history.

Classes - Event



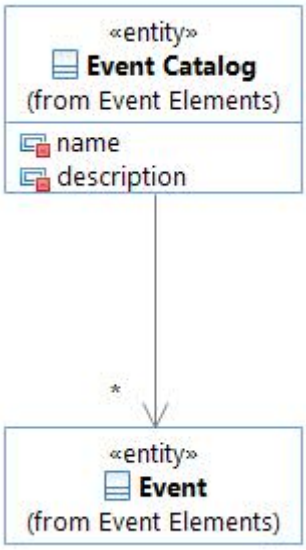
This diagram shows details of the Event and Event Hypothesis classes relevant to this UCR. Refinement of an Event results in a new Event Hypothesis. The Analyst potentially creates multiple Event Hypotheses for a given Event during a single processing stage, and designates one of them as the "preferred" Event Hypothesis for the Event for that stage (each stage can have a different preferred Event Hypothesis for the Event). Each Event also has an Event Completion Status, which reflects the Analyst's determination of the level of completeness of the Event within the stage. The Analyst specifies an Event Completion Status of "In Progress" or "Complete" when saving the Event. The transition to "Complete For Stage" is covered in a separate UCR (see "Marks Processing Stage Complete" UCR).

Classes - Association Conflict Checker



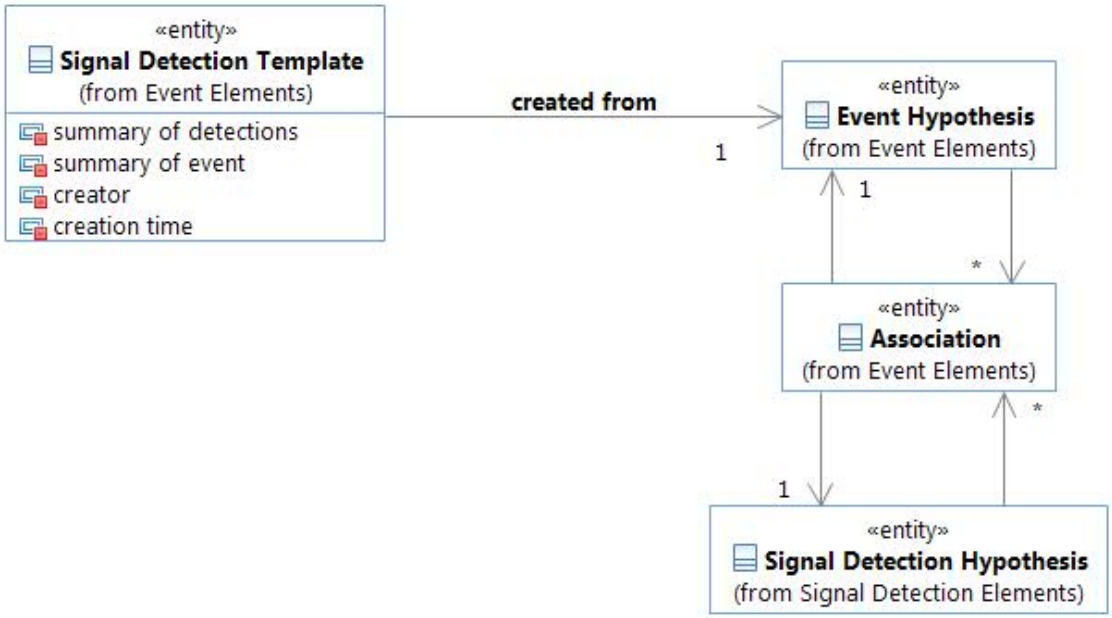
This diagram shows details of the Association Conflict Checker class. The class retrieves Events, Associations, and Signal Detections from the OSD and checks for the case where more than one Event has a preferred Event Hypothesis for the stage that has a related Association to a Signal Detection Hypothesis for the same Signal Detection.

Classes - Event Catalog



This diagram shows details for the Event Catalog class. The Analyst uses the Refines Event display to create Event Catalogs and to update Events contained in the catalogs.

Classes - Signal Detection Template



This diagram shows details for the Signal Detection Template class. The Analyst may create a Signal Detection Template via the Refines Event Display based on the Signal Detection Hypotheses associated to the current Event Hypothesis.

Class Descriptions

<<boundary>> Analyst
Represents the Analyst actor.

<<control>> Signal Feature Prediction Control
Responsible for controlling the signal feature prediction computations. Retrieves necessary data, invokes the appropriate Signal Feature Predictor Plugin to compute the desired signal feature prediction, and stores the result.

<<display>> Analyzes Events Display
Display that provides the Analyst with the ability to analyze data within a specified time interval in order to find or refine Events.

<<display>> Compares Events Display
Display that provides the Analyst with the ability to compare Events.

<<display>> Event List Display
The Event List Display provides a list of Events related to the current context of an Analyst's analysis activities. Primarily the Event List shows the Events within the selected Interval times. Alternately, the Event List shows the Events returned as result of a search or the list of Events selected by the Analyst. The Event List provides the interface to select an Event for further analysis.

<<display>> Refines Event Display
Display that provides the Analyst with the ability to refine an Event. Each saved refinement of the Event results in a new Event Hypothesis.

<<display>> Refines Event Location Display
Provides the Analyst with ability to enter Event location parameters and initiate computation of a location for an Event Hypothesis.

<<display>> Waveform Analysis Display
Displays a set of waveforms and Detection Feature Maps and provides the Analyst with the ability to interact with them (e.g. create/modify/reject Signal Detections, associate/unassociate detections and Events).

<<entity>> Association
Represents an association between a Signal Detection Hypothesis and an Event Hypothesis.

<<entity>> Event
Represents information about an Event. Keeps track of all the Event Hypotheses for the Event, which Event Hypothesis is the preferred one for each processing stage, the active analysts for the Event (i.e. whether the Event is under "active review"), whether the Event is "complete" for each processing stage, and other Event-related information.

<<entity>> Event Catalog
A named catalog of Events. Event Catalogs group Events with common features (e.g. ground truth catalogs, special event catalogs, large event catalogs, etc.) Analysts create Event Catalogs, update which Events are listed in the catalogs, and use the catalogs to help search for Events with specific features. The catalog logically lists Events rather than specific Event Hypotheses.

<<entity>> Event Hypothesis
Represents geophysical information about an Event as determined by an Analyst or through pipeline processing. There can be multiple Event Hypotheses for the same Event (e.g. different associated Signal Detection Hypotheses, different location solutions).

<<entity>> Interval
Class for tracking the status of interactive or automatic processing on a specific timeframe of data. Specialized intervals exist for Processing Stage, Processing Activity, and Processing Sequence.

<<entity>> Processing Context
Represents the context in which data is being stored and/or processed. This includes the Processing Stage (either automatic or interactive) and Interval performing the processing session (e.g. processed by Analyst vs. processed by System). For Analyst processing, may identify the Analyst work session. For System processing, may identify the Processing Sequence and/or Processing Step being executed (including a way to identify a particular Processing Sequence and Processing Step among the many possible instantiations), the visibility for the results (private vs. global), and the lifespan of the data (transient vs. persistent). This information is needed by the Processing Sequence Control to manage the execution of Processing Sequences, which may execute in the context of an Analyst refining an Event or in the context of the system initiating automatic processing. It is also needed by the Object Storage and Distribution (OSD) mechanism to determine how to store and distribute the data.

<<entity>> Processing Stage
Represents a named stage of data processing, which may be part of the System Maintainer-defined workflow or an Analyst-defined stage outside the workflow. All Processing Results are associated to a Processing Stage.

<<entity>> Signal Detection
Represents information about a Signal Detection and keeps track of all the Signal Detection Hypotheses for the Signal Detection. Represents information about a Signal Detection and keeps track of all the Signal Detection Hypotheses for the Signal Detection. For an unassociated Signal Detection the preferred Signal Detection Hypothesis is the most recently created Signal Detection Hypothesis. For an associated Signal Detection the preferred Signal Detection Hypothesis is the one associated to a preferred Event Hypothesis.

<<entity>> Signal Detection Hypothesis
Represents geophysical information about a Signal Detection as determined by an Analyst or through pipeline processing. There can be multiple Signal Detection Hypotheses for the same Signal Detection (e.g. different onset times, different phase labels).

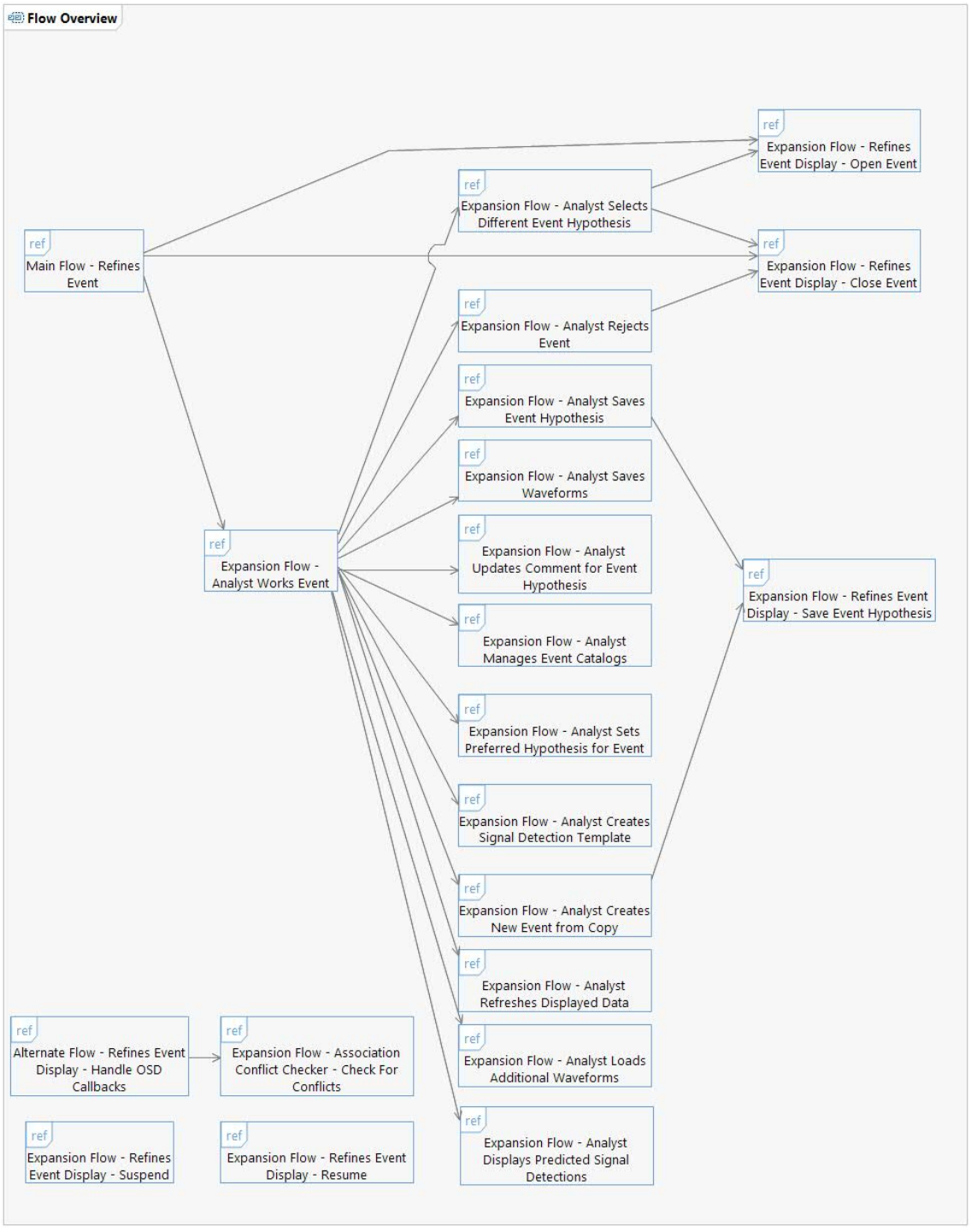
<<entity>> Signal Detection Template
A template that represents the pattern of Signal Detections for an Event (i.e. channels detected, relative positions for each detection, phases, etc.). An Analyst may apply the template to quickly build new Events that match the pattern of detections. Signal Detection Associator Plugins (see "System Builds Events using Signal Detections" UCR) may also use Signal Detection Templates to build new Event Hypotheses and associate additional detections to Event Hypotheses matching the template. Also includes summary information about the original Event from which the template was created (e.g. Event location, magnitude, etc.), as an aid to the Analyst in finding and applying a relevant template.

<<mechanism>> OSD
Represents the Object Storage and Distribution mechanism for storing and distributing data objects internally within the system.

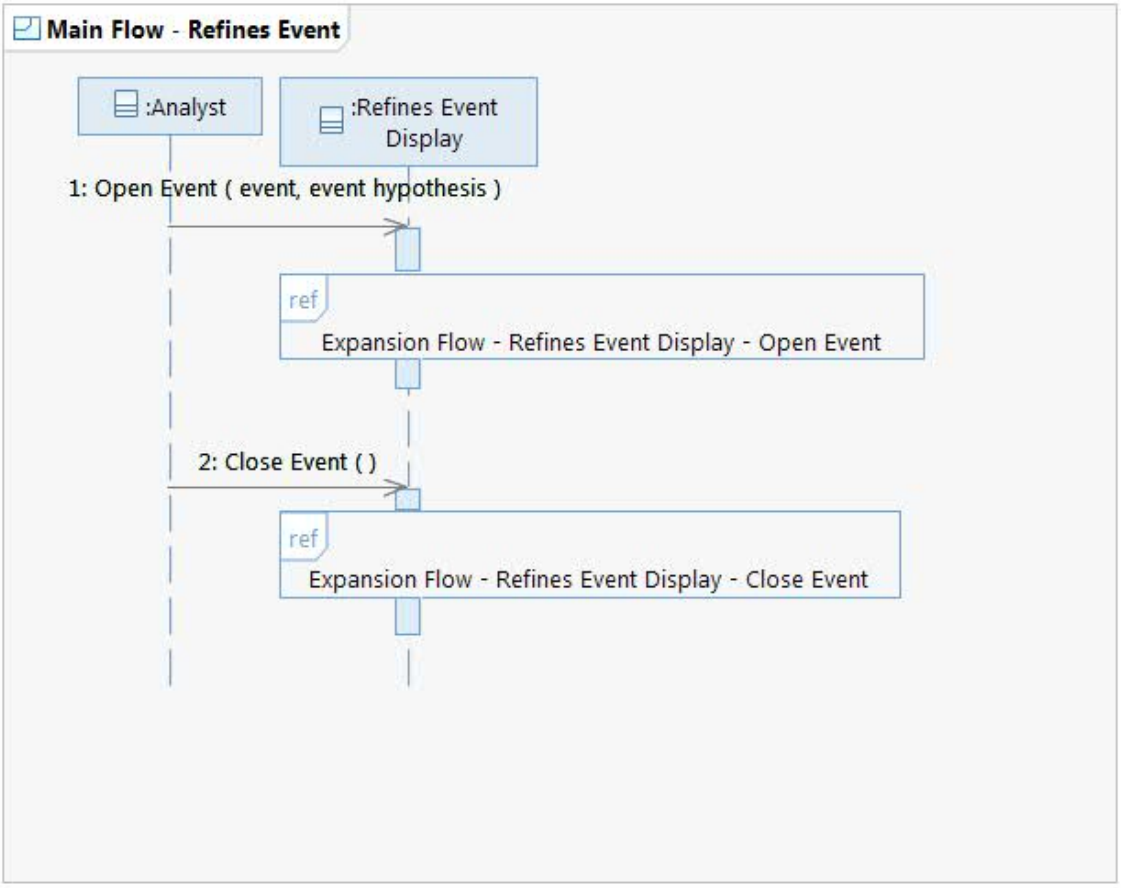
<<utility>> Association Conflict Checker
Utility class for checking that preferred Event Hypotheses within a processing stage do not share any Signal Detections.

Sequence Diagrams

Flow Overview



Main Flow - Refines Event

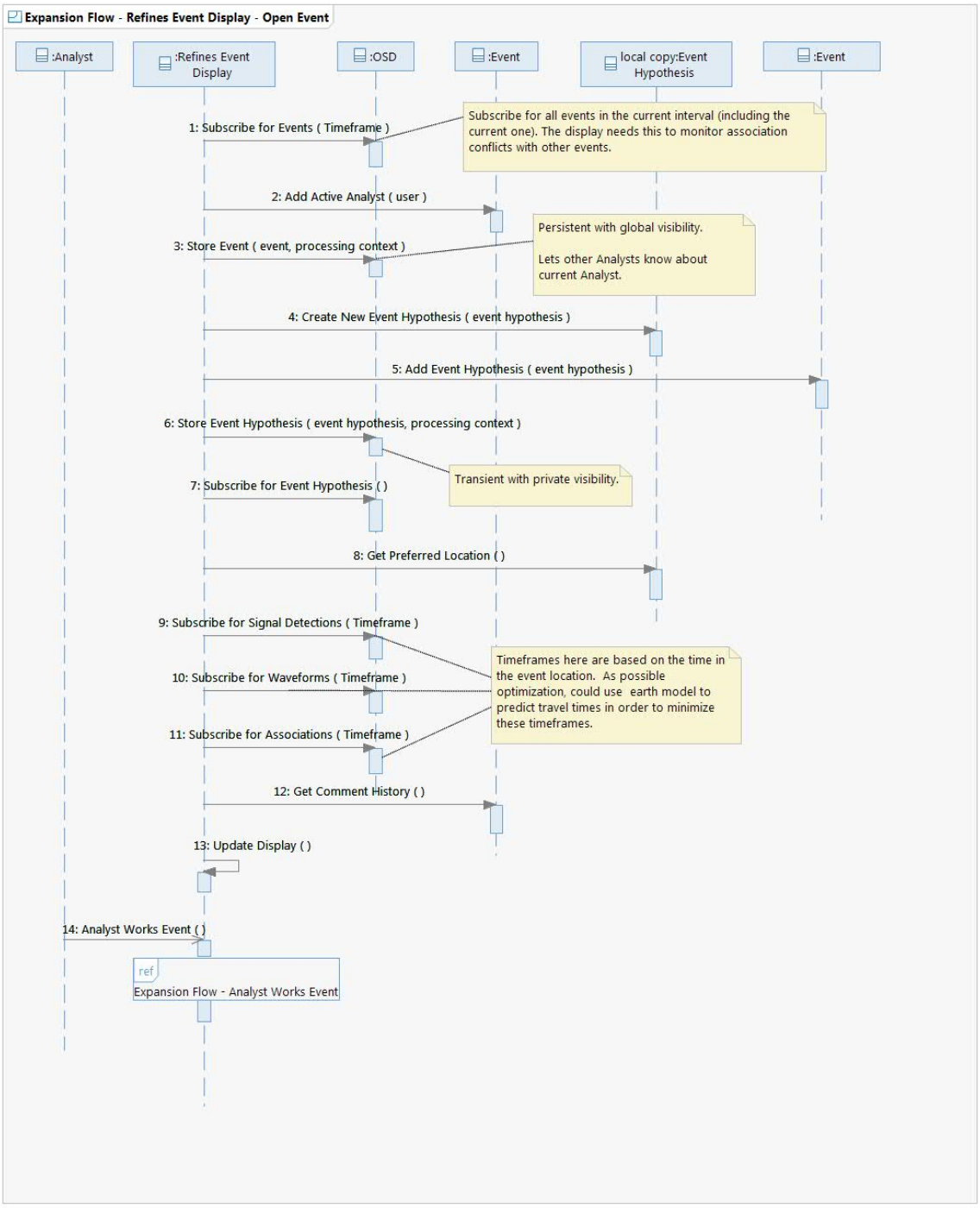


This flow shows the main flow for refining an Event. The Refines Event Display is typically opened with an Event, in which case the display automatically determines which Event Hypothesis to use as a starting point (the Analyst can select a different Hypothesis as a starting point - see "Expansion Flow - Analyst Selects Different Event Hypothesis"). The Analyst may also open the Refines Event Display with a specific Event Hypothesis to refine. In this case the display uses that Hypothesis as the starting point.

Operation Descriptions

Operation: Refines Event Display::Open Event()
Open the given Event for refinement in the current processing stage, using the given Event Hypothesis as a starting point.

Expansion Flow - Refines Event Display - Open Event



This flow shows how the Refines Event Display opens an Event for refinement. The Event Hypothesis to use as a starting point is an input to this flow. The system keeps track of all the analysts that are working an Event ("active analysts") and warns if the Event is under active review by another analyst or overlaps an interval that is under active review by another analyst (see "Alternate Flow - Refines Event Display - Handle OSD Callbacks"). The display creates a new Event Hypothesis instance based on the passed-in Event Hypothesis. The display subscribes

for detections and waveforms around the Event in order to display them. The comment history is also displayed. After the display is open the Analyst can perform several analysis functions as described in Expansion Flow - Refines Event Display - Analyst Works Event.

Operation Descriptions

Operation: OSD::Subscribe for Events()
Subscribe for changes to Event objects within the given timeframe. Callbacks are invoked on subscribers any time an Event within the timeframe is added or modified.

Operation: OSD::Subscribe for Signal Detections()
Subscribe for updates regarding Signal Detection creations, modifications, and associations occurring within the specified timeframe. This includes updates for new or modified unassociated Signal Detections.

Operation: OSD::Subscribe for Waveforms()
Subscribe for updates regarding raw and derived waveforms occurring within a specified timeframe. This includes information about what waveforms have been acquired by the System as well as what derived waveforms have been formed, but does not include the actual waveform data.

Operation: Event::Add Active Analyst()
Add the given Analyst to the set of active Analysts for the Event. If the Event has active Analysts it is said to be under "active review".

Operation: OSD::Store Event()
Store the given Event with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Operation: Event Hypothesis::Create New Event Hypothesis()
Create a copy of the given Event Hypothesis. The copy has all of the same information as the original (e.g. same detections, location, etc.), with the following exceptions:

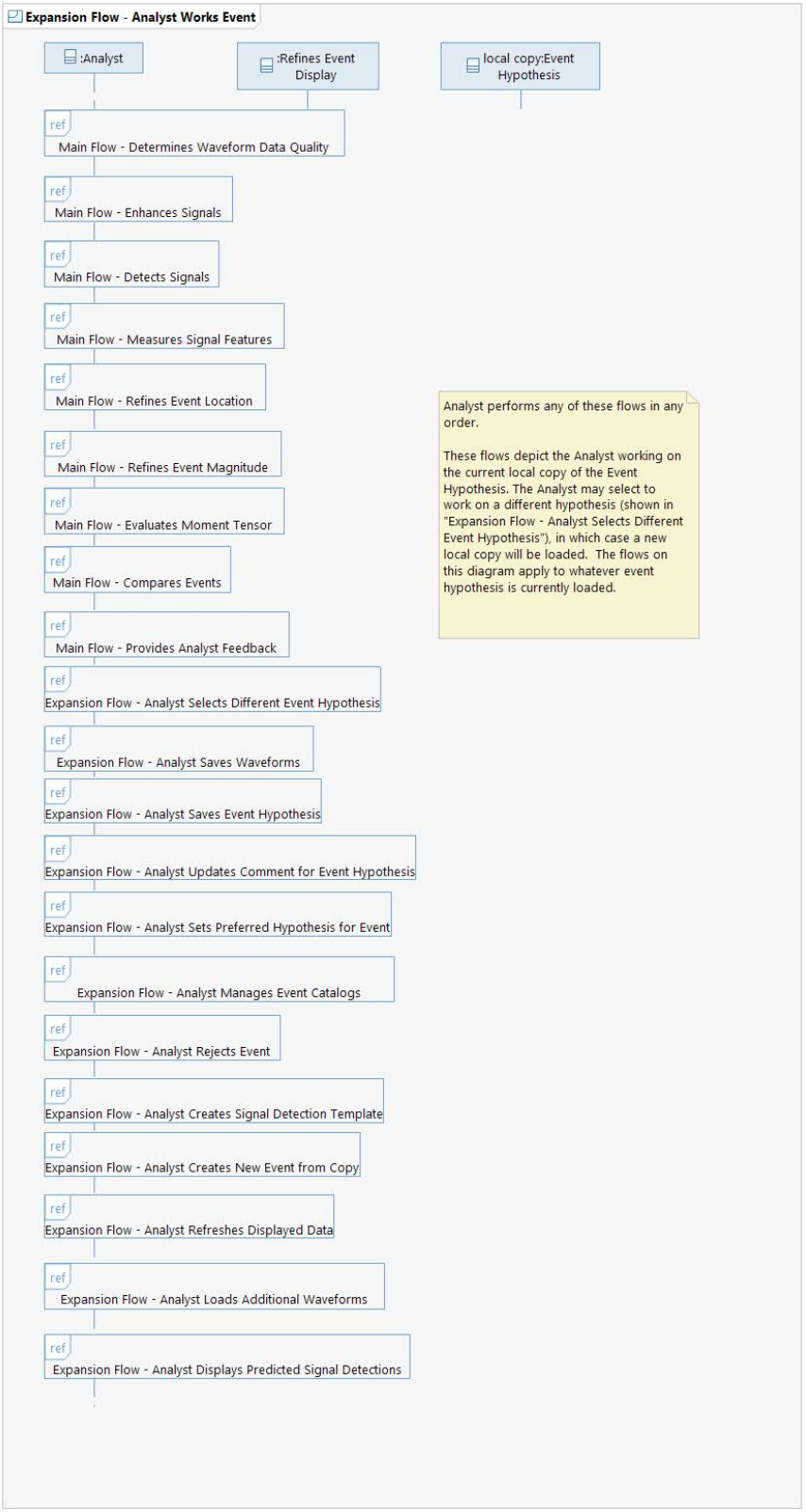
- The copy points to the original as its parent
- The copy starts out with an empty Analyst comment

Operation: OSD::Store Event Hypothesis()
Store the given Event Hypothesis with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Operation: Refines Event Display::Update Display()
Update the display of the Event Hypothesis that is currently being refined to reflect any changes that may have occurred. Indicate items that are out-of-date or inconsistent (e.g. beam may be out-of-date after refining Event location).

Operation: Event::Get Comment History()
Return all Analyst-entered comments associated with the Event.

Expansion Flow - Analyst Works Event

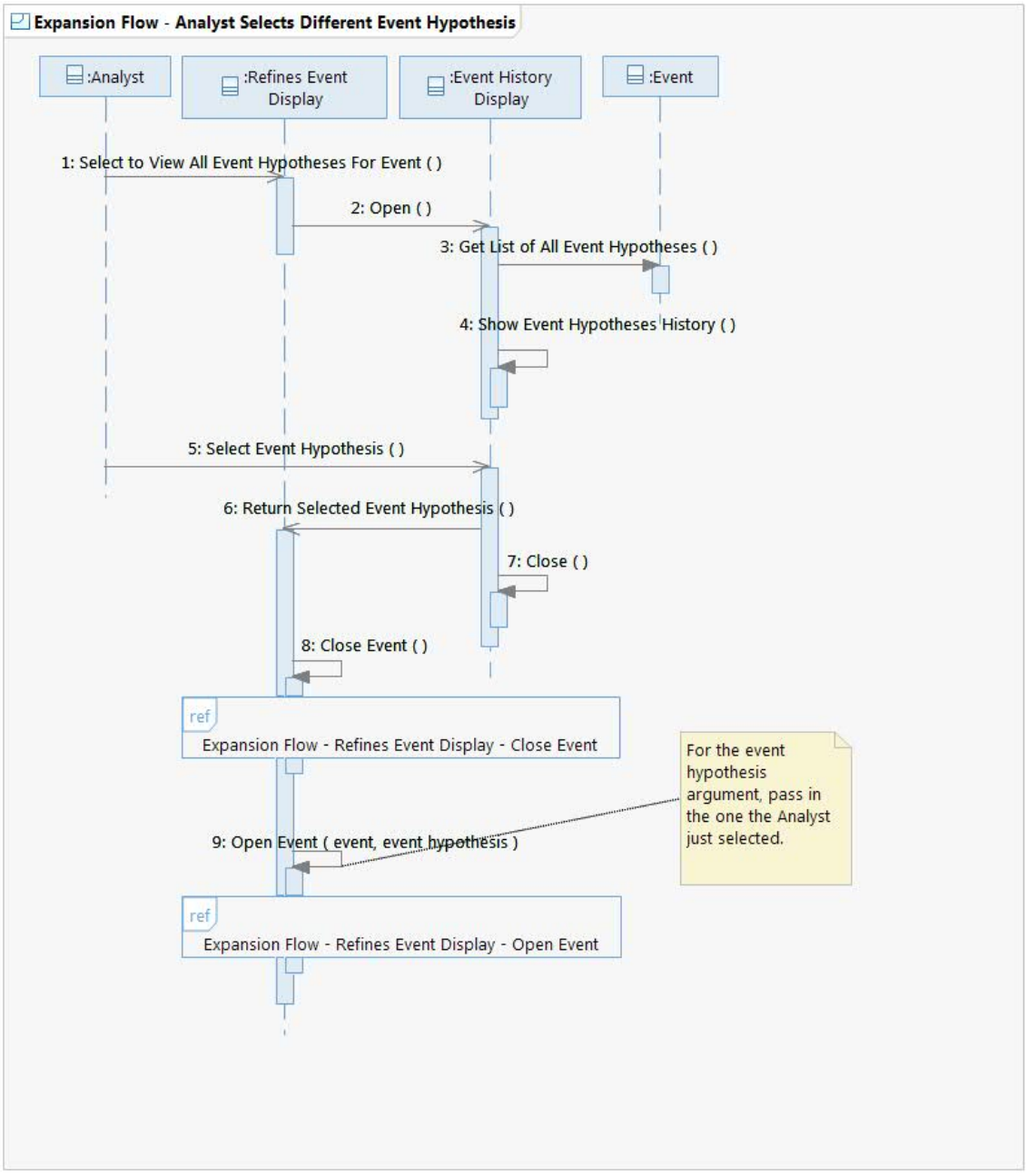


This flow shows the actions an Analyst can perform as part of refining an Event.

Operation Descriptions

None

Expansion Flow - Analyst Selects Different Event Hypothesis



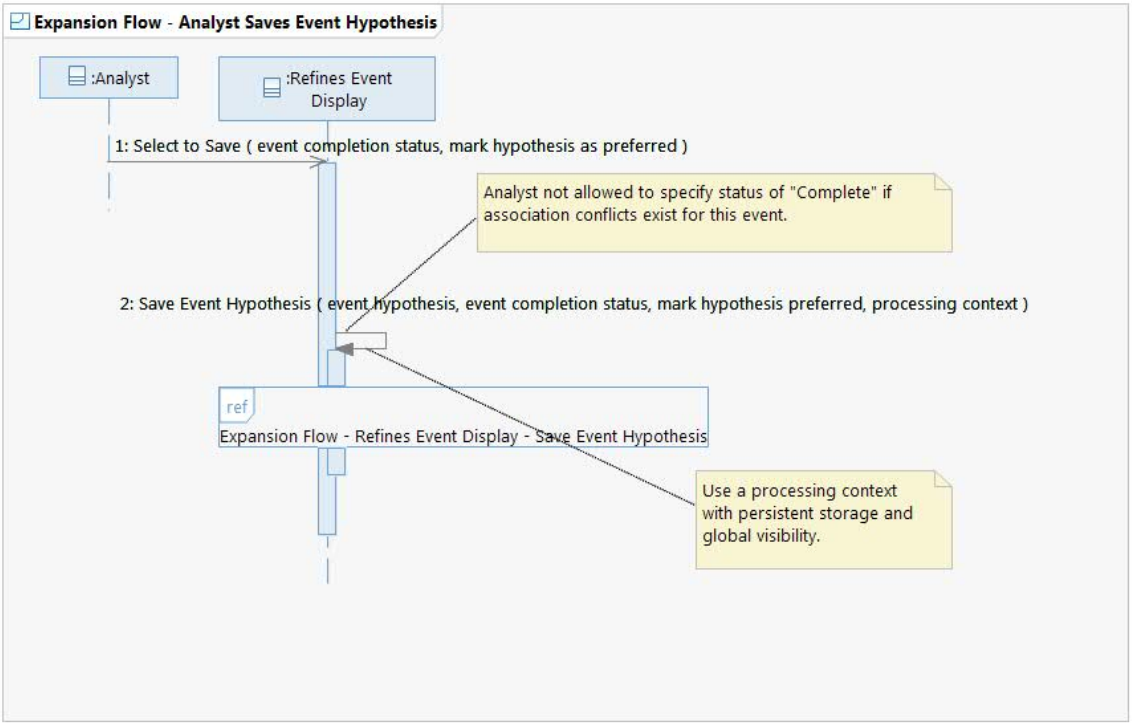
This flow shows the Analyst selecting to refine a different Event Hypothesis for the Event (other than the current one).

Operation Descriptions

Operation: Refines Event Display::Open Event()
Open the given Event for refinement in the current processing stage, using the given Event Hypothesis as a starting point.

Operation: Event::Get List of All Event Hypotheses()
Return a list of all the Event Hypothesis for the given Event, including summary information such as the processing stage and which Event Hypotheses have been designated as preferred.

Expansion Flow - Analyst Saves Event Hypothesis



This flow shows how the Analyst saves the event hypothesis they are refining to persistent storage and makes it visible to other Analysts. Once the event hypothesis is saved to persistent storage it can never be modified again (but the Analyst can create a new event hypothesis to further refine the event). When saving the event, the Analyst specifies the completion status for the event (see Event Completion Status class on diagram "Classes - Event") and whether to mark the hypothesis as the preferred one for the event. The Analyst may mark any Event Hypothesis as preferred for an Event while refining the Event (see "Expansion Flow – Analyst Sets Preferred Hypothesis for Event"). This marking supersedes any previous preferred marking for the same Event and Processing Stage. Association Conflict Checker continuously checks for and keeps track of event conflicts as the Analyst associates Signal Detection Hypotheses to Event Hypotheses (see "Expansion Flow – Association Conflict Checker – Check for Conflicts"). The Analyst may only mark an Event complete when it's preferred Event Hypothesis does not conflict with any other preferred Event Hypothesis from the same Processing Stage. If a conflict occurs after an Event is saved as Complete then Analyzes Events Display sets the Event back to Complete with Conflict (see 'Analyzes Events' UCR "Alternate Flow – Refines Event Display - Handle OSD Callbacks").

Operation Descriptions

None

Expansion Flow - Refines Event Display - Save Event Hypothesis

This flow shows how Refines Event Display saves an Event Hypothesis. This flow is provided

parameters telling it which hypothesis to save, the event completion status to use, whether the event hypothesis is the preferred hypothesis for the event, and the processing context to use when storing the hypothesis in the OSD. To support tracking Event History, Refines Event Display adds the Event Hypothesis to the parent Event. Refines Event Display also sets the Event's completion status and, if necessary, marks the hypothesis as preferred for the event. The display then stores the Event, Event Hypothesis, Association, Signal Detection, Signal Detection Hypothesis, and Waveform objects to the OSD.

Operation Descriptions

Operation: OSD::Store Event Hypothesis()
Store the given Event Hypothesis with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

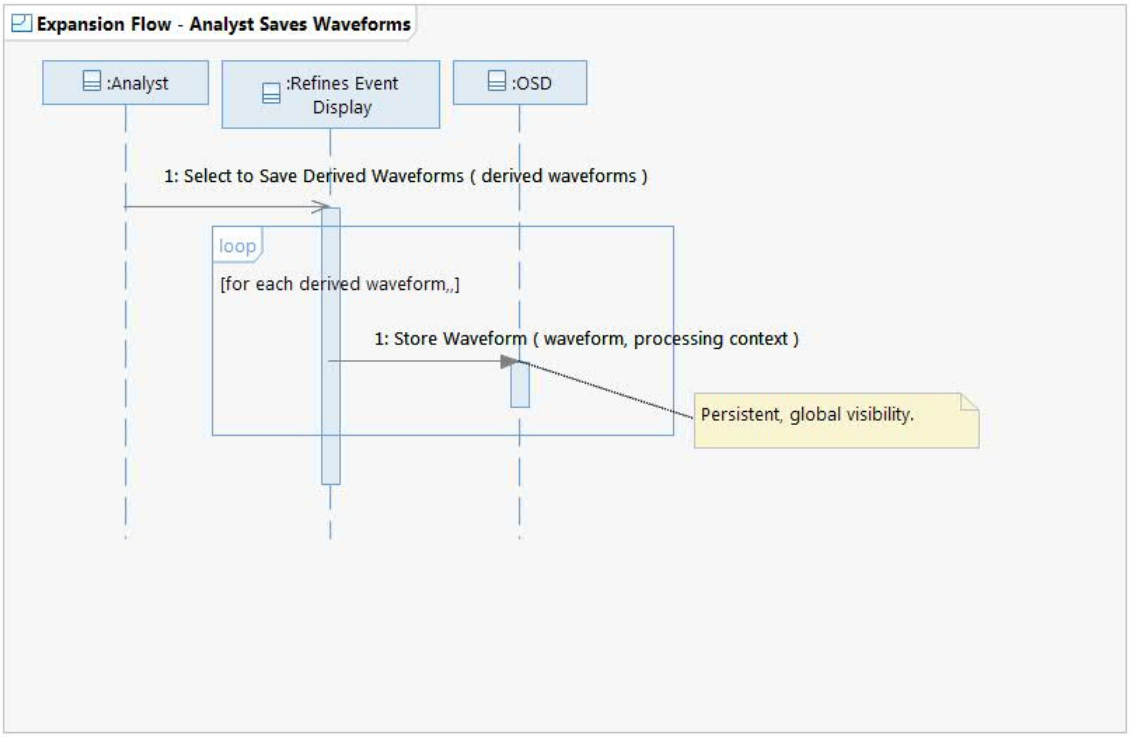
Operation: OSD::Store Event()
Store the given Event with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Operation: Event::Set Completion Status()
Set the Event Completion Status of an Event in the given processing stage to the given value.

Operation: OSD::Store Waveform()
Store the given Waveform with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Operation: OSD::Store Signal Detection()
Store the given Signal Detection with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Expansion Flow - Analyst Saves Waveforms

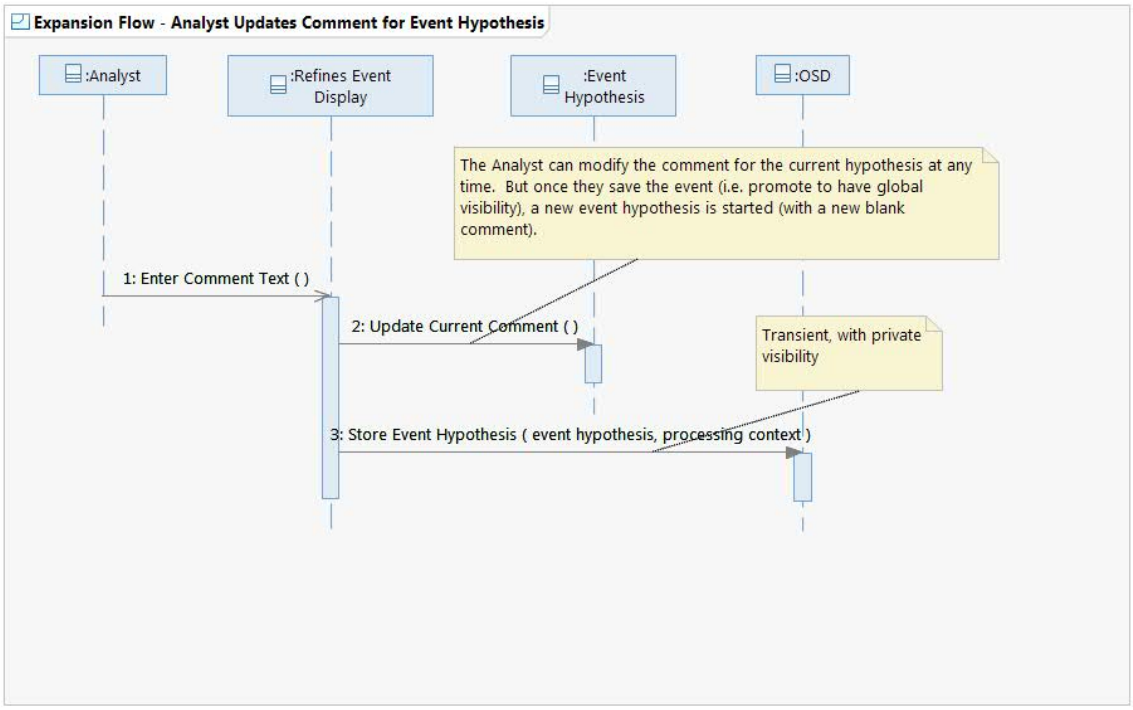


This flow shows Refines Event Display saving derived waveforms on Analyst request. The Analyst may select to save derived waveforms even when there are no signals detected on the waveforms.

Operation Descriptions

Operation: OSD::Store Waveform()
Store the given Waveform with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Expansion Flow - Analyst Updates Comment for Event Hypothesis

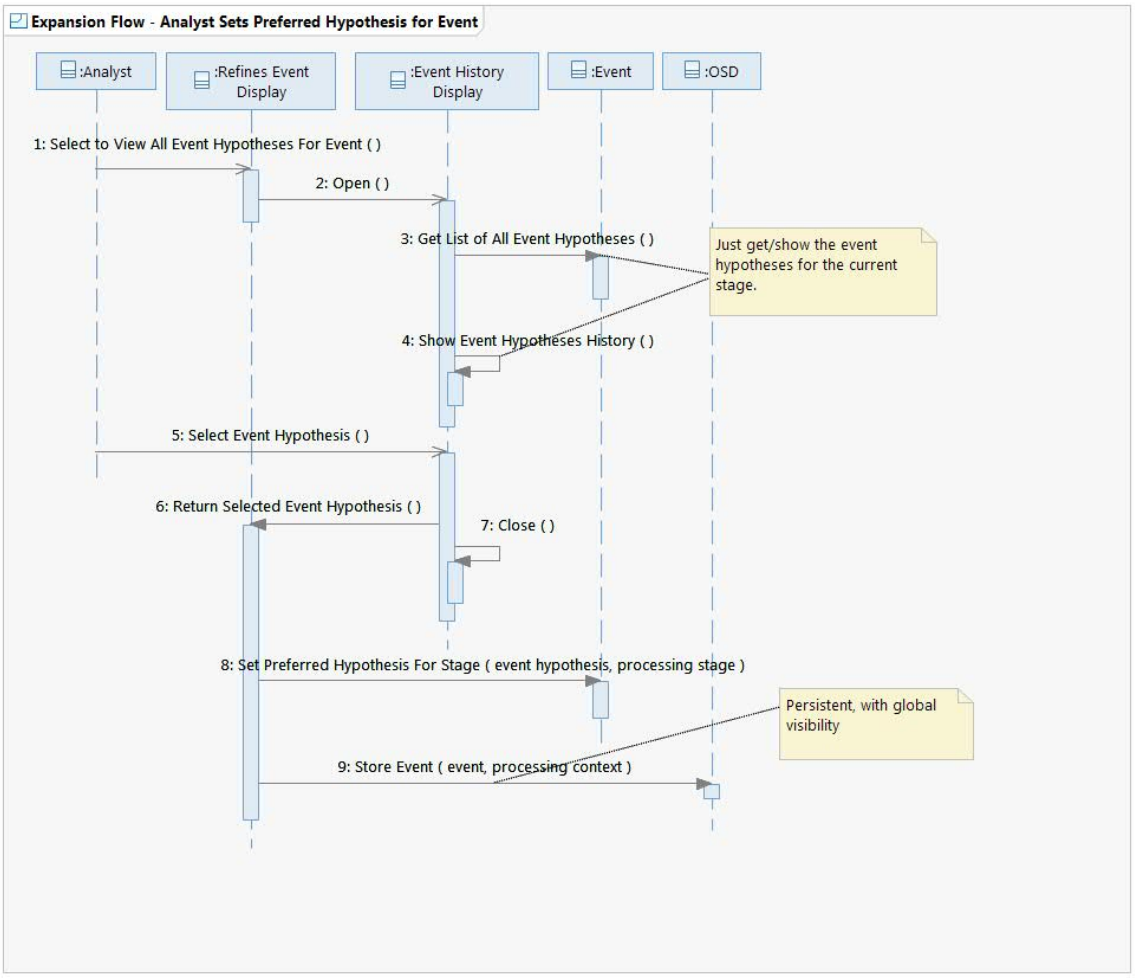


This flow shows how the Analyst updates comments on the Event Hypothesis.

Operation Descriptions

Operation: OSD::Store Event Hypothesis()
Store the given Event Hypothesis with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Expansion Flow - Analyst Sets Preferred Hypothesis for Event



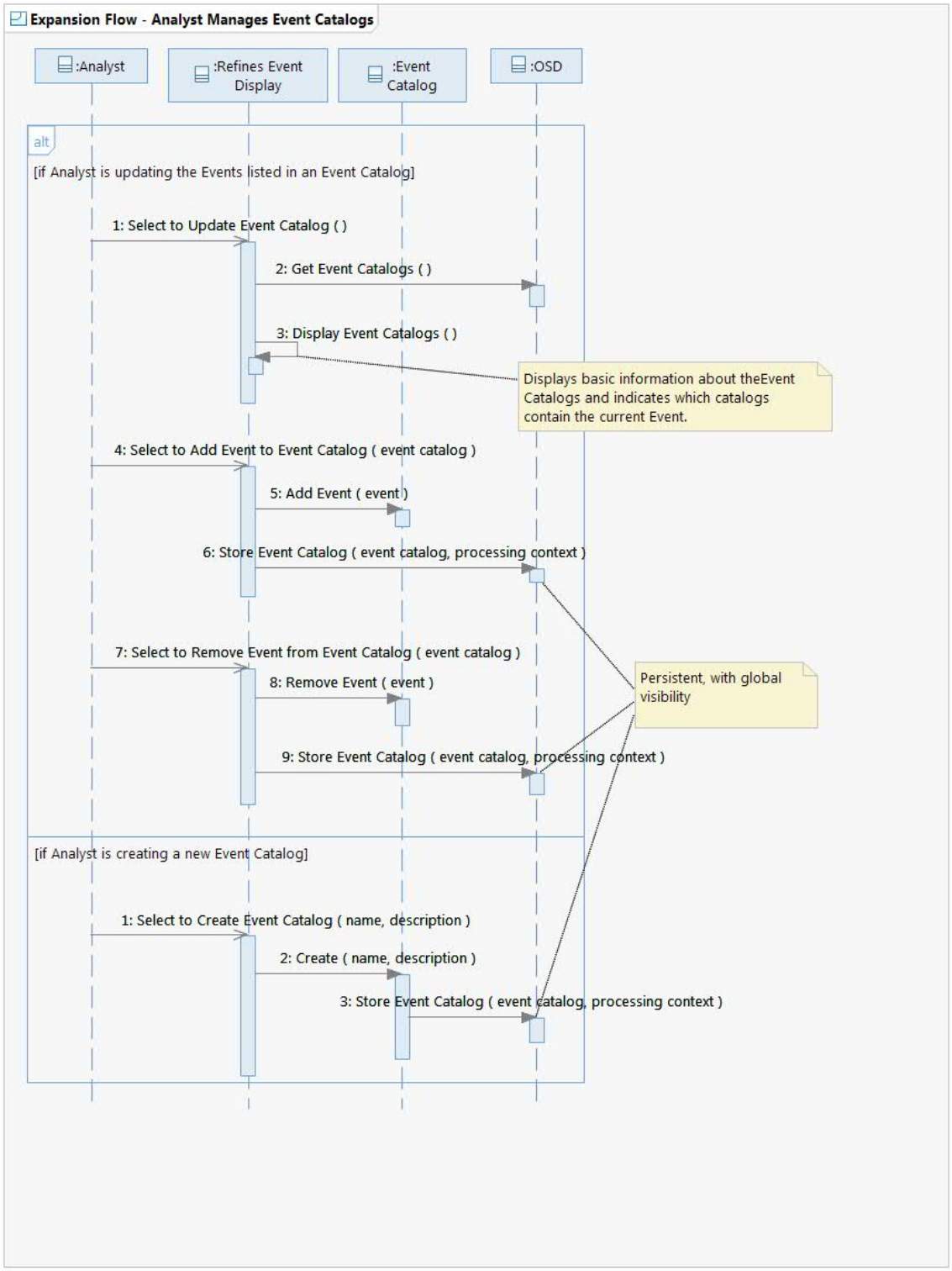
This flow shows the Analyst designating an event hypothesis as preferred for a specified processing stage. Marking an hypothesis as preferred causes the event to be immediately stored in a global context (visible to other analysts). Storing the Event triggers an OSD callback to Analyzes Events Display which determines if the Event's new preferred Event Hypothesis is in conflict and updates the Event Completion Status if necessary (see 'Analyzes Events' UCR "Alternate Flow – Refines Event Display - Handle OSD Callbacks").

Operation Descriptions

Operation: OSD::Store Event()
Store the given Event with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Operation: Event::Get List of All Event Hypotheses()
Return a list of all the Event Hypothesis for the given Event, including summary information such as the processing stage and which Event Hypotheses have been designated as preferred.

Expansion Flow - Analyst Manages Event Catalogs

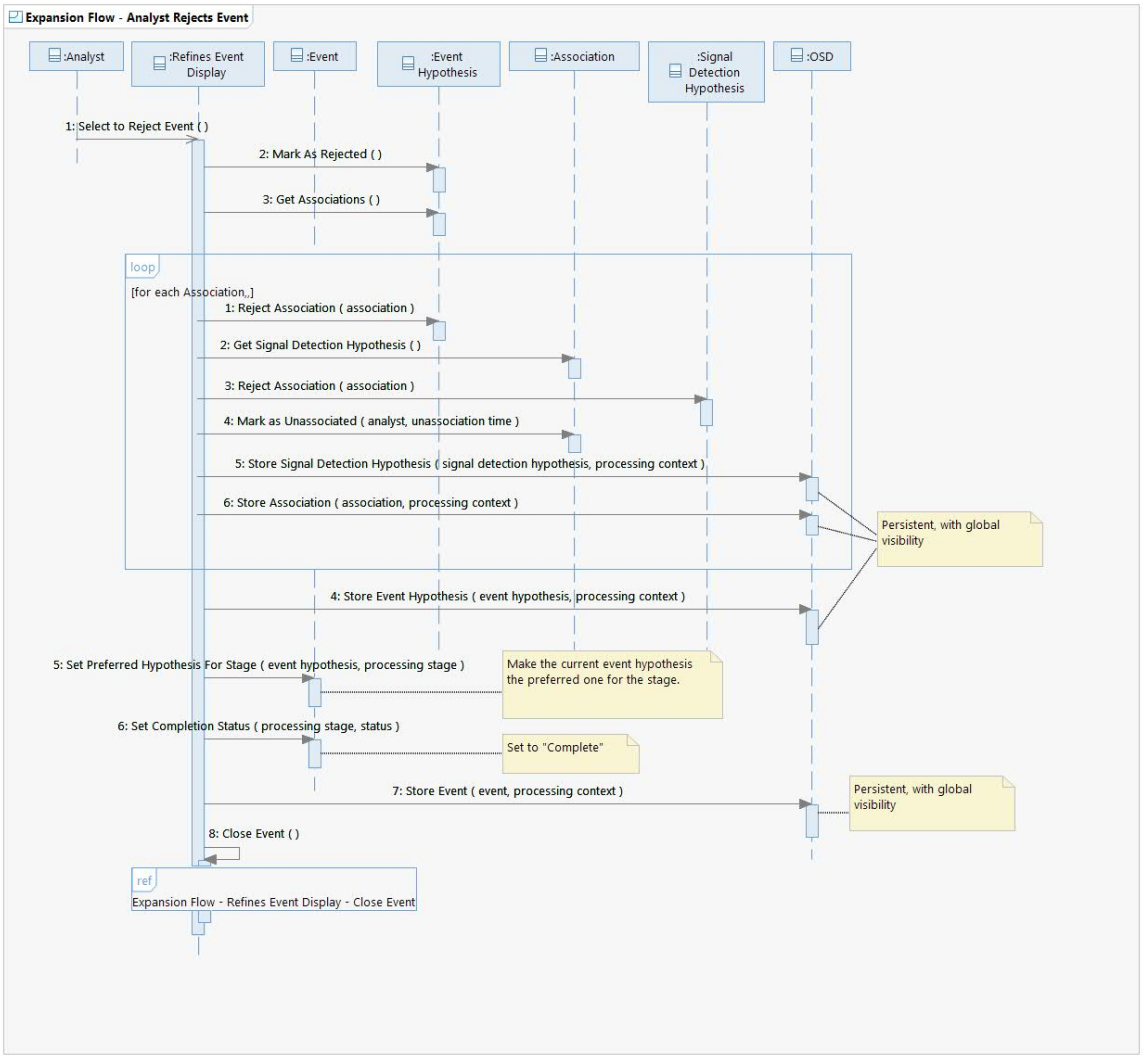


This flow shows the Analyst updating an Event Catalog by adding and removing Events from the catalog. The flow also shows the Analyst creating a new Event Catalog with a specified name and description.

Operation Descriptions

None

Expansion Flow - Analyst Rejects Event



This flow shows how the Refines Event Display handles rejecting an event. Rejecting an event is accomplished by unassociating all signal detection hypotheses from the current event hypothesis, making the current event hypothesis the preferred hypothesis for the current stage, saving the Event, Event Hypothesis, Signal Detection Hypothesis, and Association objects, and closing the Refines Event Display. Unassociating a Signal Detection Hypothesis and an Event Hypothesis removes the association relationship between the Signal Detection Hypothesis and the Association class and the association relationship between the Event Hypothesis and Association class. The Association class still has association relationships to the Signal Detection Hypothesis and Event Hypothesis classes for tracking event history and provenance.

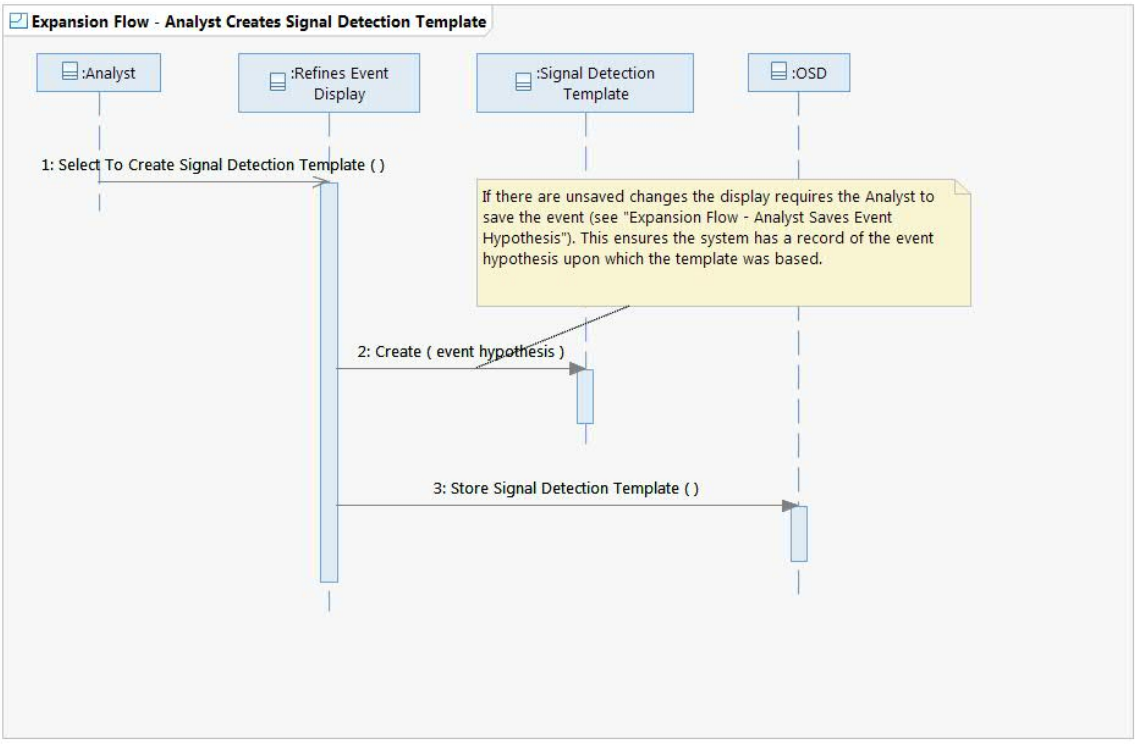
Operation Descriptions

Operation: OSD::Store Event Hypothesis()
Store the given Event Hypothesis with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Operation: OSD::Store Event()
Store the given Event with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Operation: Event::Set Completion Status()
Set the Event Completion Status of an Event in the given processing stage to the given value.

Expansion Flow - Analyst Creates Signal Detection Template

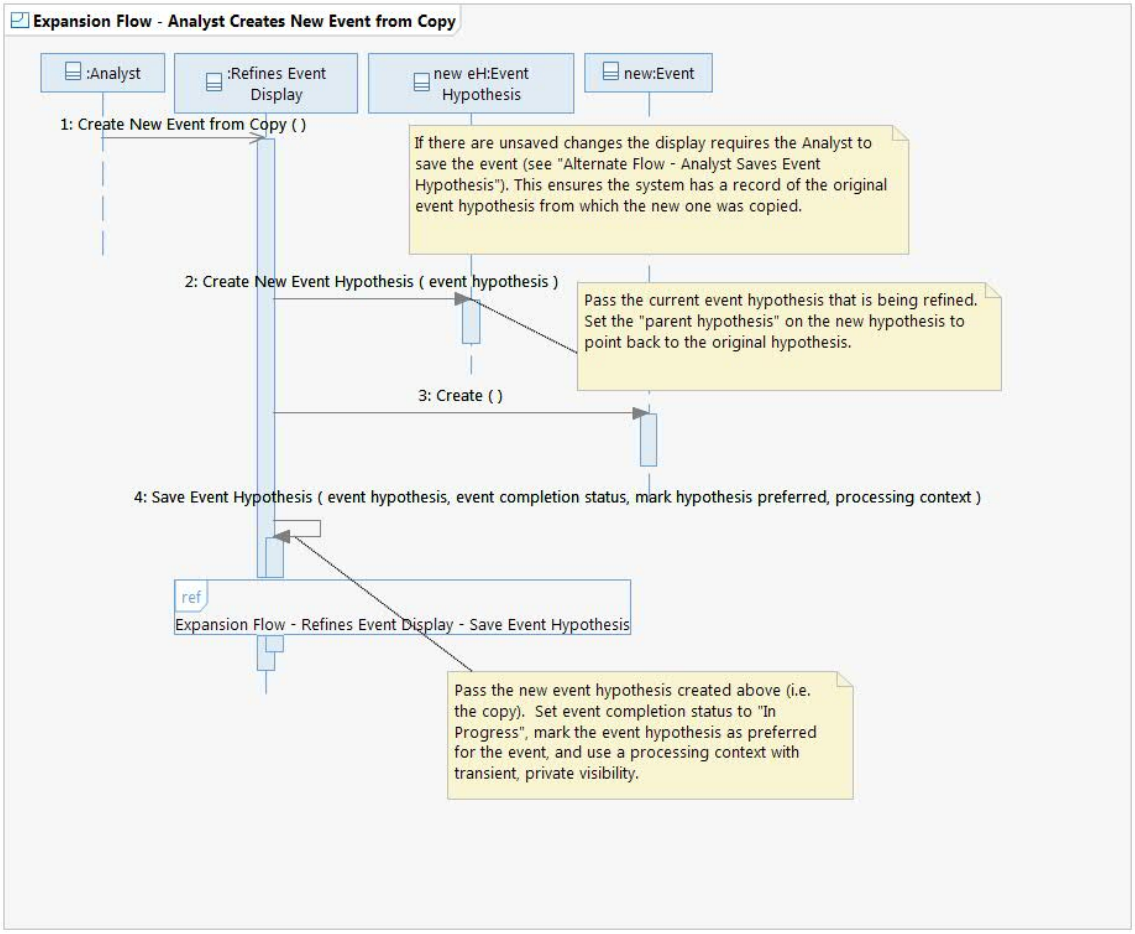


The flow shows the Analyst creating a new Signal Detection Template from the current Event Hypothesis. Stored templates may be applied by the Analyst when building a new Event or associating additional detections to an existing Event Hypothesis (see "Builds Event" UCR).

Operation Descriptions

None

Expansion Flow - Analyst Creates New Event from Copy



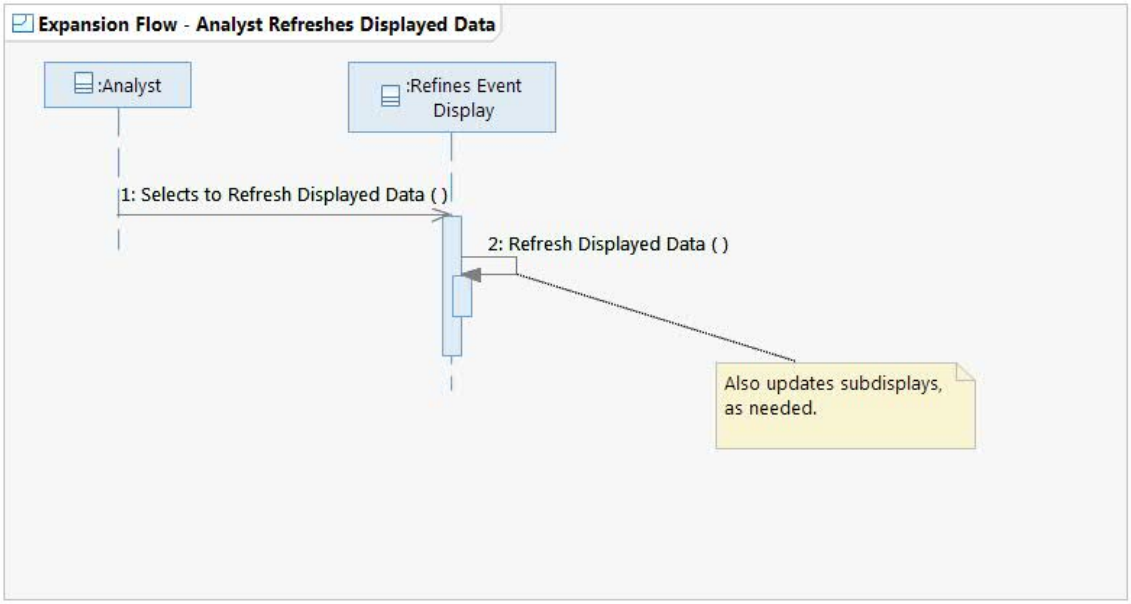
This flow shows the Analyst creating a new event by copying the Event they are currently refining. The Analyst must first save the Event and Event Hypothesis being copied in order to track provenance of the new Event. Refines Event Display stores the new Event and Event Hypothesis, triggering OSD callbacks related to the new Event, Event Hypothesis, and Associations. Analyzes Event Display responds to the callbacks by adding the new Event to the event list, checking for conflicts between the new and original Events, and marking the conflicts (see 'Analyzes Events' UCR "Alternate Flow – Analyzes Event Display - Handle OSD Callbacks"). Since the new Event Hypothesis has Associations to the same Signal Detection Hypotheses as the original Event Hypothesis, each of the associations is a conflict. The Analyst will need to refine each event individually to manually resolve these conflicts.

Operation Descriptions

Operation: Event Hypothesis::Create New Event Hypothesis()
Create a copy of the given Event Hypothesis. The copy has all of the same information as the original (e.g. same detections, location, etc.), with the following exceptions:

- The copy points to the original as its parent
- The copy starts out with an empty Analyst comment

Expansion Flow - Analyst Refreshes Displayed Data

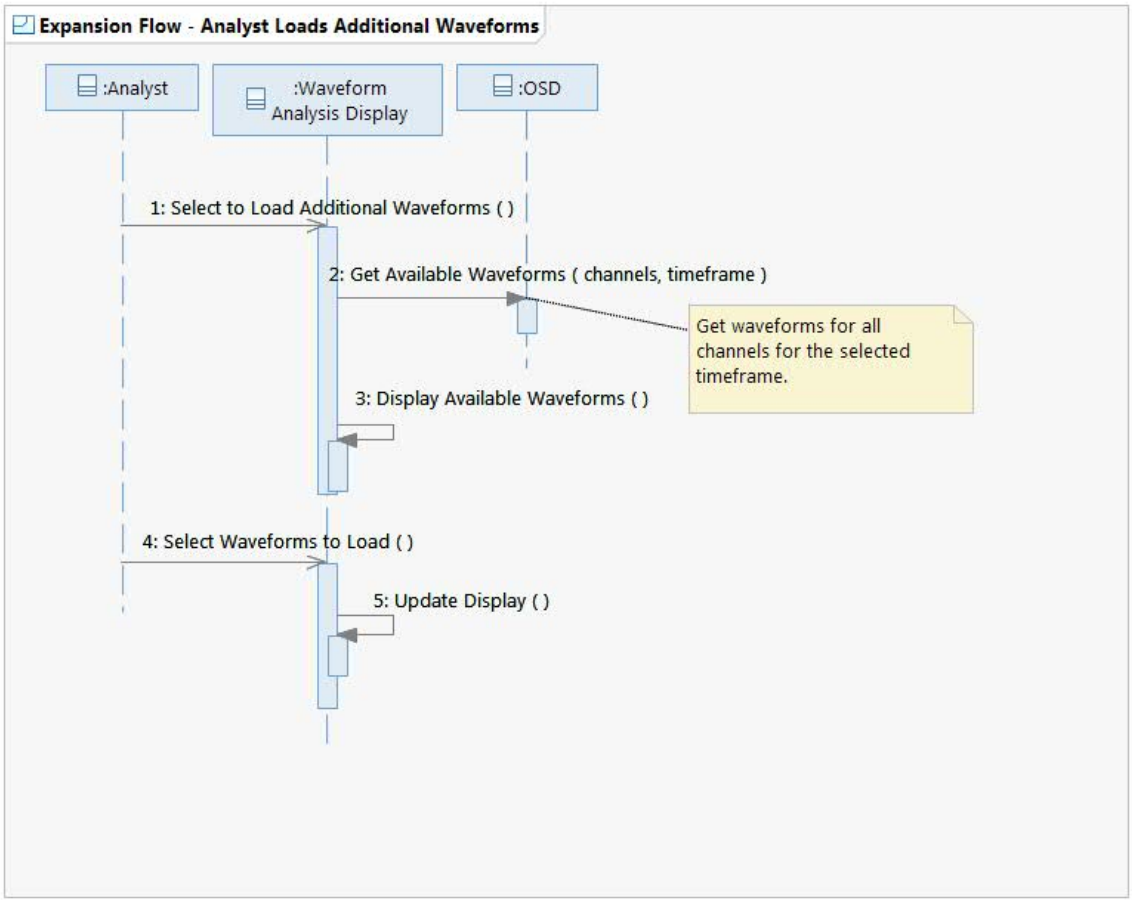


This flow shows how the Analyst refreshes his/her display to show the latest waveforms, Signal Detections and Signal Detection associations. The Analyst needs this capability since late-arriving waveforms, Signal Detections and Signal Detection associations made by other Analysts are not automatically displayed to the Analyst. The Analyst is notified when there is new data that is not shown on the display (see "Alternate Flow - Refines Event Display - Handle OSD Callbacks"). The Analyst may refresh the display to show that data at any time via this flow. Note that the display does not need to retrieve the new data from the OSD since it already has it due to subscriptions with the OSD (see "Expansion Flow - Refines Event Display - Open Event").

Operation Descriptions

Operation: Refines Event Display::Refresh Displayed Data()
Update displayed waveforms and Signal Detections to reflect the current state within the processing stage.

Expansion Flow - Analyst Loads Additional Waveforms



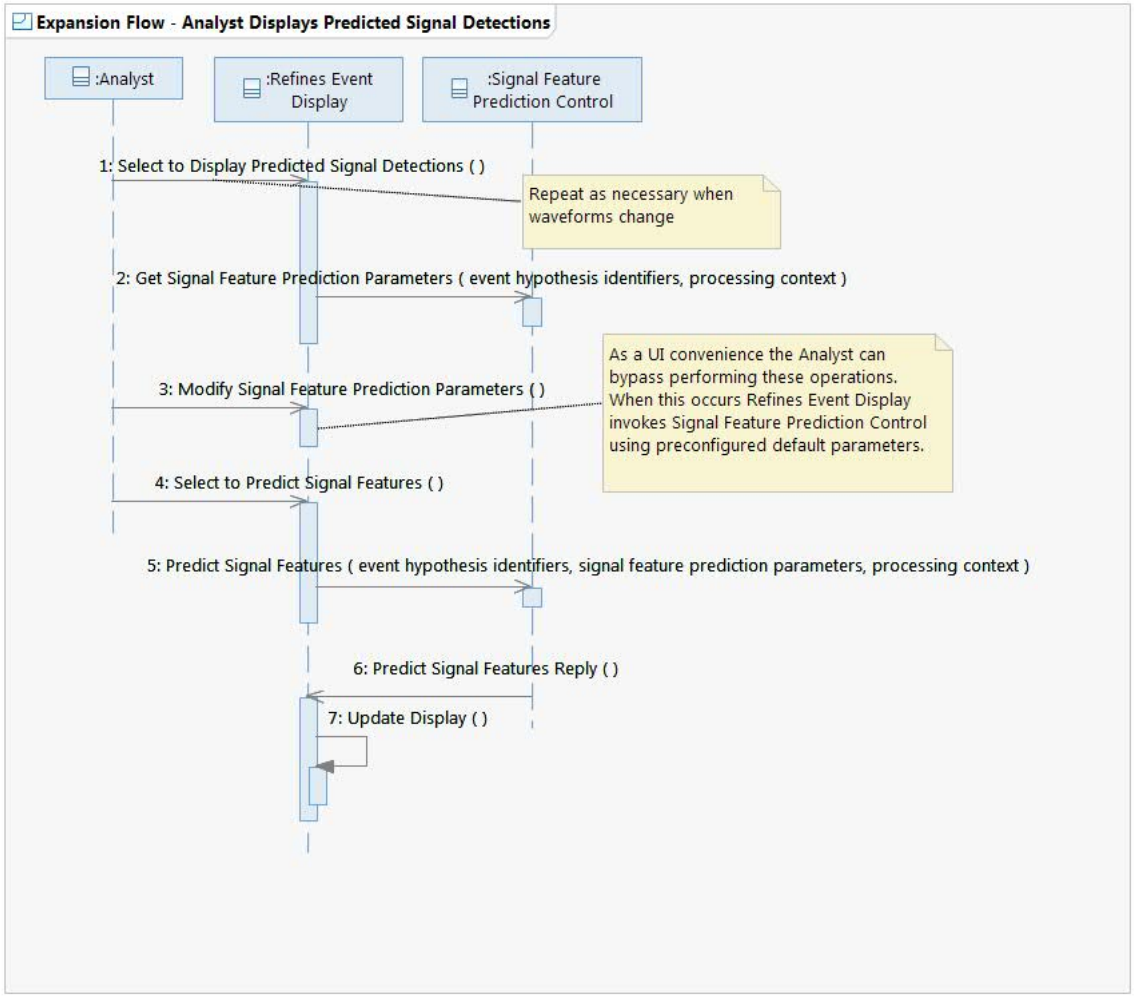
This flow shows the Analyst loading additional waveforms into Waveform Analysis Display. The Waveform Analysis Display uses the OSD to find all Waveforms stored on the System for a timeframe corresponding to the Analyst's current Processing Activity. These Waveforms could be from Stations not configured for default use in interactive Analysis (see 'Configures Station Usage' UCR). The Analyst selects which Waveforms to load and Waveform Analysis Display updates itself to display the selected Waveforms.

Operation Descriptions

Operation: OSD::Get Available Waveforms()

The Analyst can retrieve additional waveforms that were received but not configured to be used during automatic processing.

Expansion Flow - Analyst Displays Predicted Signal Detections

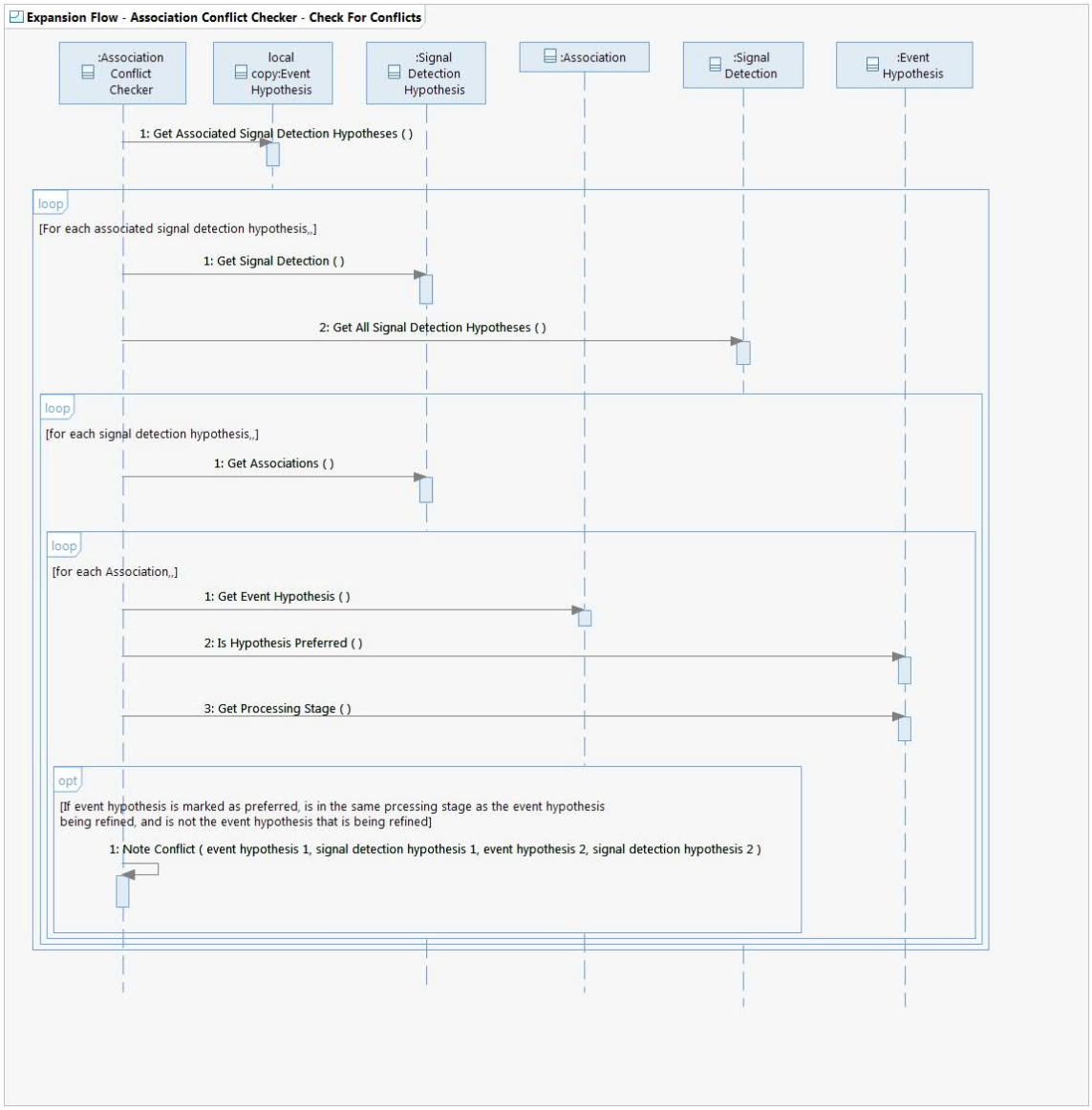


This flow shows the Analyst displaying predicted signal detections on the Refines Event Display. The Refines Event Display computes and displays predicted signal detections for all visible waveforms in all loaded events.

Operation Descriptions

Operation: Refines Event Display::Update Display()
Update the display of the Event Hypothesis that is currently being refined to reflect any changes that may have occurred. Indicate items that are out-of-date or inconsistent (e.g. beam may be out-of-date after refining Event location).

Expansion Flow - Association Conflict Checker - Check For Conflicts

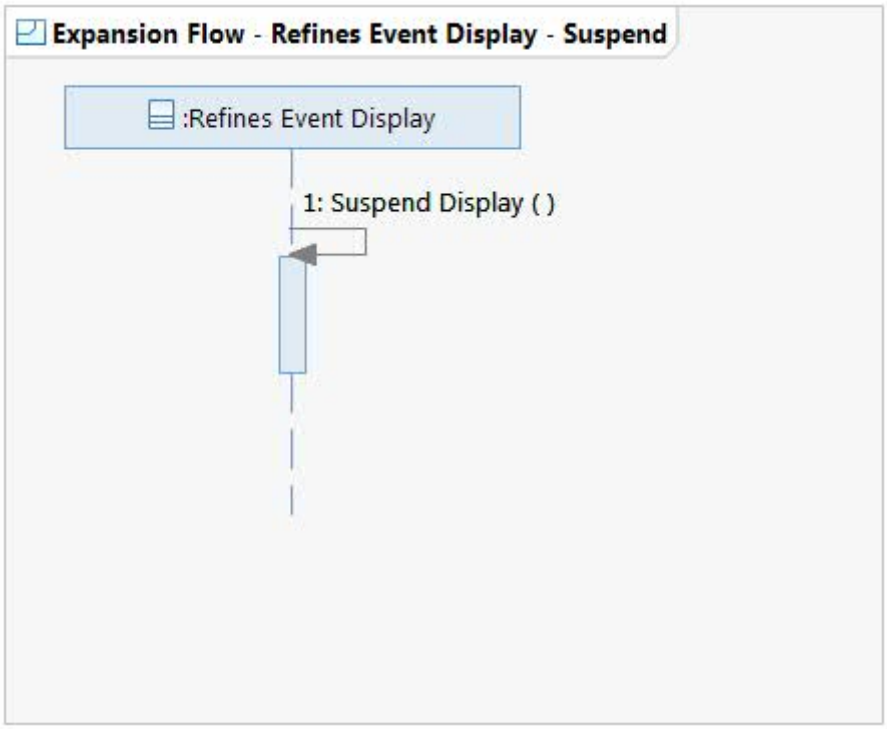


This flow shows how the Association Conflict Checker checks for association conflicts between the local (unsaved) Event Hypothesis and other Event Hypotheses. The local Event Hypothesis is input to this flow. The flow returns a list of association conflicts. Note that, by definition, a conflict can only exist with an Event Hypothesis that is marked as preferred. Note also that each Event can have at most one Event Hypothesis marked as preferred for each processing stage.

Operation Descriptions

None

Expansion Flow - Refines Event Display - Suspend

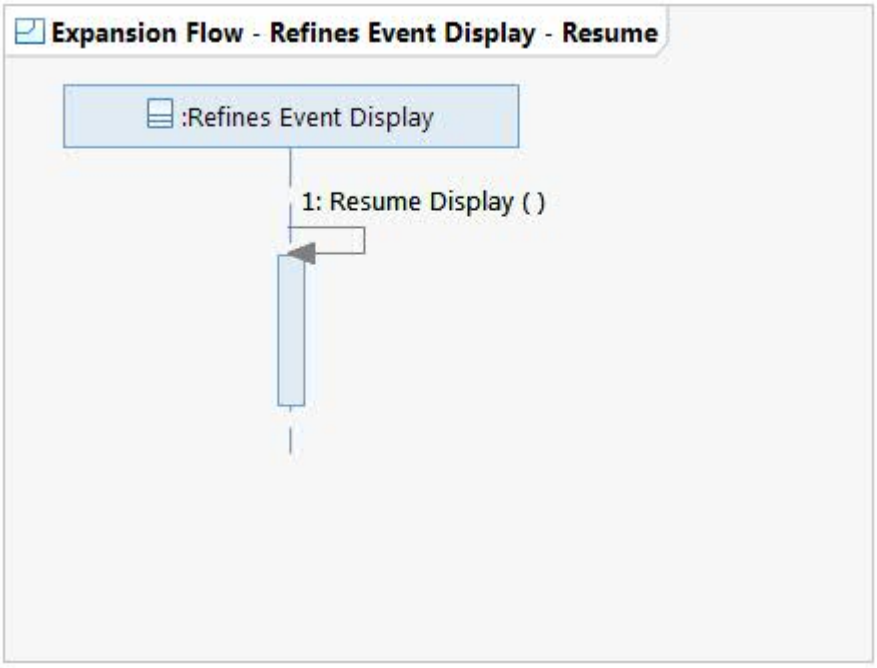


The Analyzes Event Display allows multiple Events to be selected for refinement but only one Refines Event Display can be active at a time. When one Refines Event Display becomes active the Analyzes Event Display suspends other open Refines Event Displays. While a Refines Event Display is suspended the display is hidden but the work in progress for an Event is maintained and the Refines Event display continues to receive OSD callbacks.

Operation Descriptions

Operation: Refines Event Display::Suspend Display()
The Refines Event Display for an Event is suspended when another Refines Event Display is opened. The suspended display instance retains the work in progress on the Event and continues to receive OSD callbacks but work on refining the event is temporarily halted and the display is hidden.

Expansion Flow - Refines Event Display - Resume

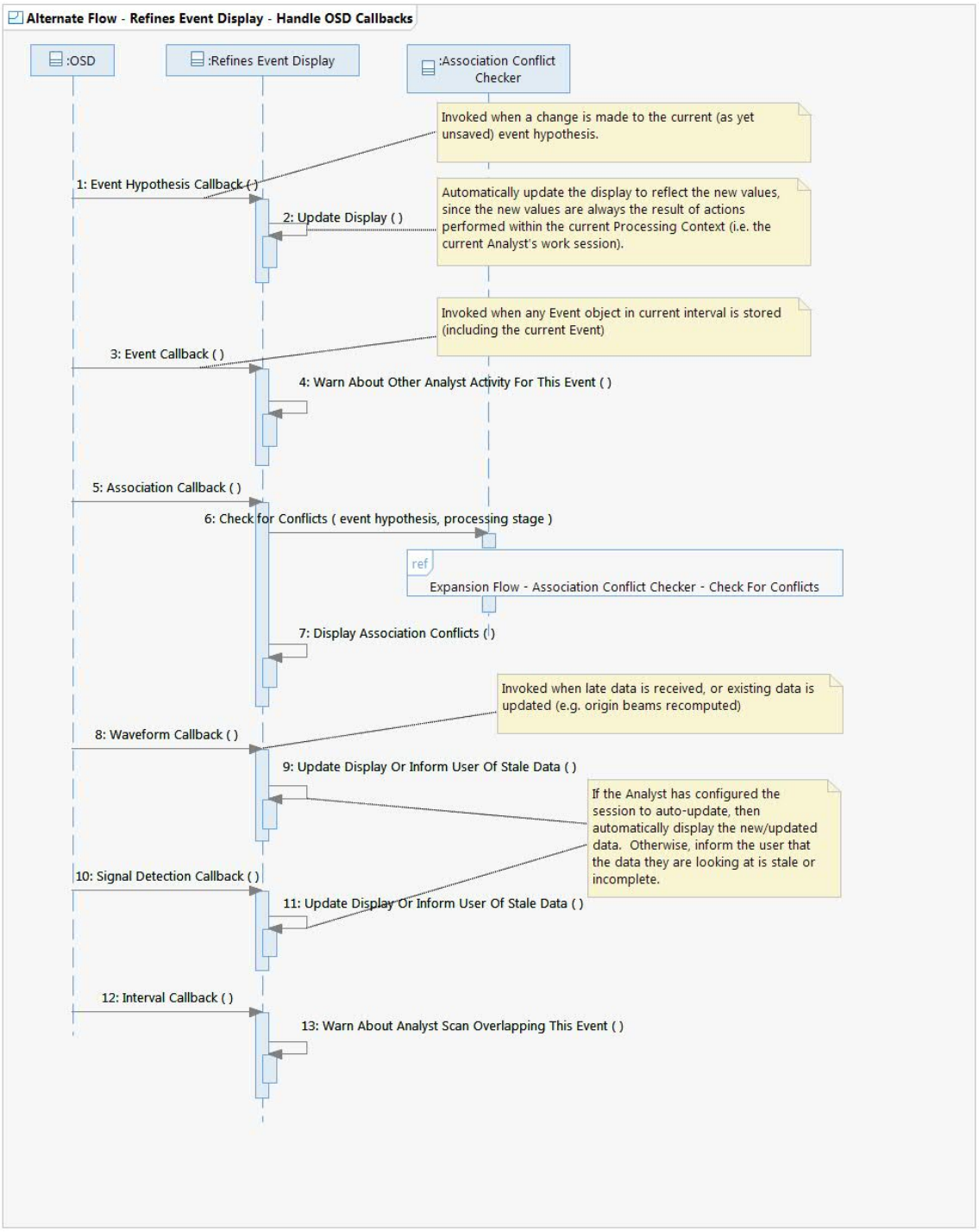


When the Analyst selects to open a Refines Event Display that was previously opened but suspended the Analyzes Event Display resumes the selected Refines Event Display. The Refines Event Display updates and becomes visible allowing the Analyst to continue to refine the Event.

Operation Descriptions

Operation: Refines Event Display::Resume Display()
When the Analyst selects to open a Refines Event Display that was previously opened but suspended the Analyzes Event Display resumes the selected Refines Event Display. The Refines Event Display updates and becomes visible allowing the Analyst to continue to refine the Event.

Alternate Flow - Refines Event Display - Handle OSD Callbacks



This flow shows how the Refines Event Display handles various callbacks from the OSD. The Refines Event Display subscribes for the current Event Hypothesis in order to monitor updates to it as a result of executed Processing Sequences. The display subscribes for all Events in the interval in order to monitor other Analyst activity on the current Event and to check for association conflicts with other Events. The display subscribes for Signal Detections and

Waveforms in order to display updates to that information made by other analysts. The display subscribes for Intervals to determine if an interval that overlaps the current Event is under active review by another analyst.

Operation Descriptions

Operation: Refines Event Display::Event Callback()
Callback invoked any time there is a change in the subscribed Event (e.g. a new Event Hypothesis for the Event is saved, or the preferred Hypothesis for a processing stage changes).

Operation: Refines Event Display::Warn About Other Analyst Activity For This Event()
Warn the current Analyst about another Analyst working on the current Event.

Operation: Refines Event Display::Signal Detection Callback()
Invoked any time the set of Signal Detections that fall within the current time interval changes. The callback indicates what changed.

Operation: Refines Event Display::Waveform Callback()
Invoked any time new raw or derived waveforms overlapping the time of the Event are received (e.g. late data, beams).

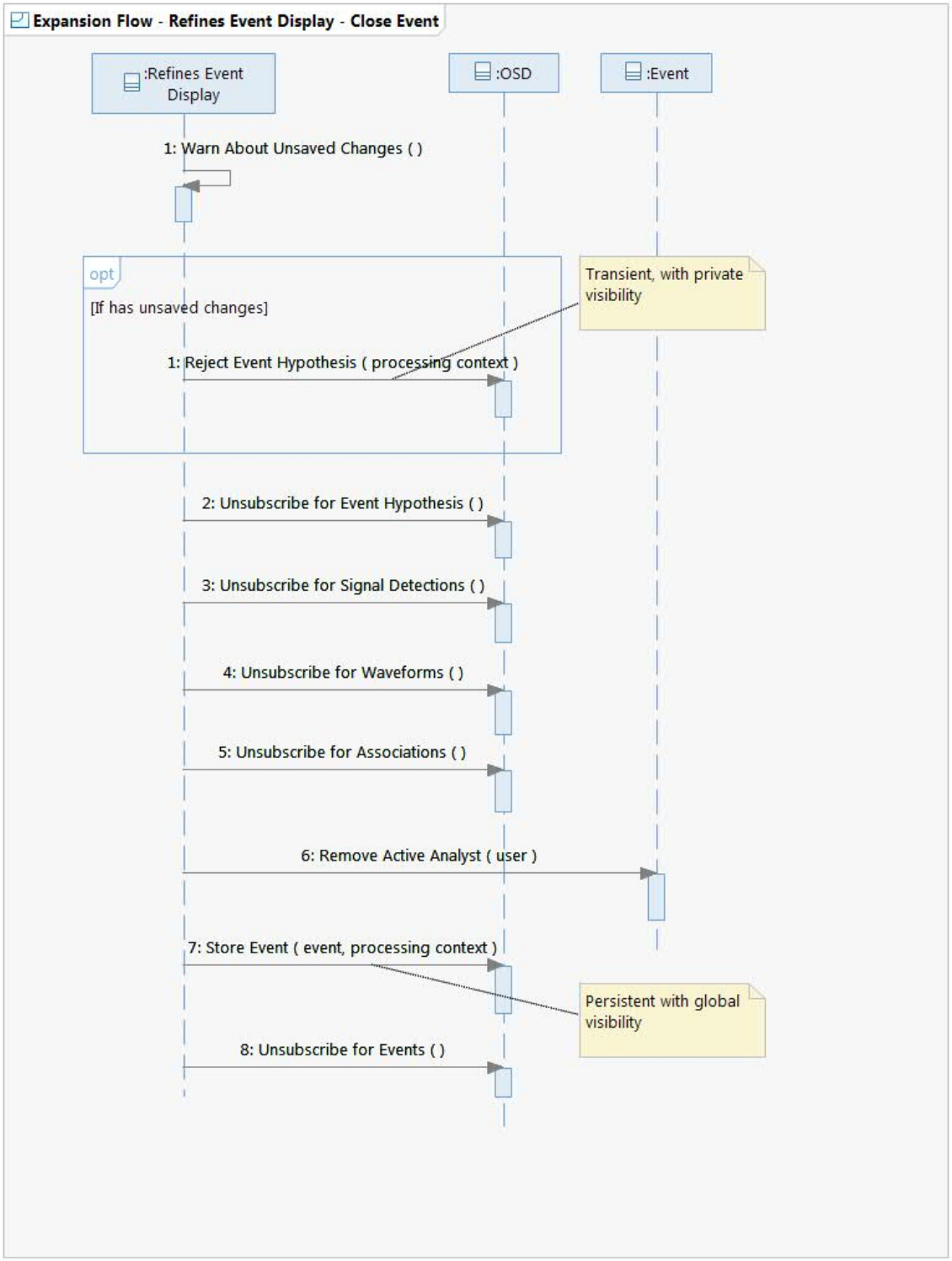
Operation: Refines Event Display::Event Hypothesis Callback()
Callback invoked whenever portions of the Event Hypothesis are changed by the system. This callback can only occur as part of automatic processing sequences executed by the Processing Sequence Control mechanism, since changes made by other Analysts are stored in separate Event Hypotheses.

Operation: Refines Event Display::Update Display()
Update the display of the Event Hypothesis that is currently being refined to reflect any changes that may have occurred. Indicate items that are out-of-date or inconsistent (e.g. beam may be out-of-date after refining Event location).

Operation: Association Conflict Checker::Check for Conflicts()
Checks the given Event Hypothesis against the other Events in the processing stage for association conflicts. Only check for conflicts between Event Hypotheses marked as preferred. A conflict exists if a Signal Detection is associated to more than one preferred Hypothesis in the processing stage.

Operation: Refines Event Display::Warn About Analyst Scan Overlapping This Event()
Warn the Analyst if the interval that overlaps the current Event being refined is under active review by another Analyst.

Expansion Flow - Refines Event Display - Close Event



This flow shows what the Refines Event Display does upon closing the current Event.

Operation Descriptions

Operation: OSD::Store Event()
Store the given Event with the given lifespan (persistent vs. transient) and visibility (private vs. global) as specified by the given Processing Context and notify relevant subscribers via callbacks.

Operation: OSD::Reject Event Hypothesis()
Remove Event Hypothesis and update Event and associated information for the removed Event Hypothesis.

State Machine Diagrams

None

SSD Mappings

General:

S-1157: [*Objective / Priority 2*] The System shall provide the Analyst the capability to view newly acquired waveform data within 1 minute of acquisition.

S-1296: [*Threshold*] The System shall store the processing time period(s) during which each Waveform QC Mask was applied to the underlying waveform data.

S-1297: [*Threshold*] The System shall store the Waveform QC Masks applied to the waveform data used for each waveform processing operation.

S-1298: [*Threshold*] The System shall store the channel masked by each Waveform QC Mask.

S-1299: [*Threshold*] The System shall store the identity of the user or processing stage creating each Waveform QC Mask.

S-1300: [*Threshold*] The System shall store the identity of the user or processing stage modifying each Waveform QC Mask.

S-1301: [*Threshold*] The System shall store the identity of the user or processing stage removing each Waveform QC Mask.

S-1302: [*Threshold*] The System shall store the time of each Waveform QC Mask creation.

S-1303: [*Threshold*] The System shall store the time of each Waveform QC Mask removal.

S-1304: [*Threshold*] The System shall store the time of each Waveform QC Mask modification.

S-1306: [*Threshold*] The System shall store the Analyst's rationale for creating a Waveform QC

Mask.

S-1307: [*Threshold*] The System shall store the Analyst's rationale for modifying a Waveform QC Mask.

S-1308: [*Threshold*] The System shall store the Analyst's rationale for removing a Waveform QC Mask.

S-1386: [*Threshold*] The System shall store the beam definition parameters for all beams.

S-1393: [*Threshold*] The System shall store all derived channels related to one or more signal detections.

S-1394: [*Threshold*] The System shall store derived waveform data with no related signal detections for the Operational Processing Time Period.

S-1421: [*Threshold*] The System shall store all signal detections.

S-1438: [*Threshold*] The System shall store time domain measurements.

S-1450: [*Threshold*] The System shall store polarization feature measurements.

S-1465: [*Threshold*] The System shall store frequency domain waveform measurements.

S-1486: [*Threshold*] The System shall store fk spectra measurements.

S-1532: [*Threshold*] The System shall provide the Analyst the capability to reject an event hypothesis.

S-1574: [*Threshold*] The System shall provide the System User the capability to view station quality metrics.

S-1576: [*Threshold*] The System shall store the station quality metrics for all stations for each event hypothesis.

S-1580: [*Threshold*] The System shall recompute the event hypothesis quality metric for an event hypothesis when any of the event hypothesis quality statistics used to calculate the event hypothesis quality metric are updated.

S-1586: [*Threshold*] The System shall provide the Analyst the capability to view event hypothesis quality metrics.

S-1588: [*Threshold*] The System shall store the event quality metric for each event hypothesis.

S-1616: [*Threshold*] The System shall provide the Analyst the capability to designate the preferred event hypothesis for each event.

S-1618: [*Threshold*] The System shall store up to 300 unique event hypotheses for each event.

S-1619: [*Threshold*] The System shall store the confidence level of each computed event hypothesis location uncertainty bound.

S-1620: [*Threshold*] The System shall store the type (i.e., confidence, coverage, or k-weighted with the associated weights) of each location uncertainty bound.

S-1621: [*Threshold*] The System shall store modeling uncertainties for model based predictions of signal detection measurements.

S-1622: [*Threshold*] The System shall store uncertainties for observed signal detection measurements.

S-1623: [*Threshold*] The System shall store the sum squared weighted residual for each event hypothesis location.

S-1624: [*Threshold*] The System shall store the defining/non-defining state for each signal detection measurement associated to a stored event hypothesis.

S-1625: [*Threshold*] The System shall store a preferred event hypothesis for each event for each processing stage.

S-1626: [*Threshold*] The System shall store the processing stage during which each event hypothesis location was created.

S-1627: [*Threshold*] The System shall store the processing stage during which an event hypothesis is modified.

S-1628: [*Threshold*] The System shall store the processing stage that rejected an event.

S-1644: [*Threshold*] The System shall provide the Analyst the capability to manually align waveforms.

S-1645: [*Threshold*] The System shall provide the Analyst the capability to align waveforms based on travel time differences.

S-1646: [*Threshold*] The System shall provide the Analyst the capability to align waveforms based on optimal lag calculated by waveform cross correlation.

S-1663: [*Threshold*] The System shall store uncertainties for all event hypothesis magnitude estimates.

S-1664: [*Threshold*] The System shall store each single station magnitude estimate for each event hypothesis.

S-1665: [*Threshold*] The System shall store each network magnitude estimate for each event hypothesis.

S-1666: [*Threshold*] The System shall store the defining/non-defining state for each station magnitude associated to a stored event hypothesis.

S-1711: [*Objective / Priority 1*] The System shall store the type of ground motion used by moment tensor calculations.

S-1712: [*Objective / Priority 1*] The System shall store the filter applied to observed and synthetic waveforms when computing moment tensor solutions.

S-1713: [*Objective / Priority 1*] The System shall store the Green functions used to compute a moment tensor solution.

S-1714: [*Objective / Priority 1*] The System shall store the Earth models used to compute a moment tensor solution.

S-1715: [*Objective / Priority 1*] The System shall store the elements of moment tensor solutions.

S-1716: [*Objective / Priority 1*] The System shall store the percentage of deviatoric moment tensor solutions belonging to the double couple components.

S-1717: [*Objective / Priority 1*] The System shall store the double couple fault plane solution computed from a moment tensor solution.

S-1718: [*Objective / Priority 1*] The System shall store the scalar seismic moment computed from a moment tensor solution.

S-1719: [*Objective / Priority 1*] The System shall store the station specific goodness of fit between theoretical and observed waveforms for moment tensor solutions.

S-1735: [*Objective / Priority 1*] The System shall store the ε value computed for moment tensor solutions.

S-1736: [*Objective / Priority 1*] The System shall store the k value computed for moment tensor solutions.

S-1737: [*Objective / Priority 1*] The System shall store the uncertainty bounds on ε and k computed for moment tensor solutions.

S-1738: [*Objective / Priority 1*] The System shall store the confidence level of uncertainty bounds on ε and k computed for moment tensor solutions.

S-1816: [*Threshold*] The System shall store the earth model and version used to compute an

earth model prediction.

S-1817: [*Threshold*] The System shall store the corrections applied to earth model predictions.

S-1818: [*Threshold*] The System shall store the correction surface used to correct an earth model prediction.

S-1819: [*Threshold*] The System shall store the predicted slowness computed from a basemodel.

S-1820: [*Threshold*] The System shall store the uncertainties of a predicted slowness computed using a basemodel.

S-1821: [*Threshold*] The System shall store the predicted azimuths computed using a phase-specific basemodel.

S-1822: [*Threshold*] The System shall store the uncertainties of predicted azimuths computed using a basemodel.

S-1823: [*Threshold*] The System shall store the predicted travel-times computed from a basemodel.

S-1824: [*Threshold*] The System shall store the uncertainties of predicted travel-times computed using a basemodel.

S-1842: [*Threshold*] The System shall store predicted amplitude attenuation.

S-1843: [*Threshold*] The System shall store predicted amplitude attenuation uncertainties.

S-1876: [*Threshold*] The System shall notify Analysts working in a common processing stage if they are concurrently modifying event hypotheses for an event.

S-1877: [*Threshold*] The System shall notify Analysts working in a common processing stage if they are concurrently modifying signal detections in the same analysis time interval.

S-1878: [*Threshold*] The System shall provide the Analyst the capability to access and view all waveform data stored on the System.

S-1885: [*Threshold*] The System shall display 24 hours of continuous waveform data before the waveform displays flatline.

S-1915: [*Threshold*] The System shall provide the Analyst the capability to process data without altering another Analyst's existing solution.

S-1917: [*Threshold*] The System shall provide the Analyst the capability to add or remove an event from an event catalog.

S-1920: [*Threshold*] The System shall provide the Analyst the capability to view any saved event hypothesis.

S-1921: [*Threshold*] The System shall provide the Analyst the capability to enter comments for an event hypothesis.

S-1922: [*Threshold*] The System shall provide the Analyst the capability to view comments for an event hypothesis.

S-1927: [*Threshold*] The System shall provide the Analyst the capability to select signal detections as processing input based on a time interval for an entire network during an analysis session.

S-1928: [*Threshold*] The System shall provide the Analyst the capability to select signal detections as processing input based on a time interval for a selected subset of stations during an analysis session.

S-1929: [*Threshold*] The System shall provide the Analyst the capability to individually select signal detections as processing input during an analysis session.

S-1930: [*Threshold*] The System shall provide the Analyst the capability to store new event hypotheses created during interactive processing.

S-1947: [*Threshold*] The System shall implement user interfaces according to the User Interface Guidelines.

S-1967: [*Threshold*] The System shall store results from all stages of data processing.

S-1985: [*Threshold*] The System shall provide the System User the capability to view event hypothesis data on an interactive map.

S-1996: [*Threshold*] The System shall provide the System User the capability to access geospatial data.

S-2042: [*Threshold*] The System shall store automatic and interactive processing parameters in the database.

S-2043: [*Threshold*] The System shall store automatic and interactive processing results.

S-2044: [*Threshold*] The System shall store the relation of processing results to processing parameters in the database.

S-2164: [*Threshold*] The System shall access requested waveform data within one (1) minute of receipt by the Data Processing Partition.

S-2167: [*Threshold*] The System shall write a 6 hour or less time block of 40Hz waveform data

within the Operational Processing Time Period with a maximum 5 second latency. (Goal: 1 second.)

S-2168: [*Threshold*] The System shall read a 6 hour or less time block of 40Hz waveform data outside the Operational Processing Time Period with a maximum 10 second latency. (Goal: 2 seconds.)

S-2169: [*Threshold*] The System shall read a 6 hour or less time block of 40Hz waveform data within the Operational Processing Time Period with a maximum 5 second latency. (Goal: 1 second.)

S-2170: [*Threshold*] The System shall write a 6 hour or less time block of 40Hz waveform data outside the Operational Processing Time Period with a maximum 10 second latency. (Goal: 2 seconds.)

S-2223: [*Threshold*] The System shall store all data and derived processing results to persistent storage as soon as the data and/or derived processing results are available.

S-2417: [*Threshold*] The System shall store hydroacoustic signal detection groups

S-2603: [*Threshold*] The System shall provide the System User the capability to access requested waveform data.

S-2604: [*Threshold*] The System shall provide the Analyst the capability to access late-arriving waveform data within one (1) minute of receipt by the Data Processing Partition.

S-3025: [*Threshold*] The System shall provide the Analyst the capability to create a signal detection template from an existing event.

S-5708: [*Threshold*] The System shall read a 6 hour or less time block of processing results within the Operational Processing Time Period with a maximum 5 second latency. (Goal: 1 second.)

S-5709: [*Threshold*] The System shall write a 6 hour or less time block of processing results within the Operational Processing Time Period with a maximum 5 second latency. (Goal: 1 second.)

S-5712: [*Threshold*] The System shall read a 6 hour or less time block of processing results outside the Operational Processing Time Period with a maximum 10 second latency. (Goal: 2 seconds.)

S-5713: [*Threshold*] The System shall write a 6 hour or less time block of processing results from outside the Operational Processing Time Period with a maximum 10 second latency. (Goal: 2 seconds.)

S-5715: [*Threshold*] The System shall store wind velocity (including uncertainty) computed

from meteorological models.

S-5716: [*Threshold*] The System shall store temperature (including uncertainty) computed from meteorological models.

S-5717: [*Extensibility*] The System shall store gravity wave corrections to temperature predictions.

S-5720: [*Threshold*] The System shall store spectrograms.

S-5722: [*Threshold*] The System shall store power spectral density.

S-6469: [*Threshold*] The System shall store detection feature maps.

IDC Specific:

S-5612: [*Threshold*] The System shall provide the Analyst the capability to request auxiliary seismic waveform data from the Data Acquisition Partition.

S-5795: [*Threshold*] The System shall compute Event Consistency checks when an event hypothesis is saved.

Notes

General:
1. In this UCR and all of its child UCRs (e.g. "Refines Event Location", "Refines Event Magnitude", etc.), the display classes store computed results to transient storage via the OSD mechanism as the event is refined in order to trigger execution of configured processing sequences. These processing sequences are configured by the System Maintainer (see "Defines Processing Sequence" UCR), and are automatically executed by the Processing Sequence Control mechanism in response to OSD callbacks (the Processing Sequence Control mechanism is shown in "System Detects Events" UCR). These changes to the Event Hypothesis are stored with private visibility such that the changes are accessible only to the Analyst work session where the changes are being made; other Analysts cannot see the updates until the Analyst chooses to save the Event Hypothesis, at which time the Event Hypothesis is stored to persistent storage via the OSD mechanism. The storage of an Event Hypothesis to persistent storage may trigger additional processing sequences, as defined by the System Maintainer.

2. The Analyst may undo/redo editing operations while refining the event, but only back to the last save.

3. See "Marks Processing Stage Complete" UCR for a state machine diagram for Event Completion Status.

4. This UCR covers creation of signal detection templates. The Analyst applies such templates

when building new events (see "Builds Event" and "Scans Waveforms and Unassociated Detections" UCRs).

5. When the Refines Event Display is suspended the instance of the display maintains the current state of the refined Event and continues to receive OSD callbacks. If performance concerns become a factor then suspended displays could unsubscribe from OSD callbacks and saved transiently.

IDC Specific:
1. The Analyst uses "Expansion Flow – Analyst Loads Additional Waveforms" to request and load auxiliary station waveform data from the Data Acquisition Partition.

2. The System Maintainer may use Defines Processing Sequence Display (see 'Defines Processing Sequence' UCR) to configure a processing sequence to perform event consistency calculations after an event is saved (see 'System Accesses Event Consistency' UCR).

IDC Use Case Realization Report

UCR-08.05 Views Event History

Use Case Description

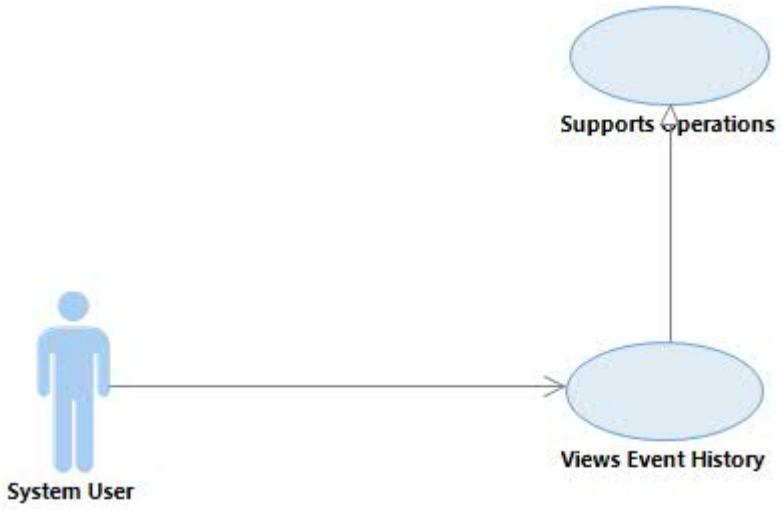
This architecturally significant use case describes how the System User observes the change history of a given event. The change history is a series of one or more saved event hypotheses. System Users view all the event hypotheses and the set of location solutions for each hypothesis. The System User views the relationship between event hypotheses including the preferred hypothesis for each processing stage. The event change history persists across work sessions for subsequent review.

This use case is architecturally significant because it describes viewing and comparing multiple versions of an event to review the history of how an event was formed and what data were available at each stage of event formation.

Architecture Description

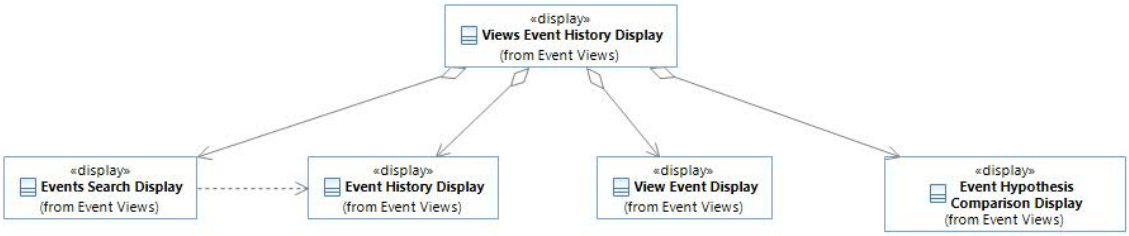
The System User opens the Views Event History Display to select and view an Event and the related Event Hypotheses. The System User selects an Event using the Event Search Display. The System User selects an event and opens the Event History Display. The Event History Display shows all the Event Hypotheses for an Event and the relationships between hypotheses stored by the OSD. The System User can select to view the information for an individual Event Hypothesis using the View Event Display or compare multiple hypotheses using the Event Hypothesis Comparison Display.

Use Case Diagram



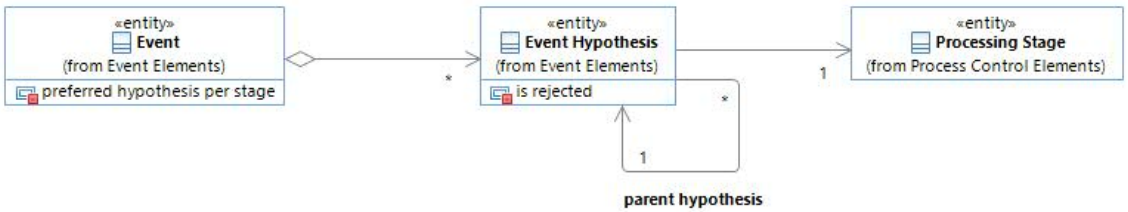
Class Diagrams

Classes - Displays



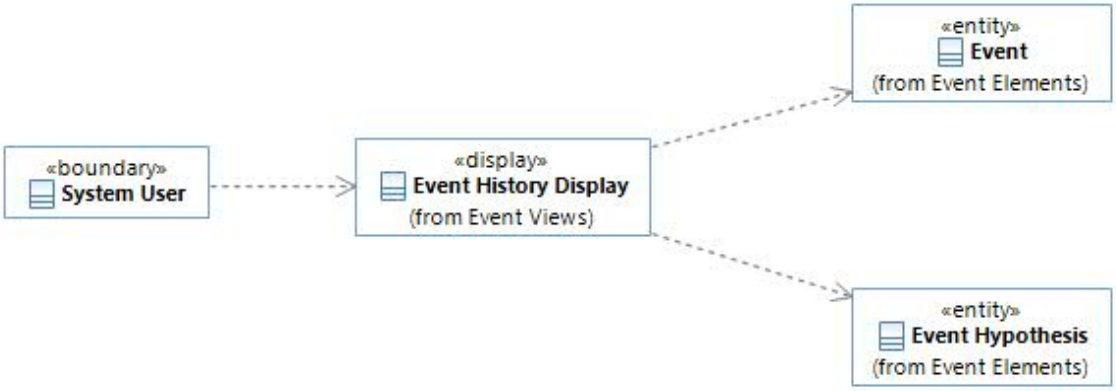
This diagram shows the display classes for selecting and viewing Events and Event Hypotheses. The System User uses the Event Search Display to select an Event. The Event History Display shows the relationships between Event Hypotheses for an Event. The View Event Display shows detailed information about an Event Hypothesis. The System User uses the Event Hypothesis Comparison Display to compare multiple hypotheses.

Classes - Event



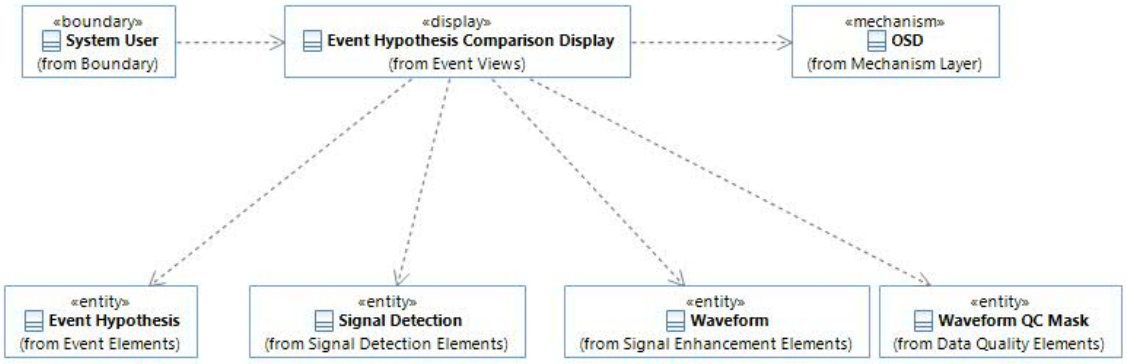
This diagram shows the relationships between Events and Event Hypotheses and the relationships between hypotheses. Each hypothesis is related to its parent hypothesis that was the basis for the child hypothesis. The parent hypothesis can be for the same event or a different event.

Classes - Event History Display



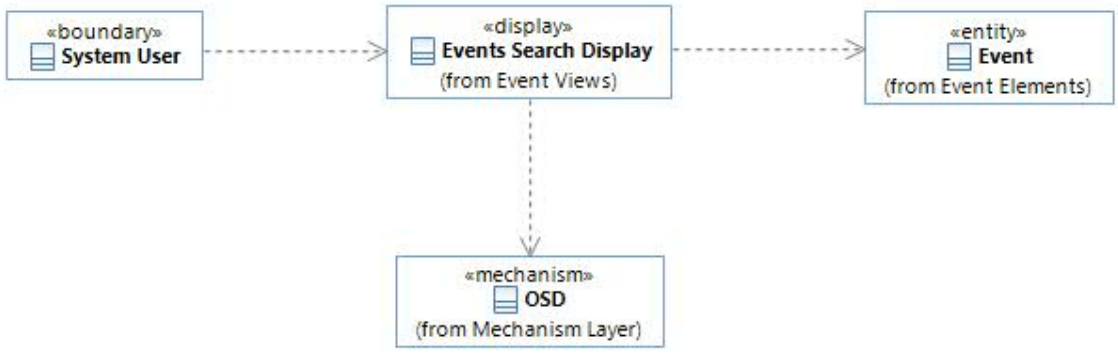
This diagram shows the Event History Display and related classes.

Classes - Event Hypotheses Comparison Display



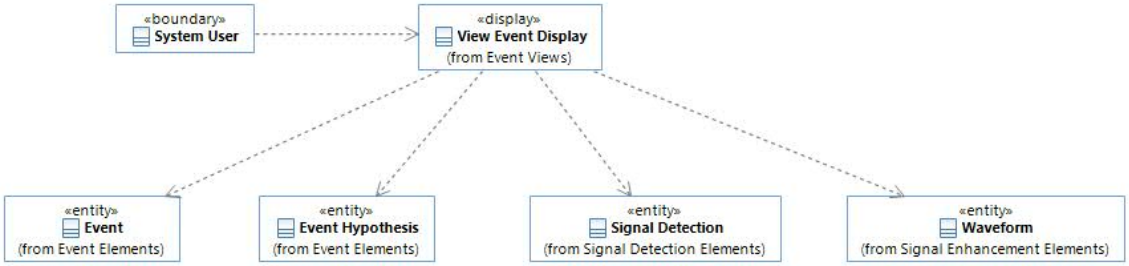
This diagram shows the Event Hypothesis Comparison Display and related classes.

Classes - Event Search Display



This diagram shows the Event Search Display. The System User uses the Event Search Display to select Events and retrieve the Events from the OSD.

Classes - View Event Display



This diagram shows the View Event Display and related classes that provide information about Events and Event Hypotheses.

Class Descriptions

<<boundary>> System User
Represents the System User actor.

<<display>> Events Search Display
Display that provides the Analyst the ability to search for similar Events and store Similar Events Search Results.

<<entity>> Event
Represents information about an Event. Keeps track of all the Event Hypotheses for the Event, which Event Hypothesis is the preferred one for each processing stage, the active analysts for the Event (i.e. whether the Event is under "active review"), whether the Event is "complete" for each processing stage, and other Event-related information.

<<entity>> Event Hypothesis
Represents geophysical information about an Event as determined by an Analyst or through pipeline processing. There can be multiple Event Hypotheses for the same Event (e.g. different associated Signal Detection Hypotheses, different location solutions).

<<entity>> Processing Stage
Represents a named stage of data processing, which may be part of the System Maintainer-defined workflow or an Analyst-defined stage outside the workflow. All Processing Results are associated to a Processing Stage.

<<entity>> Signal Detection
Represents information about a Signal Detection and keeps track of all the Signal Detection Hypotheses for the Signal Detection. Represents information about a Signal Detection and keeps track of all the Signal Detection Hypotheses for the Signal Detection. For an unassociated Signal Detection the preferred Signal Detection Hypothesis is the most recently created Signal Detection Hypothesis. For an associated Signal Detection the preferred Signal Detection Hypothesis is the one associated to a preferred Event Hypothesis.

<<entity>> Waveform

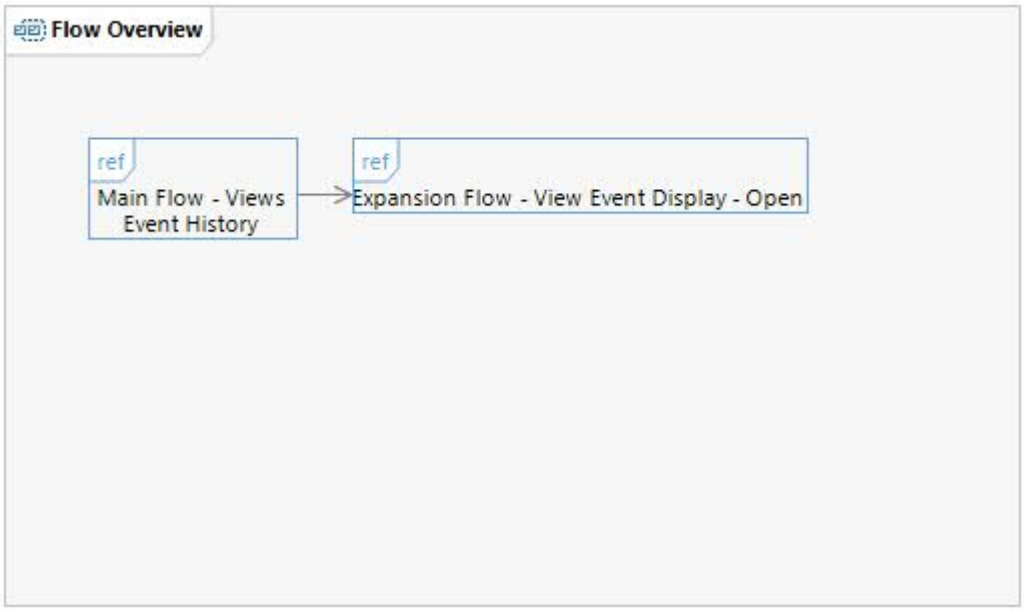
A Waveform represents a time-series of data from a Channel.

<<entity>> Waveform QC Mask
Represents information about a waveform quality control issue (e.g. data gap, repeated amplitude values, amplitude spikes, linear trends, calibration signals, invalid gain, noisy channels, authentication failures, Analyst defined, etc.) occurring on a waveform. Waveform QC Mask has attributes describing the type of quality issue it is masking, the waveform data time period it masks, and the enabled time period describing when the mask is (or was) used. Waveform QC Mask tracks the valid time for data provenance purposes since some masks will be created for transient quality issues (e.g. missing data that is later acquired).

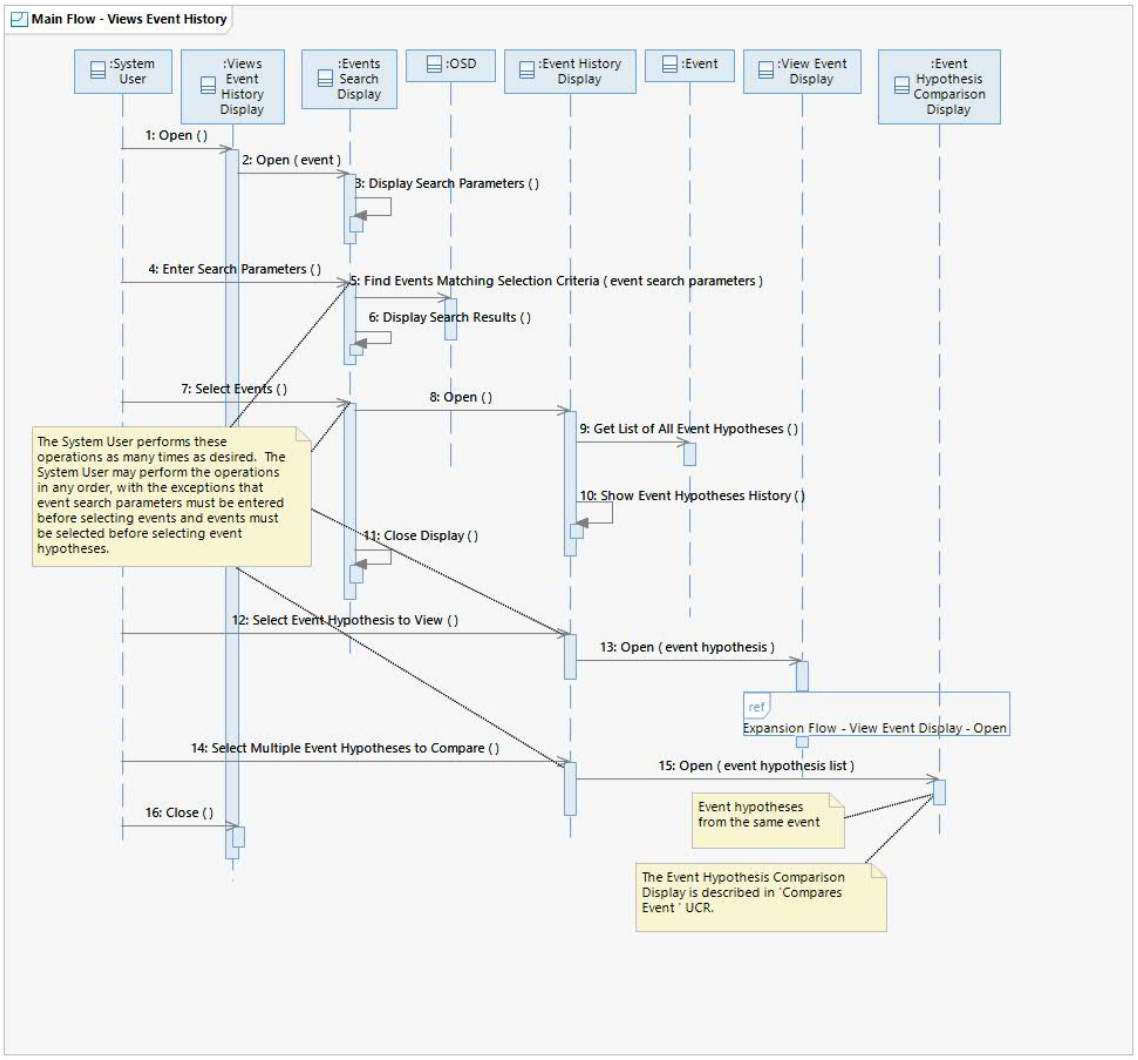
<<mechanism>> OSD
Represents the Object Storage and Distribution mechanism for storing and distributing data objects internally within the system.

Sequence Diagrams

Flow Overview



Main Flow - Views Event History



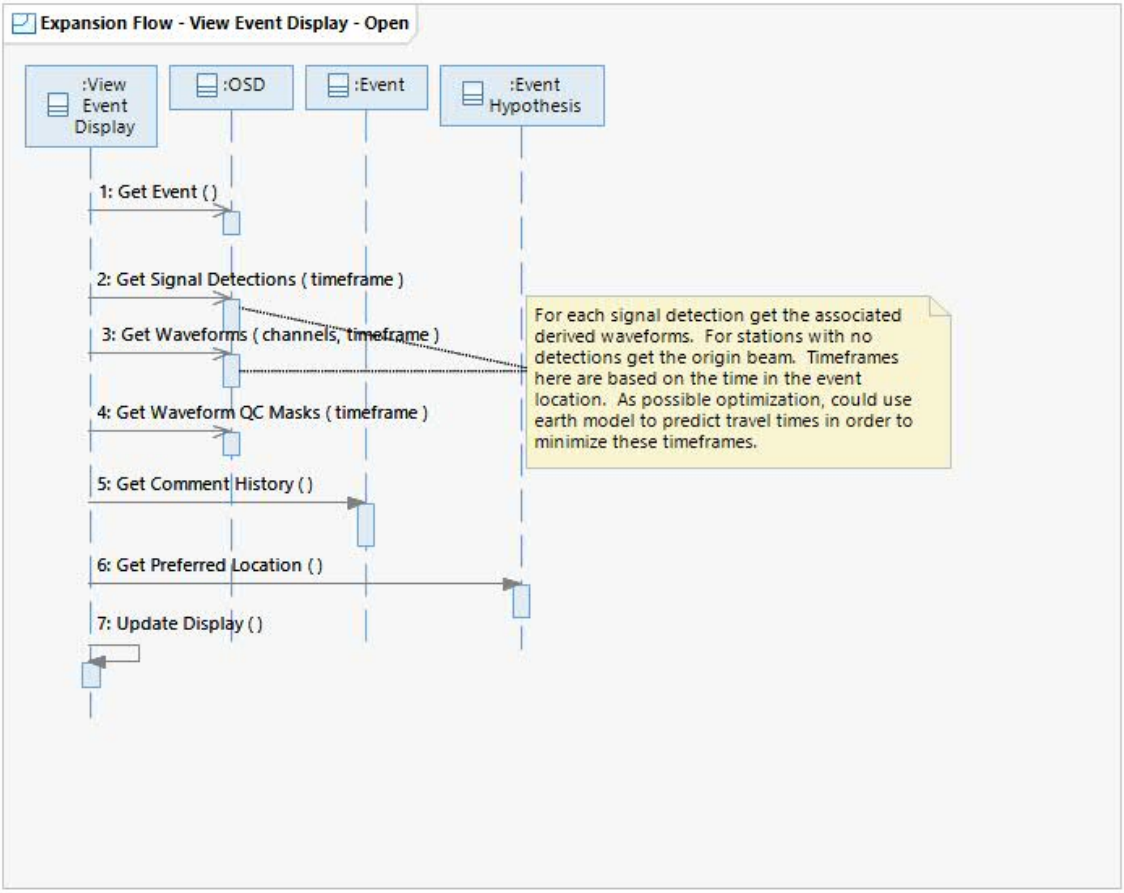
This flow shows how the System User selects an Event and views the event history. Optionally, the System User may view a read-only copy of a selected Event Hypothesis or compare multiple Event Hypotheses.

Operation Descriptions

Operation: Event::Get List of All Event Hypotheses()

Return a list of all the Event Hypothesis for the given Event, including summary information such as the processing stage and which Event Hypotheses have been designated as preferred.

Expansion Flow - View Event Display - Open



This flow shows how the View Event Display is created and the classes contributing information for the display.

Operation Descriptions

Operation: Event::Get Comment History()
Return all Analyst-entered comments associated with the Event.

State Machine Diagrams

None

SSD Mappings

General:

S-1292: [*Threshold*] The System shall provide the System User the capability to compare Waveform QC Masks generated by each processing stage for selected points in the processing history.

S-1926: [*Threshold*] The System shall provide the System User the capability to view the complete history of an event.

S-1947: [*Threshold*] The System shall implement user interfaces according to the User Interface Guidelines.

S-1985: [*Threshold*] The System shall provide the System User the capability to view event hypothesis data on an interactive map.

S-1996: [*Threshold*] The System shall provide the System User the capability to access geospatial data.

S-1999: [*Threshold*] The System shall provide the System User the capability to view tabular listings of the results of spatial processing of geospatial data.

S-2040: [*Threshold*] The System shall provide the System User the capability to retrieve stored processing results from computations.

Notes

General:
1. View Event Display is a read-only display of an Event Hypothesis and related information. The event hypothesis information is generated in 'System Detects Event' and 'Refines Event' UCs and includes station quality metrics, event hypothesis quality metrics, map displays including geographic regions and geospatial data, and both unassociated and associated signal detections. Displays will be similar to Analyst displays for 'Refines Event' UC.

2. Event Hypothesis Comparison Display shows QC Masks for associated waveforms per S-1292.

This page intentionally left blank.


Sandia National Laboratories