



Configuring and Using the Event Triggering System

Technical description, implementation details and
usage guide for the Event Triggering system

Studio Elektronike Rijeka, October 2012

Copyright © 2012 Studio Elektronike Rijeka

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior permission of the copyright owners.

The definitive version of this document is available at **www.wamster.net**.

Wamster is a trademark of Studio Elektronike Rijeka. All other brand, company, and product names are used for identification purposes only and may be trademarks that are the sole property of their respective owners.

Any comments relating to the material contained in this document may be submitted to:

Studio Elektronike Rijeka
Janka Polić Kamova 19
HR-51000 Rijeka

or by email to:

info@wamster.net

Table of Contents

Executive Summary.....	3
1 Event Triggering Overview	5
1.1 Role of Wamster in ad-hoc WAM systems	5
1.2 Event triggering in Wamster	5
1.3 High level overview of the WAMSTER system operation	6
1.4 How does event triggering work?.....	6
2 Creation of input synchrophasor data streams	8
3 Processing rules	9
3.1 Defining a processing rule for a specific quantity.....	9
3.2 Chaining processing rules	9
4 Creating event triggers.....	11
5 Using the online interface.....	12
6 Example: creating processing rules.....	14



Wamster Event Triggering system allows users to configure custom triggers for various measured and calculated quantities. All PMU devices are monitored simultaneously, allowing users to easily identify grid issues.

Executive Summary

This white paper covers key technology details behind the WAMSTER Event triggering functionality, and describes some typical use cases for configuring and analyzing events:

- **Event triggering introduction:** details behind the triggering system, its internals and its limitations.
- **Defining processing rules:** choosing quantities, preprocessing rules and filters, and chaining them to get various system indicators
- **Configuring triggers:** defining relative or absolute comparisons, processing and thresholds for event triggering
- **Analyzing event details:** examining measurement trends related to triggered events and exporting data around the event for detailed offline analysis.



Wamster system implementation details



1 Event Triggering Overview

1.1 ROLE OF WAMSTER IN AD-HOC WAM SYSTEMS

Deregulation of markets and sustained growth of the need for electrical energy has resulted in an increase of competitiveness of electrical energy production, transmission and distribution environments. As a result, operating points of these systems are being shifted towards the stability margin, leading to the increasing complexity of control systems. Additionally, the dispersion and decentralization of production facilities that is occurring in efforts to increase the share of renewable energy sources has led to conditions that are very different from those that were in effect at the time of design and construction of existing power systems. As a consequence, negative interactions between individual, often geographically distant parts of the system, give rise to various problems, causes of which are very difficult to determine using conventional measurement methods which rely on disturbance detectors and PQ measurement devices.

WAM (wide area measurement) based systems, based on synchronized phasor measurements, provide an alternative solution. To provide measurements synchronized in time, each measurement device (a PMU; phasor measurement unit) is equipped with a source of accurate time, based on the GPS (global positioning system) signal. The uncertainty of this time-base is in order of a few microseconds, which in most cases has a negligible impact on the quality of the measurement results. Due to the small error, all the measured results can be considered as unambiguously defined in time. This facilitates tracking of the disturbances through the entire system covered by the network of PMU devices. Additionally, PMU devices measure the “absolute” phasor angle, allowing the immediate calculation of phasor angle differences at remote locations during transient conditions. This allows PMU based systems to provide much more accurate insight into the state of the system.

WAMSTER system takes a step further by simplifying the installation and deployment of the WAM network of PMU devices:

- Large stationary rack-mounted PMU devices are replaced with lightweight, portable, battery-backed PMU devices. Ethernet connection is replaced (if needed) with the GPRS/UMTS connection, allowing them to be deployed easily with no need for additional communication infrastructure.
- WAMSTER provides its own infrastructure for storage and analysis of the distributed measurements as a service, removing the need for installation and maintenance of dedicated concentrator equipment.
- WAMSTER web application provides remote device monitoring, real-time measurement visualization, comparison and analysis of measured data, event triggering configuration and data export. Web server makes the data accessible from any web-enabled device, with no need for additional plugins or local software installation.

These characteristics give WAMSTER a unique opportunity: to create ad-hoc WAM systems based on the technology of synchronized phasor measurements.

1.2 EVENT TRIGGERING IN WAMSTER

Common reasons for the creation of ad-hoc WAM system can be divided into:



Wamster system implementation details

- **Baselining survey measuring:** used to assess system state through the establishment of statistical indicators of key quantities in various operating modes of the system. An example of one such project is the collection of preliminary insights into system variations under normal operating regimes, in the design stage of advanced WAMPAC systems.
- **Troubleshooting measurement:** aiming to illuminate the conditions during problematic occurrences in the system. In doing so, the detection problem is usually performed on the basis of a hypothesis about the possible cause of the issue, which uses the measurement system to either confirm the hypothesis, rejected it, or indicate that the focus of the research should be redirected to a different area.

In both cases, it is necessary to process a large amount of generated measurement data. **Event triggering** system is the part of WAMSTER system which facilitates and accelerates the arrival of new information about the dynamic changes in the system through continuous data processing and user parameterized processing algorithms.

1.3 HIGH LEVEL OVERVIEW OF THE WAMSTER SYSTEM OPERATION

PMU devices are connected to the WAMSTER server by means of data packet communication (GPRS) over the mobile phone network (GSM) in most ad-hoc formed WAM systems. Synchrophasor data is transmitted from remote units (PMU) to the WAMSTER server at speed of 10 synchrophasor frames per seconds (FPS). In this default mode, a packet containing 4 voltage and 4 current synchrophasors is sent to the server each 100ms. If Ethernet connection is used instead of GPRS link, the default reporting rate may be increased to synchronous reporting speed (50 or 60 FPS). Default GPRS reporting rate may be decreased in several steps (50-25-10-5-2 or 30-15-12-10-5-2) to a minimum reporting rate 1 FPS by user, or adjusted automatically by the WAMSTER server to adapt the necessary bandwidth according to communication channel capability degradation.

When used, GPRS link occasionally introduces omission of transmitted data packets. Frames which are skipped or corrupted due to low network conditions are automatically recollected from the device's SD memory as soon as the sufficient data flow is restored. It usually happens in few seconds, although larger gaps may occur when the GPRS link goes offline for several minutes. In the latter case, all missing data is collected in the same manner when device reconnects. This can take more time to resolve depending on network conditions and reporting rate (i.e. packet size), taking several minutes in extremely low network conditions, but all data is always recollected from remote units to the server.

1.4 HOW DOES EVENT TRIGGERING WORK?

Event triggering system is a component which continuously runs and analyzes synchrophasor measurements according to a set of user defined (parameterized) processing rules. Whenever a processed quantity or a comparison of multiple quantities exceeds a defined threshold value, an event is generated and stored in the database.

For devices operating at reporting speeds lower than the full sync resolution (50/60 Hz), whenever an event is detected, Wamster sends requests to all referenced PMU devices to collect surrounding synchrophasor data (5 seconds before and after the event) at full sync resolution (50 or 60 Hz, depending on the grid frequency). This action simplifies and accelerates later event analysis for the end user. These requests are added to a lower priority queue, to prevent interfering with real-time measurements in low network conditions. The interaction between the Wamster core collecting system and the Event triggering subsystem is presented in the Fig. 1.1.



Wamster system implementation details

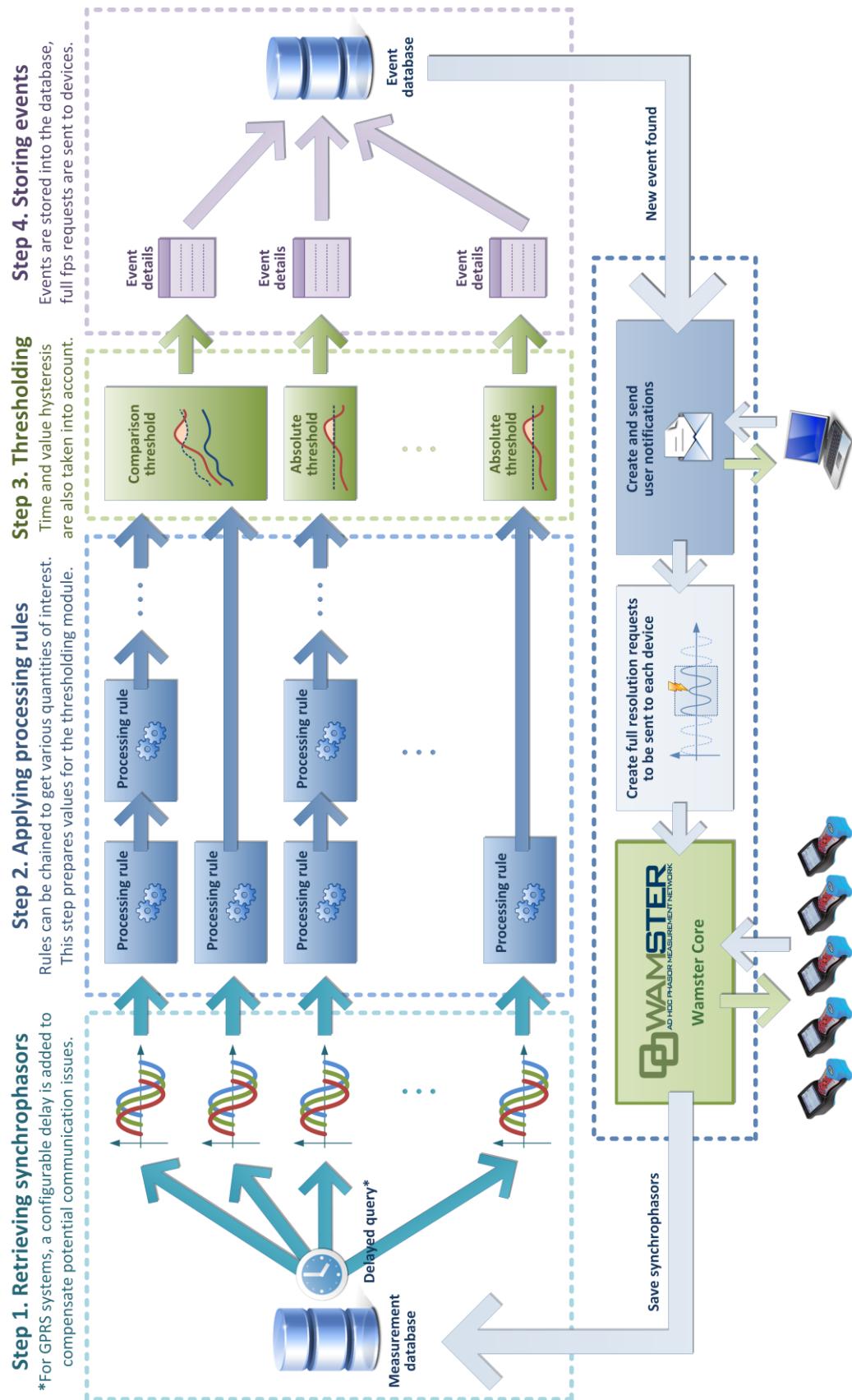


Figure 1.1: Wamster Event triggering system overview



2 Creation of input synchrophasor data streams

Event detection in Wamster is revolved around *triggering agents*. The purpose of each triggering agent is a creation of a separate synchrophasor measurement data stream to the Event triggering subsystem. Wamster can execute many triggering agents simultaneously, and each of them can work in either offline or online mode, depending on the source of the stream data.

Offline mode triggering agents are started on user request and analyze a specified historical interval. These triggering agents retrieve synchrophasor data from the database and create one or more synchrophasor data streams feeding the triggering subsystem inputs. As soon as the data for the requested interval is processed, the offline agent shuts down.

Recently received synchrophasor data is constantly processed by the online triggering agents. These agents are active through the entire lifetime of the Wamster system, and work continuously and in parallel with the Wamster core. In this case, recently received synchrophasor data is constantly retrieved from database and fed to the Event triggering subsystem.

To compensate a problem with occasionally missing frames (before they are automatically recollected), a delay is inserted by each online agent before fetching the raw data from the database. By default, data flow is delayed for 5 minutes. This has a consequence that all events are detected several minutes after they have actually happened.

Time resolution of the synchrophasor data stream fed to triggering subsystem equals to reporting rate speed of data in database. Changes in time resolution do not influence the triggering procedure – all calculations and comparisons adjust to variation in sampling rates on same channel or difference on two independent channels.



The delay introduced by online triggering agents restricts the usage of an ad-hoc WAMSTER network to “open-loop” or “post-mortem” applications. For closed loop tasks, a dedicated WAMSTER network based on reliable Ethernet or serial links must be formed; in that case, data flow for triggering is not generated from the data stored in the database, but instead retrieved directly by the WAMSTER communication core as soon as it is received from the devices.

Event triggering in closed loop applications exceeds the scope of this document.



3 Processing rules

Before defining thresholds for a trigger, Wamster needs to get the exact specification of the value which needs to be monitored. Using the online web interface, users can create and edit these specifications (called **Processing rules**), as described in the following chapters.

3.1 DEFINING A PROCESSING RULE FOR A SPECIFIC QUANTITY

Processing rules to be defined using the following properties¹:

- **Quantity:** Voltage/Current/Power
 - Allows selecting quantities for processing
- **Vector type:** Phase (+ phase number)/Positive symmetry/Total
 - For Voltage and Current, it allows user to choose between a single phase measurement and the positive symmetry component. For Power quantities, Total can be selected to monitor the total power.
- **Phasor property:** Magnitude/Angle/Frequency
 - For scalar values (P/Q), this value is ignored.
- **Processing source:** Direct/Parent value
 - Defines whether the rule uses a database (direct) quantity (selected using properties above), or is chained to an existing (parent) processing rule
- **Post processing:** Actual value/Low pass filter/Difference from low pass/Rate of change
 - Actual value: performs no post processing
 - Low pass filter: applies a PT1 low-pass filter using a user defined time constant
 - Difference from low pass: returns the difference between the actual value, and the filtered (low pass) value.
 - Rate of change: returns the rate of change (delta) between two samples.

First four parameters define monitoring value. Post processing property defines optional filtering technique applied to monitoring value. Detailed schematic describing above procedures is presented on figures 3.1 and 3.2.

To allow chaining of post processing rules, value source can be set as a Direct value (actual measurement retrieved from the database), or a Parent value (processed value from another defined rule). For processing rules which don't involve chained post processing, this settings will be set to Direct.

3.2 CHAINING PROCESSING RULES

Several processing rules can be chained. In that case, only the resulting value of the previously defined and selected (parent) rule is passed to the post processing filter, while other properties defined in the successor rule are ignored. The purpose of rule chaining is achieving complex post processing configurations that allow more diversity in disturbance detection by threshold algorithms.

¹ List of available properties as of October 2012. The set is continuously expanded upon user requests.



Wamster system implementation details

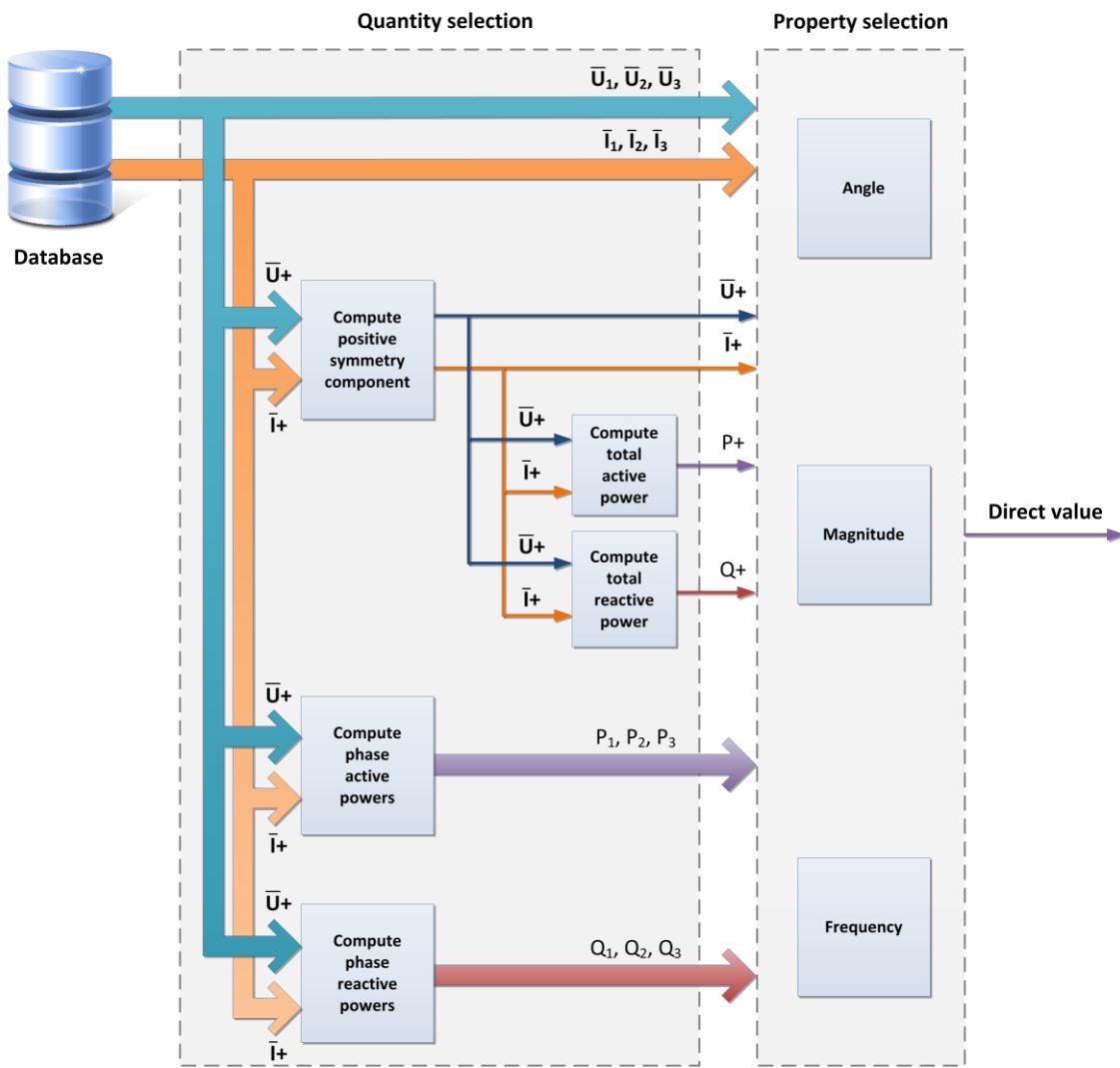


Figure 3.1: Quantity and property selection

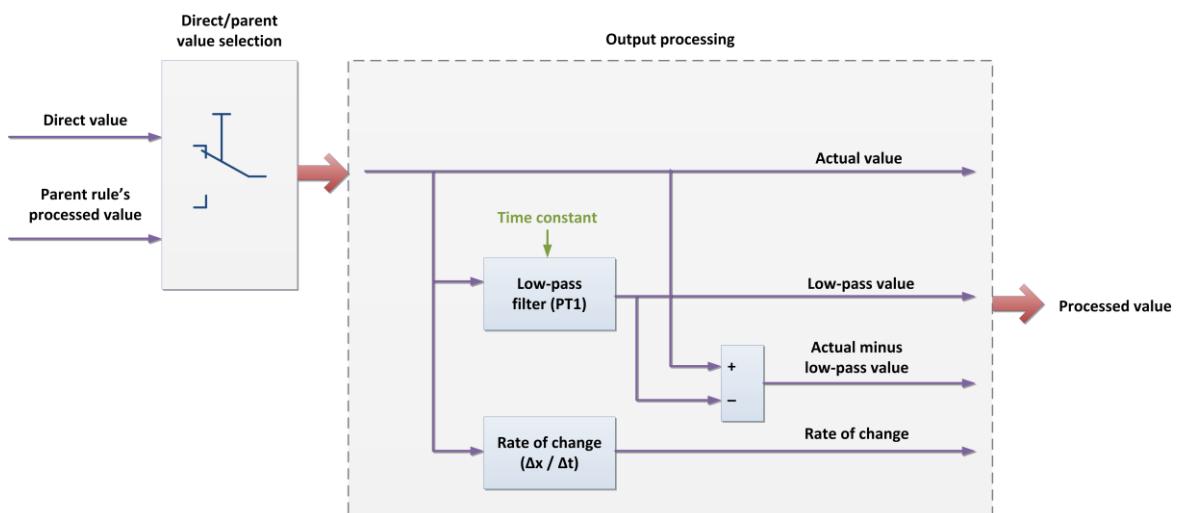


Figure 3.2: Output processing calculation



4 Creating event triggers

Resulting processed monitoring values serve as input values for event detection algorithms, where they are compared with preset thresholds. Event triggers can either be absolute or relative, meaning they either observe a single value, or compare two values defined by two different processing rules. When a particular threshold conditions are met, event is fired and appropriate action are initiated.

Event trigger subsystem is configured by the user, by adjusting the following properties:

- **Type:** Absolute/Relative
 - Absolute trigger monitor values generated by a single processing rule. Relative trigger monitors results of two processing rules and act upon the difference between two values.
- **Source:** One or two processing rules included in threshold.
- **Comparison:** Actual/Low pass/ Difference from low pass
 - after the value has been retrieved from the processing rule, triggering mechanism can itself perform additional post processing before checking the threshold. For relative triggers, post processing is applied after the values have been subtracted.
- **Expected value**
 - the nominal, expected value for quantity being monitoring. The unit defined by the processing rule applies.
- **Error threshold**
 - the minimal absolute difference between current monitoring value and expected value that fires the trigger. The unit defined by the processing rule applies.
- **Value hysteresis**
 - defines the hysteresis for closing (ending) the event expressed in the percentage of the threshold. A dead-zone of a hysteresis prevents multiple event firings when monitoring value remains near the error threshold.
- **Duration hysteresis**
 - This value defines the minimum time the value must be within threshold limits in order to be considered that particular event state has been finished. As long as the value oscillates around the threshold in shorter intervals, the event will not be finished.
- **Threshold direction:** Dip/Swell/Both
 - Direction of the threshold defines the sign of the difference between the actual and expected value. If the difference is positive the event is a **swell**. If the difference is negative, then the event is a **dip**. This property defines whether Wamster should detect dips and swells only, or if both directions should trigger the event.
 - To define different thresholds for dips and swells of the same value, two event triggers must be defined, one for each threshold direction.



5 Using the online interface

List of triggered events for a specific user can be found at the **Events** page in the Advanced Wamster menu. User needs to be logged-in in order to access this page.

Default view shows the list of detected (triggered) events and provides the interface for configuring event triggers for PMU devices.

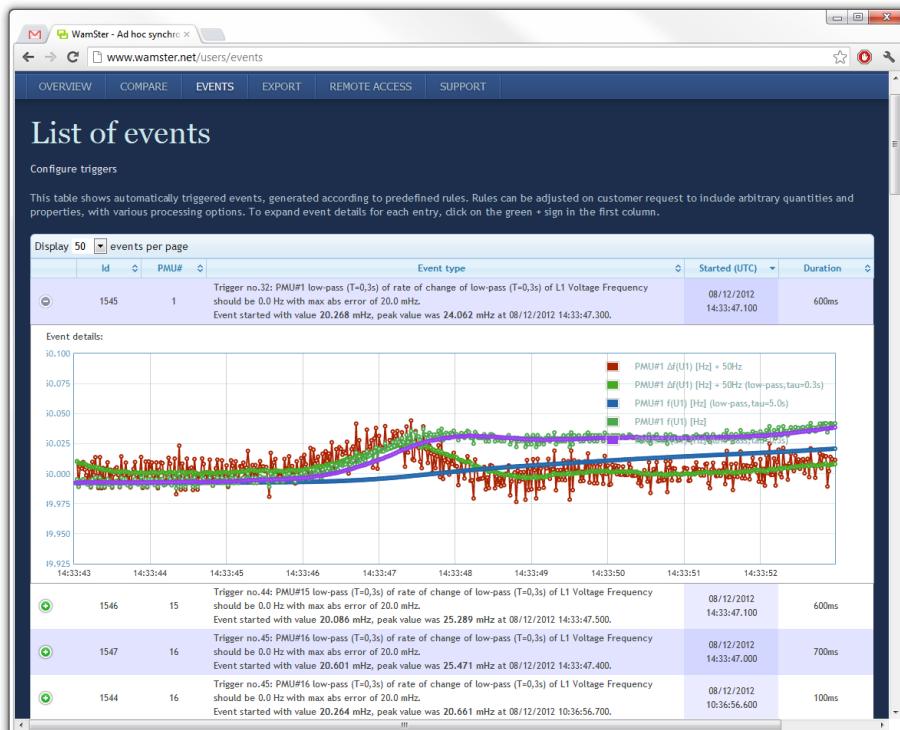


Figure 5.1: List of events. Clicking the green plus-sign button, expands details for the event.

For each entry in the table, user can view the snapshot of the quantities which triggered the event by clicking on the green Event details button at the beginning of the row. Displayed chart also allows zooming and panning.

Event table can be sorted by clicking on individual column headers, as well as filtered using text inputs shown at the bottom of the table for each column.

By clicking the "Configure triggers" link, **Event triggering setup** page is displayed:



Wamster system implementation details

Id	Source	Parent value	Pmu#	Quantity	Vector Type	Property	Phase Number	Processing	Filter time constant [s]	Description	Edit/Delete
1	Direct	None (direct)	1	Voltage	Phase	Angle	1	Actual	N/A	STER angle	Edit - Delete
9	Direct	None (direct)	1	Voltage	Pos. symmetry	Magnitude	1	Actual	N/A	STER abs.voltage	Edit - Delete
19	Direct	None (direct)	1	Voltage	Pos. symmetry	Angle	1	Actual	N/A	STER pos.sym.angle	Edit - Delete
37	Direct	None (direct)	1	Voltage	Phase	Frequency	1	Low pass	5	STER freq,low-pass(tau=5)	Edit - Delete
38	Direct	None (direct)	1	Voltage	Phase	Frequency	1	Low pass	0.3	STER freq,low-pass(tau=0.3)	Edit - Delete
45	Parent value	Id: 38	1	Voltage	Phase	Frequency	1	Delta (rate of change)	N/A	STER diff freq	Edit - Delete
46	Parent value	Id: 45	1	Voltage	Phase	Frequency	1	Low pass	0.3	STER low pass of diff freq	Edit - Delete
53	Direct	None (direct)	15	Voltage	Pos. symmetry	Magnitude	1	Actual	N/A	HE Čakovec abs. voltage	Edit - Delete
54	Direct	None (direct)	16	Voltage	Pos. symmetry	Magnitude	1	Actual	N/A	TS Žerjavinec abs. voltage	Edit - Delete

Figure 5.2: Interface for configuring event triggering rules and triggers.

Table for defining event triggers is divided into two tabs:

- **Processing rules:** used to define processing rules as described in the previous chapter, and
- **Event triggers:** creating actual event triggers.

Both rules and triggers can be edited by clicking on the red **Edit** link at the end of each row. **Delete** link allows the user to delete the rule.



6 Example: creating processing rules

For simpler event triggering scenarios, it is enough to create a single processing rule and a single trigger rule. For example, to fire an event when the **rate of change of the U2 frequency exceeds a certain value**, user will choose the **Direct** source (no chaining), **Voltage** as quantity, **Phase** as vector type, **Frequency** as the property and **Rate of change** for value processing, and then continue to define triggers for this value.

In a different scenario, user might want to fire an event when the **low-pass filtered value of the rate of change** of the positive symmetry exceeds a threshold. Since **low-pass filter** and **rate of change** are two different processing filters, two rules need to be chained. First rule in the chain will have a direct (database) source and configure the quantity and property which needs to be extracted, along with the first post processing filter (rate of change), while the second one needs to use the previous rule as the source, and the define the next processing filter:

1. **First rule (ID = 1)**
 - a. Source: **Direct (database)**
 - b. Quantity: **Voltage**
 - c. Vector type: **Positive symmetry**
 - d. Phasor property: **Frequency**
 - e. Post processing: **Rate of change**
2. **Second rule (ID = 2)**
 - a. Source: **Parent value (First rule, ID = 1)**
 - b. Post processing: **Low pass filter**
 - c. Time constant: **1 s**

Both of these rules can be used independently in triggers. First rule calculation is performed on the direct database value to get the rate of change of the frequency, and the second rule takes the result of the first calculation and passes it through a low pass filter, but the first rule can serve as the source for a trigger nevertheless.

The first step in creating actual **event triggers** is to define their **processing rules**. Unless a processing rule is referenced by a trigger (either directly, or indirectly through a chained rule), that processing rule will not be evaluated at all, in order to avoid consuming server resources.

