

synchro

Synchro Software User Guide

Version: 2.2
Release Date: 2012-09-08

Revision History

Version	Date	Description
1.2.9	2012-09-07	Documentation update.
2.2.7	2012-10-15	Documentation update.

Table of Contents

1	Introduction	1
1.1	What is Synchro?.....	1
1.2	What Synchro Does with Your Data?	1
1.3	Why Use Synchro?	2
1.4	Architecture Overview.....	3
1.5	Synchro Versions.....	3
2	SynchroWeb Interface	4
2.1	Introduction	4
2.1.1	Views	4
2.1.2	Selecting Metrics.....	4
2.1.3	Selecting Time	5
2.1.4	Selecting Statistics.....	6
2.2	Basic Controls.....	8
2.3	Window Controls.....	8
2.4	Data Controls.....	9
2.5	View Controls.....	10
2.6	Formats Description.....	11
2.6.1	PeriodStatistics Format.....	11
2.6.2	ByTimeStatistics Format.....	16
2.6.3	Miscellaneous Formats	19
3	Synchro Administration	21
3.1	SynchroServer Installation	21
3.1.1	Pre-Requisite	21
3.2	Installation Instructions	21
3.3	SynchroServer Start/Stop Procedures.....	22
3.4	SynchroClient Solaris Installation	22
3.5	SynchroClient Linux Installation.....	23
3.6	SynchroClient Windows Installation.....	25
4	SynchroServer Configuration.....	26
4.1	Basic Admin Controls	26
4.2	Action Menu	26
4.3	SynchroServer Configuration.....	27
4.3.1	Server - Configuration.....	27
4.3.2	Server – Retention	27
4.3.3	Server – Compression	28
4.3.4	Server – Aggregator.....	28
4.3.5	Server – Logging.....	28
4.3.6	Logs and Alarm Parameters	28
4.3.7	Server - Internal Parameters.....	29
4.4	Alarm Configuration	29
4.5	MAVG Configuration	30
4.6	Color Configuration	31
4.7	Users Configuration	31
4.8	Periods Configuration	31
4.9	Aggregator Configuration.....	32
4.10	Compression Configuration.....	32
4.11	Retention Configuration.....	33
4.12	Client Configuration.....	33
4.13	Dropfrom Configuration	33

synchro

Table of Contents

4.14	Memory Trigger Exception Configuration.....	34
4.15	Time Series Mapping Configuration.....	34
4.16	Value Mapping Configuration.....	34
4.17	Presets Configuration.....	34
Appendix A.	SynchroClients.....	36
A.1	SynchroLinux.....	36
A.2	SynchroWindows.....	36
Appendix B.	Application Integration.....	37
B.1	Trivial TCP Text Protocol.....	37
B.2	SynchroTail.....	39
B.3	Integration Examples.....	40
Appendix C.	SynchroServer Performance Tuning.....	47
C.1	Increase Disk Performance.....	47
C.2	Increase CPU Usage.....	47
C.3	Increase Memory Available.....	48

synchro

1 Introduction

This document provides the information required to use the Synchro software. Discrepancies in this documentation or bugs in the Synchro software should be reported to support@affatechnology.com (more information available at www.synchro-tech.com).

Thank you for using Synchro.

1.1 What is Synchro?

Synchro is a real time enterprise infrastructure-monitoring tool that acquires and manages data and events from multiple sources throughout the enterprise. Synchro collects, aggregates, historizes, alerts, visualises, and reports on your data.

Think of Synchro as a spreadsheet, only field values in Synchro can change every second, or if necessary, on a more frequent basis. Like a spreadsheet, Synchro is not aware of the meaning of the data it processes.

A field in Synchro, and the associated data that changes every second is called a **Time Series**. A typical Synchro infrastructure processes hundreds of thousands of these Time Series updates every second.

1.2 What Synchro Does with Your Data?

For each Time Series, Synchro provides two (2) types of statistics.

- ByTimeStatistics
- PeriodStatistics

For each second of the day, within each Time Series, Synchro provides the latest value (LAST), Min, Max, Average, Count, and Sum, of the data received for that second. These second level statistics are called **ByTimeStatistics**.

Additionally, in Synchro it is possible to define **Periods**. A Period describes a continuous interval for a time of the day. For each Period, and for each Time Series, Synchro provides the latest value (LAST), Min, Max, Average, Count, Sum, Time (time of the last update), and Dtime (time since the last update). These Period based statistics are called **PeriodStatistics**. By default, Synchro keeps hourly, morning, evening, work hours, and daily PeriodStatistics. Users can define as many (possibly overlapping), Periods as required.

Synchro can also “artificially” create Time Series based on your data. It can group them together, aggregate days of data together, create new Time Series using moving aggregation functions (like moving averages), and create alerts that they themselves, become Time Series.

Synchro uses four (4) identifiers to uniquely identify a Time Series. By convention, the identifiers, **Types** and **Metrics** define what is being measured, and the identifiers, **Components** and **Targets** identify what it is measured on.

One of the most useful and distinctive feature of Synchro is the possibility of using values of PeriodStatistics to select a Time Series (in addition to the name of a Time Series). This makes it easy and efficient to display a real-time list (updated every second), of the top “N” clients/products/systems, based on any Time Series.

synchro

Synchro also offers the possibility of comparing missing/added items between dates and periods of the day, making it a simple task to compare the configuration of systems to see what was changed.

1.3 Why Use Synchro?

Synchro is an outstanding monitoring and capacity-planning tool that has been successfully used in diverse contexts such as industrial infrastructure, technology infrastructure, and application monitoring.

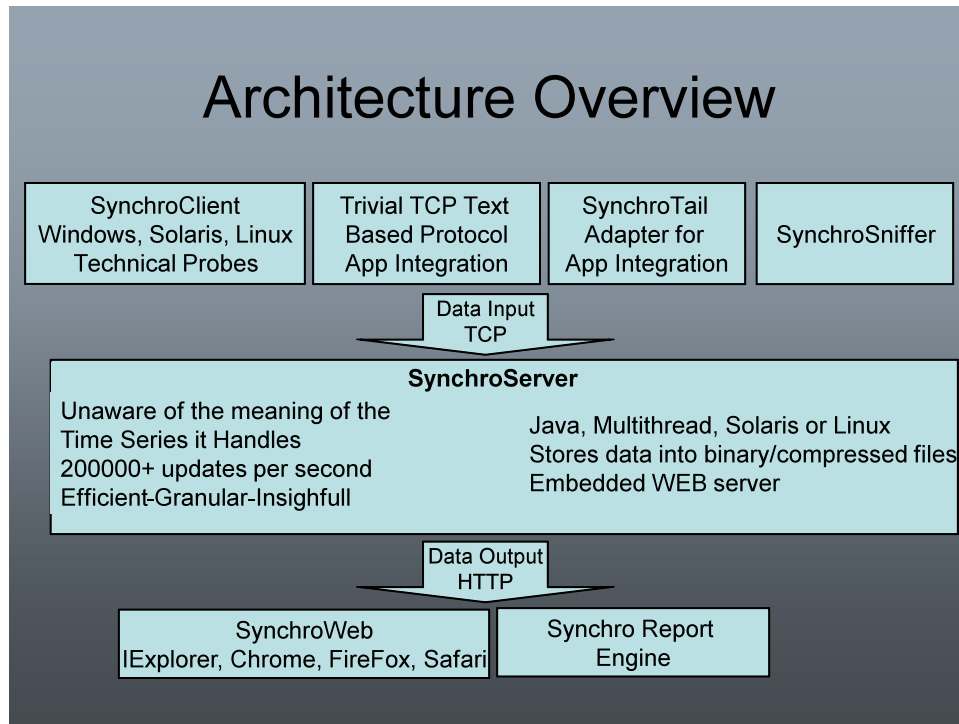
Synchro is...

- **Efficient:** Each Synchro infrastructure can process a sustained load of more than 200,000 data updates per second on commodity server hardware.
- **Granular:** By default, all data in Synchro is kept at the second level precision. Data can be historized and efficiently at the second level precision for years if required.
- **Insightful:** By pro-actively re-calculating masses of statistics every second, and allowing for aggregation and smoothing of data, Synchro provides real-time answers to questions such as:
 - Did something change?
 - What was changed?
 - What was added?
 - What is missing?
 - What are the impacts of the change?

synchro

1.4 Architecture Overview

The SynchroServer, a multi-threaded JAVA application, is at the heart of the Synchro infrastructure. It receives data from probes, applications, and adapters to aggregate, historize, report, and create alarms as needed. The data is sent to the SynchroServer via a trivial text TCP protocol. The SynchroWEB and SynchroAdmin tools are used within a WEB browser to interact the SynchroServer embedded WEB server via the HTTP protocol.



1.5 Synchro Versions

There are two distinct versions of Synchro. There is the full Commercial version described in these pages, and the Demonstration version which is valid for three months. The following table outlines the differences between the versions.

For more information on how to acquire the Commercial version of Synchro, please contact sales@affatechnology.com.

Characteristics	Commercial Version	Demonstration Version
Support	Included	Not Included
Historical Data	Included	Not Included
Alarms	Included	Not Included
Reports	Included	Not Included
All Others	Included	Included

synchro

2 SynchroWeb Interface

This section presents the interface used in SynchroWEB.

2.1 Introduction

The SynchroWEB interface is the main tool that users will utilize to interact with the SynchroServer. Users can connect and use the SynchroWeb interface from the following URL by using a web browser such as; Safari, Chrome, or Firefox:


<http://ServerName:8080/Synchro/Synchro.html>

2.1.1 Views

The SynchroWEB interface is built around the **Views** concept. Users can create Views that are composed of any number of Data Windows. Data Windows present a set of data in a diverse format (e.g., graphs, text, tables...). Views can be protected (read only) and, depending on the SynchroServer mode, can also be hidden to other users.

2.1.2 Selecting Metrics

To select which *Time Series* are to be reported on, Synchro uses PeriodStatistics.

The Search tool () , available in the Data Window/Data Controls, is used to search for Time Series. Using this tool, users can select dates, periods, and a set of Time Series to be displayed. The identifiers *Component/Target* and *Type/Metrics* are used to select Time Series based on their name. Using the sliders; Click, Ctrl-Click, and Alt-Click, users can select the Time Series to be reported on. Additionally, users can also enter a string in the overhead fields to match a set of Time Series. Strings are composed of regular expressions. Using this functionality, users can select many Time Series by entering only one string.

For example: The string “System[12345]System[1.*1]” would match System 1 through 5 and System10A1.

The table below describes the basic functionalities of regular expressions.




Character	Signification
. (dot)	Matches any single character except line break characters \r and \n.
* (star)	Repeats the previous item zero or more times.
+ (plus)	Repeats the previous item once or more
\Q...\E	Matches the characters between \Q and \E literally
^ (caret)	Matches the start of the string the regex pattern is applied to.
\$ (dollar)	Matches the end of the string the regex pattern is applied to.
[123]	Defines inside a character class.(Matches Character 1, 2 or 3).
(pipe)	Causes the regex engine to match either the part on the left side, or the part on the right side.

Note: More information on “regular expressions can” be found by searching the Internet.


synchro



Using the above selection scheme, it is possible to simultaneously select thousands of Time Series.


The following parameters are also used to refine which Time Series, and on what, will be reported.

The Function menu () used with the Sort Criteria () , and the Options () /"Data and Range"/"Maximum Number of Time Series" are used to ultimately select which Time Series are to be reported on.

2.1.3 Selecting Time

The Search tool () enables the user to select which date(s) and period(s) are to be used for Time Based and Time Series selection.

The Time Sample () is used to select the size of the *Sliding Window* that ranges from the current server time, to a specified duration in the past (in seconds). This functionality is also available in Options ()/"Data and Range"/"Time Sample".

Users can also select a specific interval of time using Options ()/"Data and Range"/"ZoomFrom", and "Data and Range"/"ZoomTo" (Format "HHMMSS").


Dates can be selected using the format YYYYMMDD in the Search tool. Dates and Periods can also be relatively defined using the current SynchroServer *Date/Period*. Below are a few examples:

Function	Description
Today	D-00
Yesterday	D-01
117 Days ago	D-117
Last Business Day	W-01
Today and Yesterday	D-00 D-01
This Week Mon-Sun	WK-00
Last Week Mon-Sun	WW-00
Last Week Mon-Fri	WW-01
This Month Mon-Sun	MO-00
This Month Mon-Fri	MO-01
Last Month Mon-Fri	MN-01
ThisHour	PH-00
LastHour	PH-01

synchro

2.1.4 Selecting Statistics

For each Time Series Synchro stores two (2) types of statistics. There are specific Synchro Formats designed to report on the different statistics.

Note: When using statistics pay attention to the Function menu () as this menu selects which aggregation function is to be used for reporting.

The **PeriodStatistics** are designed to provide an overview over a Period of time of the detailed second by second statistics. Users can define as many Periods as required. By default, Synchro provides several Periods as presented in the following table.

SynchroServer: Synchro-server1.synchro-tech.com								
Component=frean01 Type=HOST Metric=CPU Target=HOST Date=2012-08-31 Sort=Metric								
	Time	DTime	Last	Max	Average	Min	Sum	Count
00@08	07:59:59		1.45	37.97	2.77	0.66	79785.39	28799
00@24	23:59:59		1.74	37.97	2.36	0.63	204496.49	86399
08@16	15:59:59		1.30	27.71	2.19	0.71	63079.80	28800
16@24	23:59:59		1.74	30.33	2.13	0.63	61631.30	28800
H00	00:59:59		1.81	24.11	2.11	0.77	7629.08	3600
H01	01:59:59		1.25	17.10	2.22	0.83	8000.34	3599
H02	02:59:59		1.53	17.15	2.31	0.85	8332.27	3600
H03	03:59:59		2.34	14.49	2.30	0.82	8288.05	3600
H04	04:59:59		3.11	17.93	3.96	1.06	14280.96	3600
H05	05:59:59		2.66	37.97	4.97	0.87	17893.11	3600
H06	06:59:59		2.22	20.13	2.14	0.66	7709.26	3600
H07	07:59:59		1.45	22.01	2.12	0.87	7652.32	3600
H08	08:59:59		1.64	23.92	2.14	0.75	7707.29	3600
H09	09:59:59		1.51	23.91	2.13	0.80	7703.07	3600
H10	10:59:59		2.44	17.35	2.18	0.85	7865.24	3600
H11	11:59:59		1.92	20.64	2.17	0.84	7818.41	3600
H12	12:59:59		2.15	16.69	2.18	0.86	7864.86	3600
H13	13:59:59		1.57	20.41	2.13	0.80	7680.06	3600
H14	14:59:59		1.91	21.88	2.26	0.71	8170.63	3600
H15	15:59:59		1.30	27.71	2.29	0.77	8270.24	3600
H16	16:59:59		1.67	20.77	2.30	0.89	8299.99	3600
H17	17:59:59		1.53	16.46	2.11	0.84	7598.72	3600
H18	18:59:59		2.19	16.98	2.12	0.72	7647.56	3600
H19	19:59:59		1.73	22.84	2.09	0.78	7553.65	3600
H20	20:59:59		1.65	30.33	2.12	0.63	7648.46	3600
H21	21:59:59		2.56	24.37	2.10	0.77	7593.87	3600
H22	22:59:59		1.73	23.63	2.15	0.80	7749.32	3600
H23	23:59:59		1.74	24.44	2.09	0.92	7539.73	3600

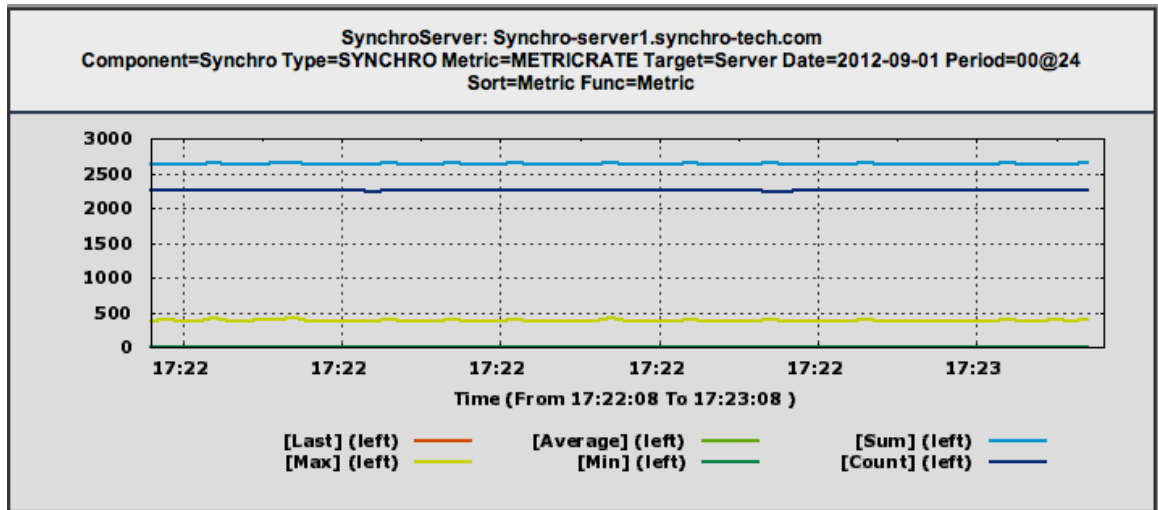
For each second of the day, and for each Time Series, Synchro provides the Last (latest value), Min, Max, Average, Count, and Sum of the data received. These second level statistics are called **ByTimeStatistics**, and are designed to provide the level of detail required to drill down into events that require more attention.

The following table demonstrates that for each second, the SynchroServer received ~2246 updates (count) for this particular Time Series. Other statistics for the data received are also available.

synchro

SynchroServer: Synchro-server1.synchro-tech.com
 Component=Synchro Type=SYNCHRO Metric=METRICRATE Target=Server Date=2012-09-01 Period=00@24
 Sort=Metric





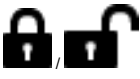
Time	Synchro					
	Last	Max	Average	Min	Sum	Count
17:22:07	1.00	408.00	1.18	1.00	2653.00	2246
17:22:06	1.00	408.00	1.18	1.00	2653.00	2246
17:22:05	1.00	382.00	1.16	1.00	2627.00	2246
17:22:04	1.00	382.00	1.16	1.00	2627.00	2246
17:22:03	1.00	382.00	1.16	1.00	2627.00	2246
17:22:02	1.00	397.00	1.17	1.00	2642.00	2246
17:22:01	1.00	382.00	1.16	1.00	2627.00	2246
17:22:00	1.00	382.00	1.16	1.00	2627.00	2246
17:21:59	1.00	382.00	1.16	1.00	2627.00	2246
17:21:58	1.00	382.00	1.16	1.00	2627.00	2246
17:21:57	1.00	408.00	1.18	1.00	2653.00	2246
17:21:56	1.00	382.00	1.16	1.00	2624.00	2243
17:21:55	1.00	382.00	1.16	1.00	2626.00	2245
17:21:54	1.00	382.00	1.16	1.00	2626.00	2245



synchro

2.2 Basic Controls

The following basic controls are available in the upper portion of the SynchroWEB interface.







Function Icon	Description
	The Home View icon is used to select the “home” view. The user uses Synchro to create the Home view (view name is home). If Home view is not defined, the “default” View is used.
	The Views icon is used to access the <i>View Selector</i> interface.
	The Report Icon is used to activate the <i>Report Selector</i> tool to view already created reports.
	View Controls provide basic view controls such as; Save, Save As, and Delete for the views. This is also the menu used to access the <i>SynchroAdmin</i> interface.
	The Lock/UnLock icons show the status of the View. Only the owner of the View can lock or unlock the View.

In addition, Synchro displays the View name, and the name of the current user in the upper right portion of the screen.



2.3 Window Controls

Window Controls are available when the mouse pointer hovers on the upper right portion of any Data Window.

Note: The View needs to be Unlocked () for the controls to appear.


Function Icon	Description
	Provides information about the status of the query sent to the server by the <i>SynchroWEB</i> interface.
	Pause and Restart the update of the Data Window.
	Duplicate (copy) the Data Window.
	Export the content of the Data Window. Export default format is MS Excel (xls). Format can be changed in “View Controls ()”/”View Options”/”ExportFormat”
	Close Data Window. Note – users cannot delete all Data Windows – one Data Window must remain open.









synchro

Function Icon	Description
	Manually Refresh the content of the Data Window when Paused.
	Expand/Collapse the Data Window.

2.4 Data Controls

Data Controls are available when the mouse pointer hovers on the upper left portion of any Data Window (below the **Title** region).

Note: The View needs to be Unlocked () for the controls to appear.

Function Icon	Description
	The Search tool is used to select the <i>Time Series</i> to be reported on. In addition, sort criteria and the Options / “Data and Range”/“Maximum Number of Time Series” when used together, select which Time Series are to be reported on.
	The Option tool provides access to the “Window Control”, “Data and Range”, and “Format Specific” options.
	The Time Sample is used to select the size of the Sliding Window that ranges from the current server time to the specified duration in the past (in seconds). This functionality is also available with Options/“Data and Range”/“Time Sample”. The values available in this menu can be changed in the SynchroAdmin interface.
	Selects the Periods be used for data selection (<i>PeriodStatistics</i>) and for time selection. This functionality is also available in the Search tool. The values available in this menu can be changed in the SynchroAdmin interface.
	Selects the Dates to be used for data selection. This functionality is also available in the Search tool. The values available in this menu can be changed in the SynchroAdmin interface.
	Selects the Format of the data to be displayed. See Formats description below for details.
	The Sort Criteria is used to select which Time Series are to be displayed, and also establishes the order in which they are to be displayed. A reverse sort is also available in Options/“Data and Range”/“Reverse Sort”.
	The Function tool is used to select which aggregation function to display. This functionality is also available in Options/“Data and Range”/“Function”. Possible values are LAST, MIN, MAX, AVG, COUNT, and SUM


synchro

2.5 View Controls

View Controls are available in the upper right portion of the screen under the (⚙️) icon. The following menu appears when the mouse pointer cursor hovers over the icon.

Function Icon	Description
Share...	Enables the user to create an URL where all Basic Controls are removed. This is useful to provide a simple “read only” interface for specific users.
Save...	Save the current View
Save As...	Save the current View using a new name.
Delete	Delete current View.
View Options	View Options present some controls over the View. More information is available below.
SynchroAdmin	This menu item launches the <i>SynchroAdmin</i> interface. This interface is the tool the Synchro Server Administrator uses to configure the Synchro Server.
Logout	Logout logs the user out. Login information is requested as needed when a user attempts to access a functionality that requires login information.

The View Options menu presents the following controls over the View:

Function Icon	Description
Export Format	Selects which report format is to be used to export the data in “Window Controls”/  . The “Query URL Only” export format is used to export a query that can be used in another application such as; a spreadsheet, a browser, another SynchroWEB interface, or wget.
View Background Color	Sets the View Background Color in a format acceptable in HTML: i.e.: red, green, blue, #RRGGBB, rgb(0,0,0),and rgba(0,0,0,0))
BackGround Image URL	Set the background image. Example: http://SynchroServerName/download/Diapositive1.png
Maximum Items Retrieved in a Search	Sets the maximum number of Time Series that can be extracted in a single search (default 1000).
Snap to Grid	Snap to Grid is useful when placing windows in a View.
Force Login	Forces the user to login to see a View.
Size Style	Size of a View can be either “Fixed Size”, where the view size is independent to the size of the browser screen, or “Resize to Fit” (default), where the View size and included items resize

synchro

Function Icon	Description
	automatically according the screen size.
View Width/Height	When “Fixed Size” <i>Size Style</i> is selected, these parameters determine the size of the View (unit pixels).

2.6 Formats Description

There are three (3) types of formats in Synchro. Some formats are designed to report on **PeriodStatistics**, some on **ByTimeStatistics** and some others are designed to provide complementary information to Views.

Note: Unless stated, the following examples relate to a single Time Series.

The SYNCHRO/METRICRATE (TYPE/METRIC) below, describes the number of individual Time Series that updated the SynchroServer each second.

Notice that a particular date and period (H08 – from 08:00:00 to 08:59:59) was selected.

Search 1
Search 2
Search 3

Date	Component (1)	Target (1)	Type (1)	Metric (1)																																																	
<div style="border: 1px solid gray; padding: 2px; margin-bottom: 5px;"> <div style="background-color: #333; color: white; padding: 2px; text-align: center;"> « Aug 2012 » </div> <table style="font-size: 8px; text-align: center; width: 100%;"> <tr> <td>S</td><td>M</td><td>T</td><td>W</td><td>T</td><td>F</td><td>S</td> </tr> <tr> <td>29</td><td>30</td><td>31</td><td>1</td><td>2</td><td>3</td><td>4</td> </tr> <tr> <td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td> </tr> <tr> <td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td> </tr> <tr> <td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td> </tr> <tr> <td>26</td><td>27</td><td>28</td><td>29</td><td style="background-color: #90EE90;">30</td><td style="border: 1px solid blue;">31</td><td>1</td> </tr> <tr> <td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td> </tr> </table> <div style="margin-top: 5px;"> Period H08 <input style="width: 100%;" type="text"/> </div> </div>	S	M	T	W	T	F	S	29	30	31	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1	2	3	4	5	6	7	8	Synchro	Server	SYNCHRO	METRICRATE
S	M	T	W	T	F	S																																															
29	30	31	1	2	3	4																																															
5	6	7	8	9	10	11																																															
12	13	14	15	16	17	18																																															
19	20	21	22	23	24	25																																															
26	27	28	29	30	31	1																																															
2	3	4	5	6	7	8																																															

1(0/0)

Results
Results 1
Results 2
Results 3
Debug
Query
Sort by ↓

	Time	DTime	Last	Max	Average	Min	Sum	Count
Synchro	08:59:59		1.00	418.00	1.17	1.00	9482971.00	8084986

2.6.1 PeriodStatistics Format

The following formats present PeriodStatistics in diverse form; some of the most important features of each format are presented.


Statistics (HTML)

Offers a view of the PeriodStatistics for a particular period.

synchro

SynchroServer: Synchro-server1.synchro-tech.com						
Component=Synchro Type=SYNCHRO Metric=METRICRATE Target=Server Date=2012-08-30 Period=H08						
Sort=Last						
	Last	Max	Average	Min	Sum	Count
Synchro	1.00	418.00	1.17	1.00	9482971.00	8084986

Highlights:

- Function Selection – The Function menu () selects which aggregation function to display.
- Dynamic color selection
- Compare Dates/Periods in “Option Window/Format Specific” (Date format YYYYMMDD) allows to compare statistics from two Dates/Periods. Using this feature, the system displays what Time Series are missing in RED and the ones that were added in YELLOW. All the other Time Series (normal) are displayed in the default color. The following example displays missing/added processes of a host on a particular period of two (2) days.

SynchroServer: Synchro-server1.synchro-tech.com								
Component=frean01 Type=PROC Metric=CPU Period=00@08 Sort=Last								
	Time	DTime	Last	Max	Average	Min	Sum	Count
updatedb.mlocat_20120830	06:25:09		10.39	118.64	75.16	0.00	601.34	8
SynchroClient.jar_20120830	07:59:59		1.95	37.20	3.62	0.00	104418.22	28800
kworker-15:1_20120830	07:55:17		1.22	1.32	1.02	0.81	65.76	64
kworker-1:1_20120830	07:57:11		1.21	1.27	1.03	0.85	41.24	40
kworker-19:1_20120830	07:48:08		1.19	1.22	1.01	0.85	18.25	18
ksoftirqd-21_20120830	07:44:14		1.14	1.14	1.04	0.96	3.13	3
ksoftirqd-9_20120830	05:37:01		1.14	1.14	1.08	1.03	2.17	2
jbd2-dm-0-8_20120830	07:58:51		1.11	15.44	2.12	0.72	1348.42	636
ksoftirqd-11_20120830	07:57:03		1.11	1.23	1.03	0.89	11.39	11
kworker-11:1_20120831	04:21:59	00098	1.19	1.19	1.00	0.81	14.03	14
ksoftirqd-7_20120831	02:43:02		1.14	1.14	1.08	1.02	2.16	2
ksoftirqd-1_20120831	03:55:19		1.12	1.22	1.02	0.84	16.38	16
kworker-6:1_20120831	04:19:35	00242	1.10	1.20	1.02	0.92	12.26	12
kworker-9:1_20120831	04:22:54	00043	1.10	1.29	1.00	0.85	15.02	15
ksoftirqd-23_20120831	03:58:03		1.09	1.36	1.06	0.92	7.44	7
ksoftirqd-8_20120831	03:28:02		1.09	1.12	1.10	1.09	2.21	2
kworker-13:1_20120831	04:12:08	00689	1.08	1.25	1.03	0.89	16.56	16
ksoftirqd-10_20120831	--:--		1.07	1.07	1.07	1.07	1.07	1
SynchroServer.jar_20120831	04:23:36	00001	44.43	543.47	38.64	6.01	611284.74	15816
Injector.pl_20120831	04:23:36	00001	11.59	31.55	11.93	0.00	188755.83	15816
kworker-7:1_20120831	04:21:14	00143	1.25	1.37	0.99	0.87	23.90	24
khungtaskd_20120831	04:15:01	00516	1.10	1.10	1.10	1.10	1.10	1
ksoftirqd-12_20120831	03:32:02		1.06	1.06	0.97	0.86	4.89	5
ksoftirqd-13_20120831	03:58:54		1.06	1.20	1.04	0.95	6.26	6

- Alarms can be displayed on the window in this format

synchro

SynchroServer: Synchro-server1.synchro-tech.com								
Component=frean01 Type=HOST Metric=CPU Target=HOST Date=2012-09-01 Period=00@24 Sort=Last								
	Time	DTime	Last	Max	Average	Min	Sum	Count
frean01	17:02:09	00001	40.56	53.51	2.28	0.71	139107.37	60791

- Displays descriptive statistics

SynchroServer: Synchro-server1.synchro-tech.com								
Component=frean01 Type=HOST Target=HOST Date=2012-09-20 Period=00@24 Sort=Last								
	Time	DTime	Last	Max	Average	Min	Sum	Count
MEM	09:48:56	00002	24.68	36.77	27.59	20.74	975046.66	35337
CPU	09:48:56	00002	4.16	28.21	2.27	0.56	80265.22	35337
USER	09:48:56	00002	3.85	27.82	1.90	0.51	67446.49	35337
SYSTEM	09:48:56	00002	0.31	3.84	0.25	0.00	8938.64	35337
SWAP	09:48:56	00002	0.01	0.01	0.01	0.01	353.37	35337
WAIT	09:48:56	00002	0.00	4.71	0.01	0.00	655.85	35337
Max		2.00	24.68	36.77	27.59	20.74	975046.66	35337.00
Min		2.00	0.00	0.01	0.01	0.00	353.37	35337.00
Sum		12.00	33.01	101.36	32.03	21.82	1132706.23	212022.00
Count		6.00	6.00	6.00	6.00	6.00	6.00	6.00
Average		2.00	5.50	16.89	5.33	3.63	188784.37	35337.00
Median		2.00	2.08	16.26	1.08	0.26	38192.56	35337.00
Stddev		0.00	8.75	14.41	9.99	7.65	353070.04	0.00

Metric Table (HTML)

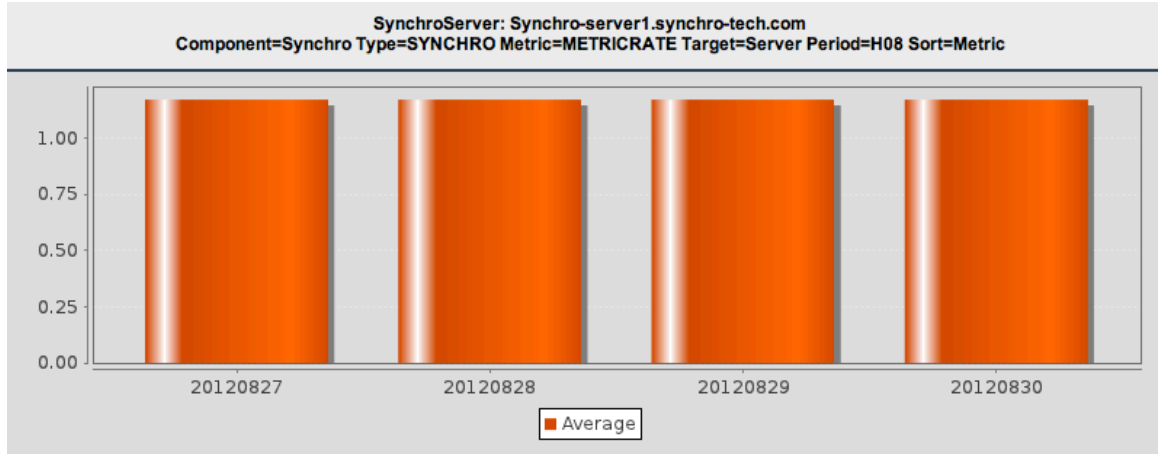
Displays a cross table of Metrics vs. the different dimensions of the Time Series. In this example, it displays the different Metrics available for a list of processes on a host.

SynchroServer: Synchro-server1.synchro-tech.com					
Component=frean01 Type=PROC Date=20120831 Period=00@08 Function=Max Sort=Max					
	CPU	MEM	MEMMBYT	RSS	RSSMBYT
SynchroServer.jar	543.47	69.98	16899.96	21.29	5143.61
SynchroClient.jar	46.06	4.54	1098.50	1.04	253.48
Injector.pl	31.55	0.12	29.18	0.02	6.45
flush-251:0	13.22	0.00	0.00	0.00	0.00
gnuplot.linux-3	10.34	0.19	45.93	0.08	21.39
jbd2-dm-0-8	8.85	0.00	0.00	0.00	0.00
kworker-23:1	1.59	0.00	0.00	0.00	0.00
irqbalance	1.37	0.06	15.41	0.00	0.51
kworker-7:1	1.37	0.00	0.00	0.00	0.00
ksoftirqd-23	1.36	0.00	0.00	0.00	0.00
ksoftirqd-3	1.36	0.00	0.00	0.00	0.00
kworker-0:1	1.35	0.00	0.00	0.00	0.00
kworker-14:1	1.35	0.00	0.00	0.00	0.00
rsyslogd	1.30	0.75	181.21	0.00	1.05
kworker-9:1	1.29	0.00	0.00	0.00	0.00

StatBarGraph

Displays a dynamic histogram of selected statistics.

synchro

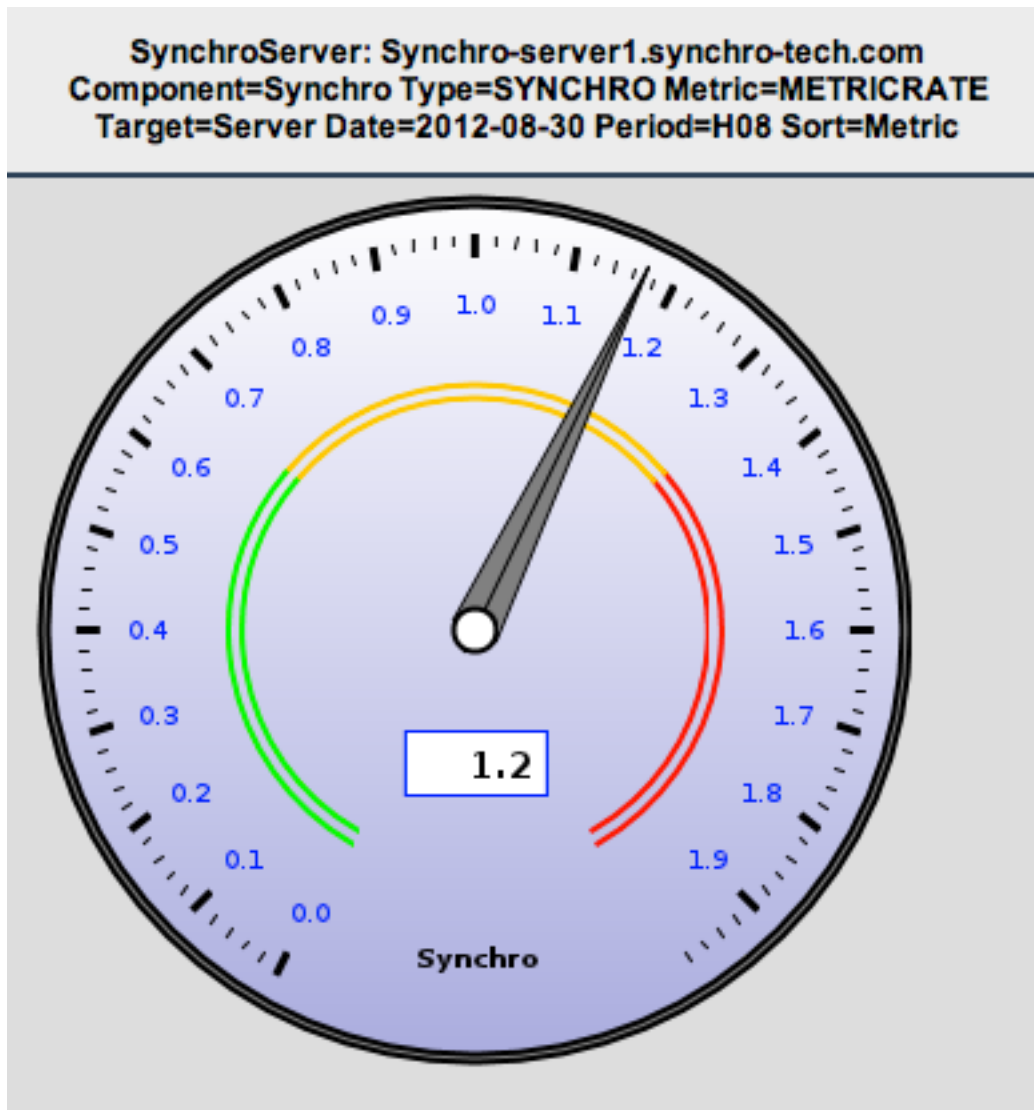


TIP: Use “WK-00” to select the current week of statistics, then “Sort/Metrics” to order them by name (Metric), and then “Options/Reverse Sort” to reorder them in increasing order.

Dial (Graph)

Displays a dynamic dial of selected statistics.

synchro



TIP: In Options, change the Min/Max of the dial and set the number of steps of the dial.

Custom Text (Graph)

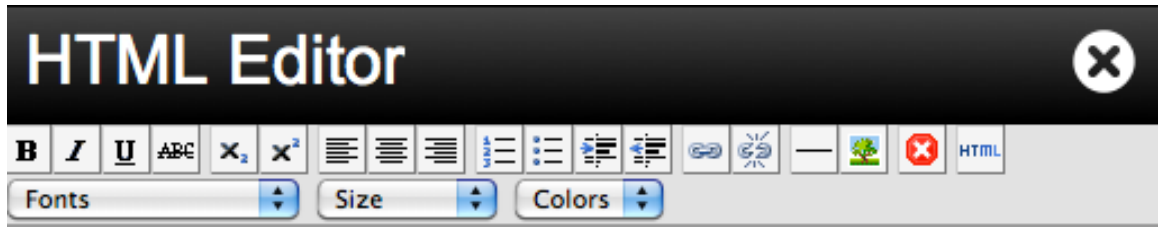
This format enables the user to create a text-based window that can display text and embedded statistics.

This is an example of Custom Text Format.

Synchro average value for the 20120827 was 1.17 for the day before it was 1.17.

To do this, use the “Options/Format Specific/Custom Text” editor to write the text. Individual statistics can then be inserted into the text using the “Options/Format Specific/Show Result Table” in the format $\${ColumnLine}$ as per the following example:

synchro



This is an example of Custom Text Format.

$\{A1\}$ average value for the $\{E1\}$ was $\{K1\}$ for the day before it was $\{K2\}$.

This is an example of Custom Text Format.

Synchro average value for the 20120827 was 1.17 for the day before it was 1.17.

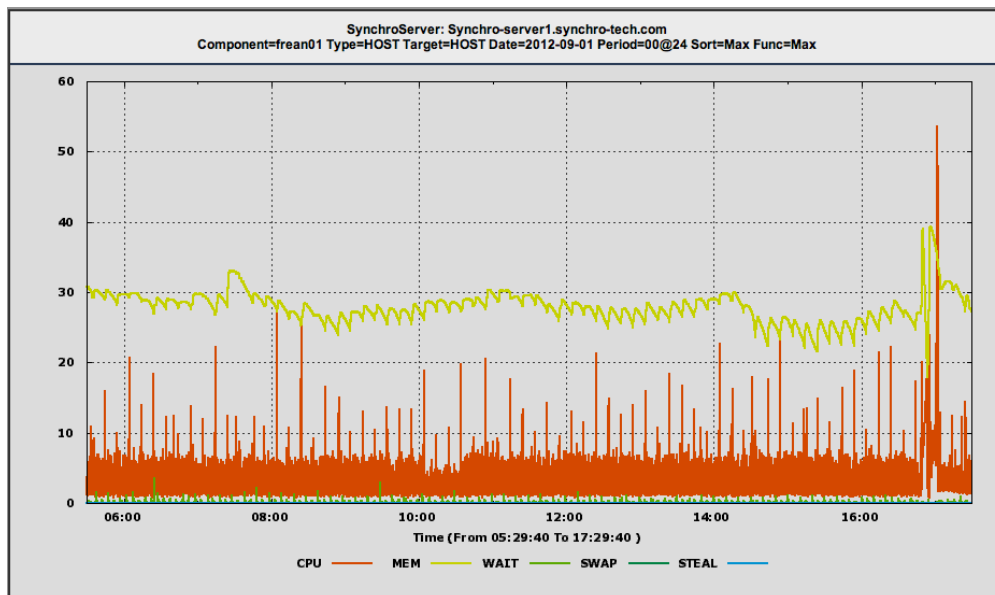
A	B	C	D	E	F	G	H	I	J	K	L	M	N	
0	Component	Type	Metric	Target	Date	Period	Time	DTime	Last	Max	Average	Min	Sum	Count
1	Synchro	SYNCHRO	METRICRATE	Server	20120827	H08	08:59:59	832524	1.00	423.00	1.17	1.00	9482422.00	8084987
2	Synchro	SYNCHRO	METRICRATE	Server	20120828	H08	08:59:59	832524	1.00	423.00	1.17	1.00	9482400.00	8084986
3	Synchro	SYNCHRO	METRICRATE	Server	20120829	H08	08:59:59	832524	1.00	418.00	1.17	1.00	9481998.00	8084990
4	Synchro	SYNCHRO	METRICRATE	Server	20120830	H08	08:59:59	832524	1.00	418.00	1.17	1.00	9482971.00	8084986

2.6.2 ByTimeStatistics Format

This section presents the format to report on, second by second.

Graph By Time (Graph)

This format (might be the most often used) displays the data against time for a series of Time Series.



Highlights:

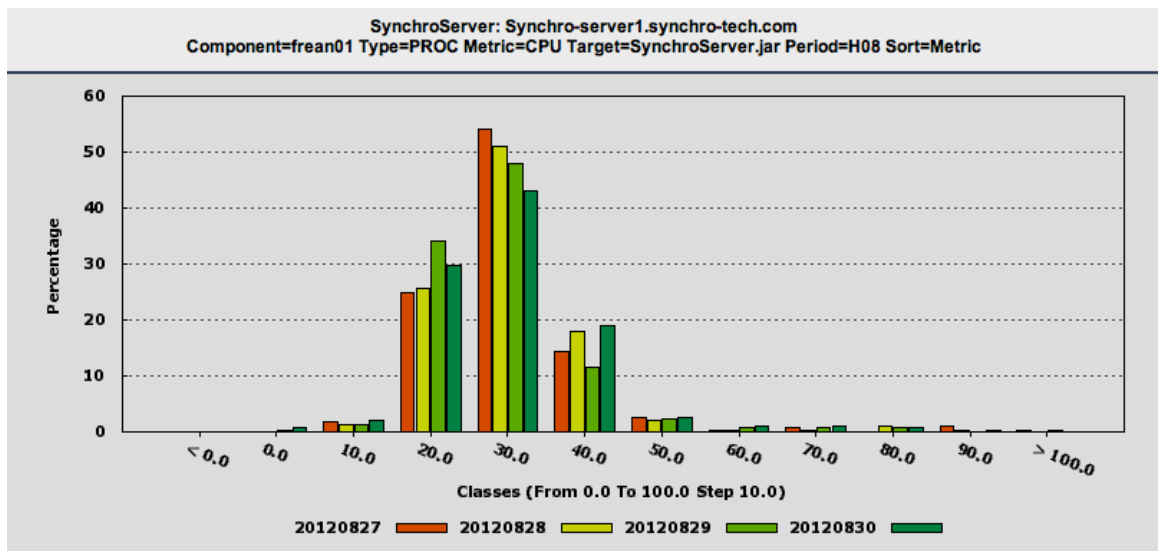
synchro

- Can display several aggregate functions and Time Series simultaneously.
- Can overlap or in-line several days of the same Time Series.
- Supports seconds scale
- Supports different types of line/points
- Can group, second by second, any displayed Time Series (Data and Range/GroupByTime)

Histogram Graph (Graph), Histogram (HTML), Cumulative Histogram (HTML)

Displays a dynamic histogram of the selected data. It expresses the percentage of events for each class.

The following example represents the percentage of time that a particular process is using a certain percentage of CPU over several days.



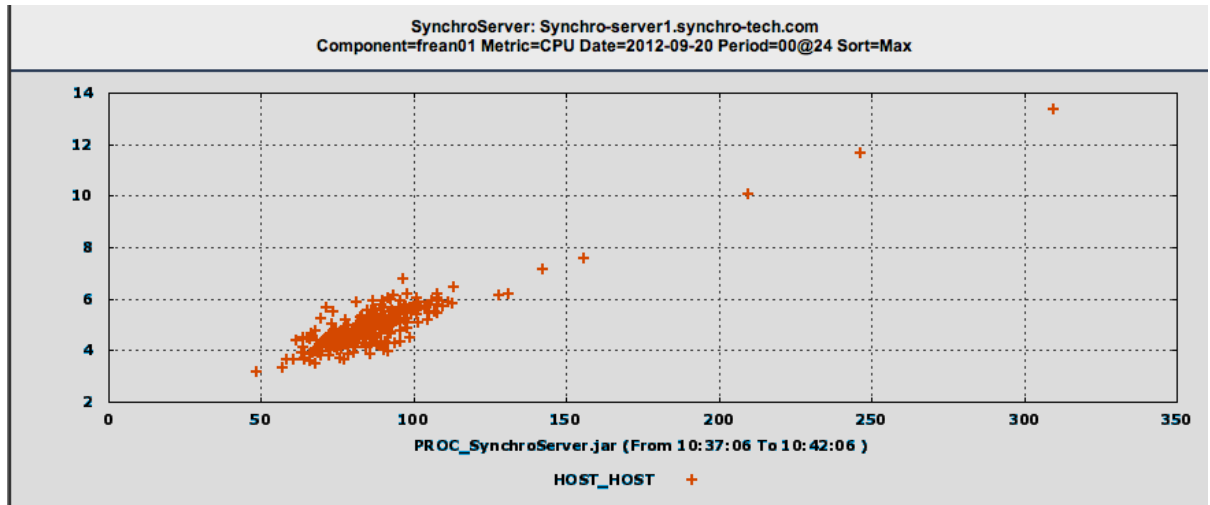
Highlights:

- Use “Options/Format Specific/” Min/Max/Step to set the parameters of the histogram.

Scatter Plot (Graph)

Displays the value of one Time Series against the values of another. In this example, it displays the percentage of CPU used by a process, and CPU usage on the host.

synchro



Highlights:

- Supports different types of line/points

Data By Time (HTML)

Displays the data against time for Time Series. In this example, it displays the percentage of CPU used by a process, and CPU usage on the host.

SynchroServer: Synchro-server1.synchro-tech.com Component=frean01 Metric=CPU Date=2012-08-30 Period=H08 Sort=Metric												
Time	PROC_SynchroServer.jar						HOST_HOST					
	Last	Max	Average	Min	Sum	Count	Last	Max	Average	Min	Sum	Count
08:05:09	79.35	79.35	79.35	79.35	79.35	1	1.73	1.73	1.73	1.73	1.73	1
08:05:08	38.33	38.33	38.33	38.33	38.33	1	2.12	2.12	2.12	2.12	2.12	1
08:05:07	42.33	42.33	42.33	42.33	42.33	1	2.16	2.16	2.16	2.16	2.16	1
08:05:06	79.06	79.06	79.06	79.06	79.06	1	3.90	3.90	3.90	3.90	3.90	1
08:05:05	63.52	63.52	63.52	63.52	63.52	1	2.58	2.58	2.58	2.58	2.58	1
08:05:04	50.78	50.78	50.78	50.78	50.78	1	2.63	2.63	2.63	2.63	2.63	1
08:05:03	40.65	40.65	40.65	40.65	40.65	1	2.08	2.08	2.08	2.08	2.08	1
08:05:02	31.68	31.68	31.68	31.68	31.68	1	1.97	1.97	1.97	1.97	1.97	1
08:05:01	29.30	29.30	29.30	29.30	29.30	1	2.68	2.68	2.68	2.68	2.68	1
08:05:00	42.33	42.33	42.33	42.33	42.33	1	1.68	1.68	1.68	1.68	1.68	1
MAX	79.06	79.06	79.06	79.06	79.06	1.00	3.90	3.90	3.90	3.90	3.90	1.00
MIN	29.30	29.30	29.30	29.30	29.30	1.00	1.68	1.68	1.68	1.68	1.68	1.00
SUM	447.24	447.24	447.24	447.24	447.24	10.00	23.73	23.73	23.73	23.73	23.73	10.00
SUM%	18.83%	18.83%	18.83%	18.83%	18.83%	0.42%	1.00%	1.00%	1.00%	1.00%	1.00%	0.42%
COUNT	10	10	10	10	10	10	10	10	10	10	10	10
AVERAGE	44.72	44.72	44.72	44.72	44.72	1.00	2.37	2.37	2.37	2.37	2.37	1.00
MEDIAN	41.44	41.44	41.44	41.44	41.44	1.00	2.14	2.14	2.14	2.14	2.14	1.00
STDDEV	15.10	15.10	15.10	15.10	15.10	0.00	0.62	0.62	0.62	0.62	0.62	0.00

Highlights:

- Displays descriptive statistics.
- Reverse time display
- Dynamic color selection
- Display Statistics

synchro

2.6.3 Miscellaneous Formats

These formats allow adding external content to the Views.

Custom Image (Graph)

Enables the user to add an external image from any WEB server by entering the URL of the image. This format is useful for adding graphs and logos to your Views. Use the “Options/Window Control/Update” to set the refresh rate in seconds to the desired frequency.

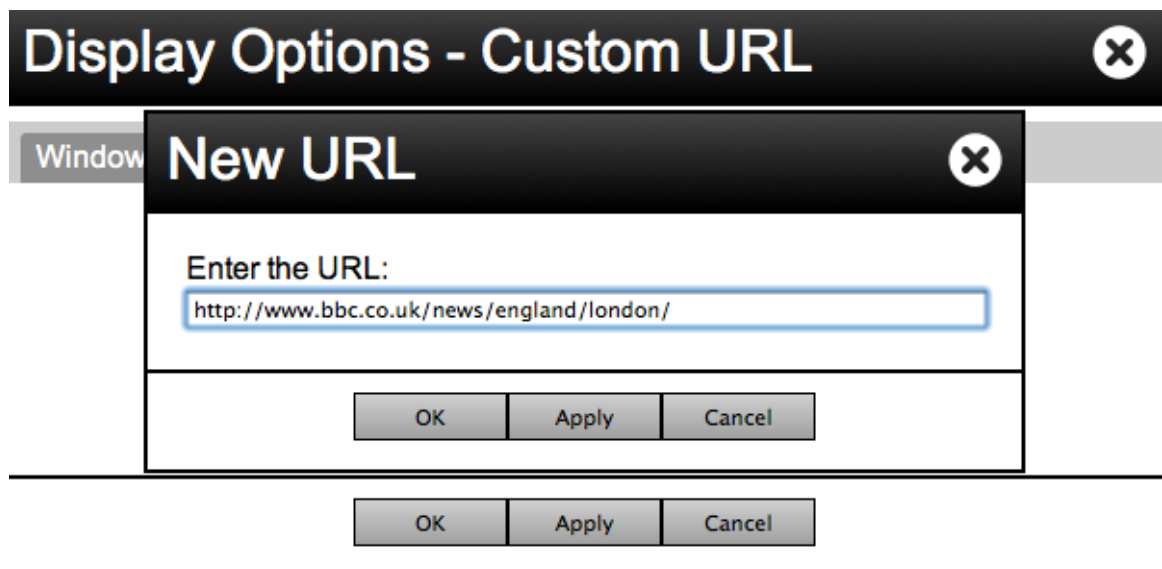


Highlights:

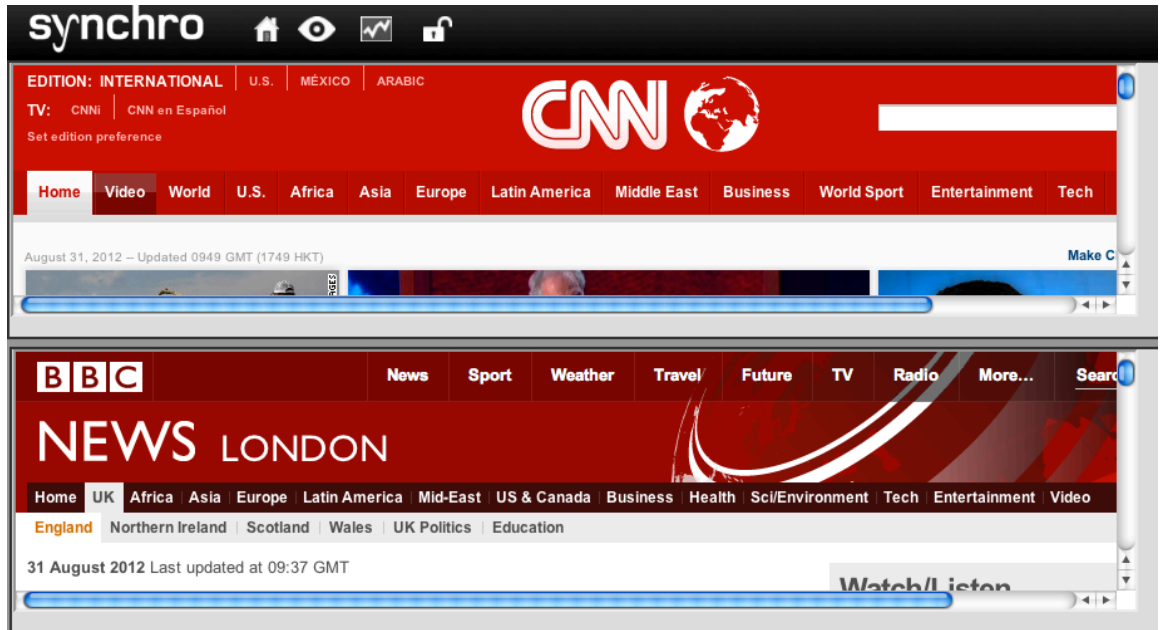
- Image can be stretched, cropped, or set to actual size.

Custom URL (HTML)

Enables the user to add the content of other web sites into a Synchro View by entering its URL in “Options/Format Specific/Custom URL” as follows. This format is useful when adding content from other WEB sites, and other SynchroServer to your Views. Use the “Options/Window Control/Update” to set the refresh rate in seconds to the desired frequency.



synchro



synchro

3 Synchro Administration

3.1 SynchroServer Installation

This section presents instruction to conduct the installation of the SynchroServer and SynchroClient software.

3.1.1 Pre-Requirement

SYSTEM: SynchroServer needs to be installed on a UNIX server (i.e., Linux SUSE, RedHat, Centos, Ubuntu..., or Solaris). It needs at least 1 core, 2GB of RAM, and enough disks to store data as per the retention policy.

A conformable configuration for SynchroServer would include 8 cores, 12 GB of RAM and SSD drives.

Note: Slower disks can be used to store historized data. Please refer to the “Synchro Performance Tuning” section for more information.

User: To install SynchroServer, the SynchroServer Administrator needs to decide on a user that will own the software installation, data and processes. Typically a dedicated user gets created for this purpose (for example: synchro, synchrodev synchroprod...). The user must have access to enough data to keep historical data according to the retention policy. Also, the user needs access to Java 6+ to run Synchro. Make sure Java is in the user's path.

```
synchro@SynchroHost:~$ java -version
java version "1.6.0_16"
Java(TM) SE Runtime Environment (build 1.6.0_16-b01)
Java HotSpot(TM) Server VM (build 14.2-b01, mixed mode)
synchro@SynchroHost:~$ which java
```

3.2 Installation Instructions

1. Make the package Executable

```
synchro@host:~$ chmod +x ./SynchroServer-2.0.4.bin
```

2. Run the Setup Program

```
synchro@host:~$ ./SynchroServer-2.0.4.bin
LICENSE AGREEMENT
```

```
IMPORTANT - PLEASE READ THIS FIRST. THIS IS A LICENSE
AGREEMENT.
```

```
AFFA TECHNOLOGY INC. ("AFFA") IS WILLING TO LICENSE THE
SYNCHRO SOFTWARE (THE "SOFTWARE") YOU (HEREAFTER REFERRED
TO AS "YOU") ONLY UPON THE TERMS AND CONDITIONS CONTAINED
IN THIS LICENSE AGREEMENT ("AGREEMENT").
```

...

Note: The license agreement will only appear on the demonstration version.

...

```
Do you accept the license agreement? [yes/no]
yes
Licence accepted, installing SynchroServer...
Backing up SynchroServer/www/Synchro to backup/20121011.135513
Backing up SynchroServer/www/SynchroAdmin to backup/20121011.135513
```

synchro

```
Backing up SynchroServer/www/SynchroDashBoard to backup/20121011.135513
Backing up SynchroServer/www/SynchroPlayer to backup/20121011.135513
Extracting and running setup.sh...
Replacing /SYNCHRO-INSTALL-DIR/ by /home/synchro/ in config files.
Modifying file: /home/synchro/SynchroServer/bin/start-server.sh
Modifying file: /home/synchro/SynchroServer/bin/stop-server.sh
Modifying file: /home/synchro/SynchroServer/bin/synchro-server.sh
Modifying file: /home/synchro/SynchroServer/bin/synchro-client.sh
Modifying file: /home/synchro/SynchroServer/bin/start-client.sh
Modifying file: /home/synchro/SynchroServer/bin/stop-client.sh
Modifying file: /home/synchro/SynchroServer/bin/synchrojmx
Modifying file: /home/synchro/SynchroServer/conf/server.ini
Done.
This will be used to configure remote SynchroClients
What is the name of this server [frea01]?

Preparing SynchroServer/dist/server.conf...
Verifying java installation
All done, installation complete.
```

SynchroServer is now installed. You can now connect and use the SynchroWeb interface at the following URL (from a web browser e.g, IE or Firefox):

<http://ServerName:8080/Synchro/Synchro.html>

3.3 SynchroServer Start/Stop Procedures

To START the SynchroServer:

```
synchro@SynchroHost:~$ cd
synchro@SynchroHost:~$ : SynchroServer/bin/start-server.sh
SynchroClient stopped.
SynchroServer started.
synchro@SynchroHost:~$
```

To STOP the SynchroServer:

```
synchro@SynchroHost:~$ cd
synchro@SynchroHost:~$ : SynchroServer/bin/stop-server.sh
SynchroClient stopped.
SynchroServer stopped.
synchro@SynchroHost:~$
```

3.4 SynchroClient Solaris Installation

Download Synchro – you can download and install as many as needed. They will connect automatically to the SynchroServer they were downloaded from.

The user that will run the SynchroClients needs access to Java 6+ to run Synchro. Make sure Java is in the user's path. This is not a pre-requisite for the Windows Synchro Client.

```
synchro@SynchroHost:~$ java -version
java version "1.6.0_16"
Java(TM) SE Runtime Environment (build 1.6.0_16-b01)
Java HotSpot(TM) Server VM (build 14.2-b01, mixed mode)
synchro@SynchroHost:~$ which java
```

Download the Solaris SynchroClient

```
wget http://ServerName:8080/download/SynchroSolaris.tar.gz
prompt $ wget http://ServerName:8080/download/SynchroSolaris.tar.gz
--2009-07-02 13:37:49-- http://ServerName:8080/download/SynchroSolaris.tar.gz
Resolving ServerName... xxx.xx.26.112
Connecting to ServerName [xxx.xx.26.112]:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 985896 (963K) [application/x-gzip]
Saving to: `SynchroSolaris.tar.gz'
```

synchro

```
100%[=====
==>] 985,896    --.-K/s    in 0.006s

2009-07-02 13:37:49 (151 MB/s) - `SynchroSolaris.tar.gz' saved [985896/985896]

#Untar the Client
prompt $ tar -zvxf SynchroSolaris.tar.gz
SynchroSolaris/
SynchroSolaris/updateclient.sh
SynchroSolaris/bin/
SynchroSolaris/bin/SynchroClient.jar
SynchroSolaris/bin/logging.conf.xml
SynchroSolaris/bin/start-client.sh~
SynchroSolaris/bin/start-client.sh
SynchroSolaris/lib/
SynchroSolaris/lib/slf4j-api-1.4.3.jar
SynchroSolaris/lib/logback-core-0.9.8.jar
SynchroSolaris/lib/mina-core-1.1.7.jar
SynchroSolaris/lib/snmp4j.jar
SynchroSolaris/lib/logback-classic-0.9.8.jar
SynchroSolaris/README

#Start the Client

prompt $ SynchroSolaris/bin/start-client.sh
Starting: SolarisStatThreadNew

#See the interface in a web browser (IE,Chrome, Firefox...)using the URL below
http://ServerName:8080/Synchro/Synchro.html

Use the "Views" menu (upper right of the screen) to select what you want to see.
```

To START the SynchroClient:

```
synchro@SynchroHost:~$ cd
synchro@SynchroHost:~$ : SynchroLinux/bin/start-client.sh
SynchroClient stopped.
SynchroServer started.
synchro@SynchroHost:~$
```

To STOP the SynchroClient :

```
synchro@SynchroHost:~$ cd
synchro@SynchroHost:~$ : SynchroLinux/bin/stop-client.sh
SynchroClient stopped.
SynchroServer stopped.
synchro@SynchroHost:~$
```

3.5 SynchroClient Linux Installation

Download Synchro – you can download and install as many as needed. They will connect automatically to the SynchroServer they were downloaded from.

The user that will run the SynchroClients requires access to Java 6+ to run Synchro. Make sure Java is in the user's path. This is not a pre-requisite for the Windows Synchro Client.

```
synchro@SynchroHost:~$ java -version
java version "1.6.0_16"
Java(TM) SE Runtime Environment (build 1.6.0_16-b01)
Java HotSpot(TM) Server VM (build 14.2-b01, mixed mode)
synchro@SynchroHost:~$ which java
```

Download the Linux SynchroClient

```
wget http://ServerName:8080/download/SynchroLinux.tar.gz
prompt $ wget http://ServerName:8080/download/SynchroLinux.tar.gz
--2009-07-02 13:37:49-- http://ServerName:8080/download/SynchroLinux.tar.gz
Resolving ServerName... xxx.xx.26.112
Connecting to ServerName|xxx.xx.26.112|:8080... connected.
```

synchro

```
HTTP request sent, awaiting response... 200 OK
Length: 985896 (963K) [application/x-gzip]
Saving to: `SynchroLinux.tar.gz'
```

```
100%[=====
==>] 985,896      --.-K/s   in 0.006s
```

```
2009-07-02 13:37:49 (151 MB/s) - `SynchroLinux.tar.gz' saved [985896/985896]
```

```
#Untar the Client
prompt $ tar -zxvf SynchroLinux.tar.gz
SynchroLinux/
SynchroLinux /updateclient.sh
SynchroLinux /bin/
SynchroLinux /bin/SynchroClient.jar
SynchroLinux /bin/logging.conf.xml
SynchroLinux /bin/start-client.sh~
SynchroLinux /bin/start-client.sh
SynchroLinux /lib/
SynchroLinux /lib/slf4j-api-1.4.3.jar
SynchroLinux /lib/logback-core-0.9.8.jar
SynchroLinux /lib/mina-core-1.1.7.jar
SynchroLinux /lib/snmp4j.jar
SynchroLinux /lib/logback-classic-0.9.8.jar
SynchroLinux /README
```

```
#Start the Client
```

```
prompt $ SynchroLinux /bin/start-client.sh
Starting: LinuxStatThreadNew
```

```
#See the interface in a web browser (IE,Chrome, Firefox...)using the URL below
http://ServerName:8080/Synchro/Synchro.html
```

Use the "Views" menu (upper right of the screen) to select what you want to see.

To START the SynchroClient:

```
synchro@SynchroHost:~$ cd
synchro@SynchroHost:~$ : SynchroLinux/bin/start-client.sh
SynchroClient stopped.
SynchroServer started.
synchro@SynchroHost:~$
```

To STOP the SynchroLinux :

```
synchro@SynchroHost:~$ cd
synchro@SynchroHost:~$ : SynchroLinux/bin/stop-client.sh
SynchroClient stopped.
SynchroServer stopped.
synchro@SynchroHost:~$
```

synchro

3.6 SynchroClient Windows Installation

Download Synchro – you can download and install as many as needed. They will connect automatically to the SynchroServer they were downloaded from.

Monitoring a Windows Host (in a browser)

```
prompt $ wget http://ServerName:8080/download/SynchroClient.exe  
prompt $ wget http://ServerName:8080/download/SynchroSolaris.tar.gz
```


Execute the installation package. The Synchro client will install itself as a WINDOWS service.

It is possible to deploy the SynchroWindows without having the installation script displaying the notification window. For this just a “-quiet” on the command line.




synchro

4 SynchroServer Configuration

This section describes the SynchroServer configuration. The SynchroAdmin interface is the tool the Synchro Server Administrator uses to configure the Synchro Server.

It can be accessed via the SynchroWeb interface under the “View Control”/ . Configuration parameters are saved in the <SynchroServer/conf> directory. These configurations are not affected by the installation or upgrade process.

4.1 Basic Admin Controls

Function Icon	Description
	The Action menu provides functionality like Stop the Server, Open and Close the Statistics ports...
	All SynchroServer configurations can be set in this menu. This includes the SynchroServer configuration, Users, Moving Averages, Alarms, Periods...
	Let the user defined the content of the “Window Data Controls” presets

4.2 Action Menu

Function Icon	Description
Stop Server	Stops The SynchroServer
Garbage Collect	Manually run the Java JVM garbage collection process
Close Stat Port	Close the Statics acquiring port thus disconnection all probed and agents – only the query port will remain open (For SynchroWEB and SynchroAdmin interfaces).
Open Stat Port	Opens that statistics collection port permitting clients and probes to connect.
Load Layouts	Manually Load the Views Layouts from disk without rebooting the server.
Load Reports	Manually Load the Reports from disk without rebooting the server.
Load Presets	Manually Load the Data Controls Presets from disk without rebooting the server.
Run Retention	Manually run the retention process
Run Compression	Manually run the compression process
Run Reports	Manually run the report creation process.

synchro

4.3 SynchroServer Configuration

This section presents the most important configuration parameters of the SynchroServer.

4.3.1 Server - Configuration

Function Icon	Description
Synchro Title	Sets the SynchroServer name that appears in the title of Data Windows
Host Name	Is the host name or IP address of the host that run the SynchroServer – this parameter is inserted in the SynchroClients packages so that they will connects to the SynchroServer without shaving to set the parameter each time.
Stat Port	TCP port that is used by clients and probes to connect to the SynchroServer and send Time Series statistics.
HTTP Port	Port used by SynchroWEB and SynchroAdmin to query the SynchroServer.
Redirect HTTP Root	Parameter that set the redirect from http://SynchroServer to http://SynchroServer/Synchro/Synchro.html
Lock View	Parameter that sets the SynchroServer in a mode where a login is required to access any View
Memory Allocation	The amount of memory available to the SychroServer is fixed at start time. By default, the SynchroServer will not use more than 60% of available physical memory available on the host running the server. This can be change using this parameter. (unit: Megabyte)

4.3.2 Server – Retention

Function Icon	Description
Data Execute Time	Sets the execute time of the retention utility on the SynchroServer Data
Log Execute Time	Sets the execute time of the retention utility on the SynchroServer Logs
Log Nb Days	Sets the number of days the SynchroServer will keep its internal logs
Report Execute Time	Sets the execute time of the report utility
Report Nb Days	Sets the number of days the SynchroServer will keep the reports it created

synchro

4.3.3 Server – Compression

Function Icon	Description
Execute Time	Sets the execute time of the compression utility

4.3.4 Server – Aggregator

Function Icon	Description
Aggregator Time	Sets the execute time of the Aggregator utility

4.3.5 Server – Logging

Function Icon	Description
Use Synchro Log	Determine if SynchroServer will create logs
Synchro Log Format	Sets Synchro Log formats String – See below for the parameters
Use Syslog	Determine if SynchroServer will use the Unix Syslog
Syslog Format	Sets the Syslog formats String – See below for the parameters
Syslog Facility	Sets the Syslog facility to use
Syslog Path	Path of the Syslog logd
Syslog Protocol	Sets the Syslog access mode
Syslog Host/Port	Sets the Host and the Port to access the Syslog
Use Email	Determine if SynchroServer will use emails to send alarms
Email Log Format	Sets the Email alarm formats String – See below for the parameters
Email From	Sets the address of the sender
Smart SMTP Server	Name or IP of the Email server
Smart SMTP Server Port	Port to use to connect to the Email server to send Alarms

4.3.6 Logs and Alarm Parameters

%D: Date

%A: alarm name

%f: feed time HHMMSS

%F: feed time HH:MM:SS

%p: SynchroServer pid

synchro

%C: Component

%T: Type

%M: Metric

%t: target

%d: dtime

%l: last

%m: max

%a: average

%n: min

%s: sum

%c: count

%O: old alarm value

%N: new alarm value

%L: alarm value in syslog severity level: INFO, WARN, CRIT, FAIL

4.3.7 Server - Internal Parameters

Function Icon	Description
Merge Engine Count	Number of internal engine that processes incoming Statistics from clients and probes
Result Engine Count	Number of internal engine that processes incoming HTTP queries from SynchroWEB interfaces
Freemem Trigger Level	Percentage of free memory threshold after which the SynchroServer switched into memory protect mode and disconnects all clients and probe leaving only the HTTP port open. Note that a disconnection exception list can be configured – See Memory Trigger Exception below
GCRuntime Trigger Level	Duration in ms of the Java Garbage Collect process after which the SynchroServer switched into memory protect mode and disconnects all clients and probe leaving only the HTTP port open. Note that a disconnection exception list can be configured – See Memory Trigger Exception below
Synchro Internal Metrics	Sets the level of detail of the SynchroServer logs.

4.4 Alarm Configuration

Synchro can generate alarms on PeriodStatistics of any Time Series based on pre-defined thresholds.

Alarm level range values are defined as; 0=Normal, 1=Warning, 2=Critical, 3=Fatal, 3+=Fatal. Alarms can be set on the following statistical aggregation functions of any Time

synchro

Series for a specific period: LAST, AVERAGE, LAST, MAX, MIN, DTIME (Delta in seconds since last update). Each Alarm in Synchro produces a Time Series named after the name of the alarm. This enables the user to track and report on the resulting Time Series as for any other Time Series.

ALARMS INTEGRATION

The alarm engine produces a log of the activity of the Alarm Time Series. Alarm logs can be found here: **SynchroServer/log/AlarmLog-<Date>.log**

It is also possible to use Syslog and Email for integration. Syslog and Email configuration can be configured in the Server Configuration tab of the SynchroAdmin. The string that defines the alarm can be changed, as needed – See Alarm string definition and parameters in the SynchroServer parameters.

Function Icon	Description
Name	Name of the Alarm
Filter	Set the Time Series on which the alarm will be defined
Email	List of Email addresses to send the Alarm
Function	Aggregation function t set the Alarm -
OK/Warn/Critical/Fatal Trigger Level	Sets the minimum value after which an Alarm of the respective lever will be raised
Trigger Alarm On	Sets if an Alarm will be raised when the Alarm level augments, diminishes or both
Trigger Alarm if level equal or higher to	Sets the minimum level after which and Alarm is raised

4.5 MAVG Configuration

Any number of dynamic moving averages can be defined. The resulting Time Series name becomes the Metric name. It is possible to create alarms on the Moving average Time Series instead on the underlying data. This is useful to create less sensitive alarms.

Function Icon	Description
Name	Name of the moving average
Duration	Length in seconds of the moving average
Filter	Set the Time Series on which the moving average will be defined
Algo	Selects a moving average algorithm SumDivCount uses avg = sum of value divided by count of value received (doesn't count NA or missing values) SumDivDuration uses avg = sum/duration so NA are simulated as 0

synchro

4.6 Color Configuration

Configuration parameters for:

- Alarm: Color of alarm message displayed in Statistics Format.
- Background: Color of the Data Window Background
- Data: Color of Data displayed in Statistics and DataByTime.
- Line: Color of line to be displayed in GraphByTime
- Static: Color of Non dynamic color.

Colors supported also "Hide" and "Transparent".

Function Icon	Description
Range	Defines a range of values that will use the color defined
Description	Name of the range – this name may be applied to the data in some cases
Minimum Value	Minimum value that will use the color
Color	Color definition acceptable in HTML: ie: red, green, blue, #RRGGBB, rgb(0,0,0), rgba(0,0,0,0)

4.7 Users Configuration

Function Icon	Description
Full Name	Full Name of the user
User Name	User name that will be used for login
UID	User identification – generated by the SynchroServer
GID	Group ID – Users in group "1" gets their selected data window generated as reports.
Report Time	Time at which the reports are generated for this particular user
Read-Only Account	Sets the account as read only – so its can't create or modify Views
Read views of users	Sets that account so it can only see the Views created by a specific set of users

4.8 Periods Configuration

In Synchro it is possible to define Periods. A Period describes a continuous interval of time of the day. For each Period and for each Time Series, Synchro provides the Last (Last value), Min, Max, Average, Count, Sum, Time (Time of last update), DTimes (Time since last update). These Periods based statistics are called PeriodStatistics. By default, Synchro keeps hourly, morning, evening, work hours, and daily PeriodStatistics. Users can define as many, possibly overlapping, Periods as needed.

synchro

Function Icon	Description
Name	Name of the Period
Start Time	Start Time – Format: HHMMSS
End Time	End Time – Format: HHMMSS

4.9 Aggregator Configuration

Synchro can group together several days of data for particular Time Series. At a time set in the SynchroServer configuration, the Aggregator reads historical data from Time Series to create a new one that represents the historical data.

Function Icon	Description
Description	Name of the Time Series
Number of Days	Number of days in the pas to read and group together
Dates	<date-reg-ex> <reldays> <rel-day-of-week> <dates> date-reg-ex := YYYYMMDD (reg ex with .* [^ \$ and numbers to describe a date with format YYYYMMDD) reldays := D-XX D-XX..D-XX (where XX is number of days relative to current date when filter runs) rel-day-of-week := W-XX (Where XX is number of working days relative to current date when filter runs not counting week-end ie: D-01 on monday is Friday)
Filter	Set the Time Series on which the Aggregation will be defined

4.10 Compression Configuration

Data compression rules can be defined to compress data. Rules can be prioritized, as needed using the UP/DOWN arrows.

Synchro stores raw data it receives in binary sparse files in the directory <SynchroServer/data/metrics>. These sparse files can be change any time. Data can be inserted “out of order” form the time perspective.

Compression of data runs daily to limit the amount of disk used by the raw sparse file. Once compressed, the data becomes read only (cannot be changed) – these compressed files are stored in the directory <SynchroServer/data/metrics.gz>.

FOR LARGE SYNCHRO CONFIGURATIONS

Synchro does a significant amount of small IO in numerous sparse files that take a relatively small amount of disk space – it is a recommended practice to set these files on fast disks (SSD drives are perfect for this job).

The compressed data (read only) can take significantly more space (depending retention policy) and is less IO intensive – it is a recommended practice to set this data on slower large disks infrastructure. The Synchro administrators can easily change the location of the directories using symbolic links.

synchro

Function Icon	Description
Number of Days	Number of days after which the data will be compressed and become read only
Dates to keep	Exception to the rule can be made in the following format: <date-reg-ex> <reldays> <rel-day-of-week> <dates> date-reg-ex := YYYYMMDD (reg ex with .* [^ \$ and numbers to describe a date with format YYYYMMDD) reldays := D-XX D-XX..D-XX (where XX is number of days relative to current date when filter runs) rel-day-of-week := W-XX (Where XX is number of working days relative to current date when filter runs not counting week-end ie: D-01 on monday is Friday)
Filter	Set the Time Series on which the rule will be defined

4.11 Retention Configuration

Data retention rules can be defined to destroy unused data. Rules can be prioritized as needed using the UP/DOWN arrows.

Function Icon	Description
Number of Days	Number of days after which the data will be deleted
Dates to keep	Exception to the rule can be made in the following format: <date-reg-ex> <reldays> <rel-day-of-week> <dates> date-reg-ex := YYYYMMDD (reg ex with .* [^ \$ and numbers to describe a date with format YYYYMMDD) reldays := D-XX D-XX..D-XX (where XX is number of days relative to current date when filter runs) rel-day-of-week := W-XX (Where XX is number of working days relative to current date when filter runs not counting week-end ie: D-01 on monday is Friday)
Filter	Set the Time Series on which the rule will be defined

4.12 Client Configuration

Sets the configuration of the SynchroClient running on the SynchroServer. See APPENDIX A – to learn more about the SynchroClients configuration.

4.13 Dropfrom Configuration

Sets a List of hosts or IPs (one per line) from which the SynchroServer will not accept connection on the STAT port (stop receiving Time Series values).

If needed, it is also possible to close the STAT port all together so the server becomes “read only” – leaving only the HTTP port open for SynchroWEB interfaces.

synchro

4.14 Memory Trigger Exception Configuration

The SynchroServer can automatically switch to protect mode in condition where memory available is too low. This condition is detected using 2 indicators:

- Percentage of free memory threshold
- Duration in ms of the Java Garbage Collect process

These parameter values can be set in the SynchroServer configuration menu. Once one of these thresholds is reached, SynchroServer switches into memory protect mode and disconnects all clients and probes leaving only the HTTP port open.

The Memory Trigger Exception Configuration List is composed of a set of hostnames of IPs (One that will not be disconnected by SynchroServer when switched into protect mode.

4.15 Time Series Mapping Configuration

The SynchroServer can create new Time Series based of existing ones by grouping them together. The Time Series mapping configuration defines a list of rules that tells the SynchroServer how to group the Time Series together. This scheme is also a simple way to change the name of a Time Series as needed.

The syntax is:

Component1_Type1_Metric1_Target1= ComponentA_TypeA_MetricA_TargetA

Component2_Type2_Metric2_Target2= ComponentA_TypeA_MetricA_TargetA

4.16 Value Mapping Configuration

The SynchroServer can receive events using the trivial Text based TCP protocol – See APPENDIX B for more details. These events can be represented by a string characters like “Open”, “Close” that are converted by the SynchroServer into numerical values to be processed as such like any other Time Series.

SC01 [m]stat server.time type_metric_target=~sometextvalue

The Value Mapping configuration defines a set of rules that tells the SynchroServer how to handle these strings of characters by associating them with a numerical value.

Format:

sometextvalue=[number]

4.17 Presets Configuration

Window Controls are available when the mouse pointer hovers on the upper left portion of any Data Window (below the Title region). The Preset configuration allows the modification of some of these menus as needed.

Appendix A. SynchroClients

SynchroClients are used to collect technical metrics from computers. The metrics are collected every seconds and includes the following technical measurements when available:

Target	Type	Metric
Host Level	HOST	CPU, MEM, SWAP, PLIST, RUNQ, NICE, STEAL, SYSTEM, USER, WAIT, (Globally and per core when available)
All Processes running on the host	PROC	CPU, MEMPCT, RSPCT, RSSMBY, MEMMBY
All network interface attached to the host	NET	RXPCKSEC, TXPCKSEC, TXKBYSEC, RXKBYSEC
All disk attached to the host	DISK	TPS, WSEC, RSEC, KBYWSEC, KBYRSEC, IOWAIT, QLEN, AWTIME, IOMSEC, STIME
All File Systems Mounted	FS	AVAIL-GB, SIZE-GB, USAGE-PCT, USED-GB

A.1 SynchroLinux

It is possible to throttle the SynchroLinux SynchroClient. There is a configuration file present on each SynchroLinux clients:

```
synchro@affa2:~$ cat SynchroLinux/conf/client.ini
CPU=yes
DISK=yes
MEM=yes
NET=yes
PROC=yes
FS=yes
PROC-CPU-THRESHOLD=0.01
PROC-MEM-THRESHOLD=0.011
PROC-RSS-THRESHOLD=0.01
```

The Boolean switches CPU (CPU usage and other metrics at host level), DISK (per disks subsystem metrics), MEM (memory usage at the host level), NET (network interfaces metrics), PROC (processes metrics), FS (file system metrics) are used to turn on and off the associated metric collection by the SynchroClient.

If the PROC switch is on (“yes”) (Process metrics collection is activated), you can use the PROC-XXX-THRESHOLD to fix the level of sensibility to processes (expressed in percentage). For example a value PROC-CPU-THRESHOLD=10 will report statistics only for processes that use at least 10% of one core of CPU (thus removing some of processes and keeping only “important processes”).

A.2 SynchroWindows

It is possible to deploy the SynchroWindows without having the installation script displaying the notification window. For this just a “-quiet” on the command line.

Appendix B. Application Integration

This section presents information on how to integrate Synchro with others applications.

B.1 Trivial TCP Text Protocol

The following is a description of the protocol to use.

Here is the client protocol to integrate an application to Synchro.

Protocol Overview:

- The protocol is text based.
- Each message is one line of text.
- Messages must be cleaned form any of there caracters:[\|~|#@] and not printables
- Each field of the message is separated by a tab character.
- Each message starts with a special header.
- You can test the protocol using: telnet <server> 9123
- When the client connects, it must identify itself using the init message. The server will answer with the conf message.
- When init is done, the client can now start sending stat messages.
- Sending dumpconf message is completely optional

Here is the description of the different messages:

***Client init message:** The `/*init*/` message is mandatory and must be the first message sent to the server. This message is used to identify the client at the end of the connection.

Header:	SC01
Command:	<code>/*init*/</code>
Parameter:	<client name>
End-of-message:	\n

For example, this would be the C string for the `*init*` message if the client name is Computer1:

```
sprintf( s, "SC01\tinit\t%s\n", "Computer1" );
```

Client stat message:The `/*stat*/` message is used to send the metric values to the server.

Header:	SC01
Command:	<code>/*stat*/</code>
Parameter1:	< YYYYMMDD.HHMMSS >
Parameter2:	<metric value>
...	
Parameter<n>:	<metric value>

This would be the C string for the `/*stat*/` message:

```
sprintf( s, "SC01\tstat\t20080101.235559\tPROC_CPU_perl=32.99\n", );
```

The date format is:

```
sprintf( s, "%04d%02d%02d.%02d%02d%02d", year, month, day, hour, minute, second );
```

Alternatively, the user can select to use the server time instead of the client. To use the server time replace “YYYYMMDD.HHMMSS” by “server.time” string.

For example: the metric value format is (PROC is the type, CPU is the metric, perl is the target, 32.99 is the actual cpu time of the perl process):

```
sprintf( s, "%s_%s_%s=%8.2f", "PROC", "CPU", "perl", 32.99 );
```

You can append many metrics to the message, as long as they are all from the same date/time. Each metric value must be separated by a \t.

```
sprintf( s, "SC01\tstat\t20080101.235559 \tPROC_CPU_perl=32.99\tPROC_MEM_perl=4.32\n", );
```

***Client dumpconf message:** The /*dumpconf*/ message is not mandatory and is used to send back to the server some configuration information about the client.

```
Header:          SC01
Command:         /*dumpconf*/
Parameter1:     <filename>
Parameter2:     <line1>
Parameter3:     <line2>
...
Parameter<n>:   <line<n>>
```

This would be the C string for a /*dumpconf*/ message:

```
sprintf(s, "SC01\tdumpconf\tfile1\tline1 \tline2\n");
```

***Server conf message:** The server /*conf*/ message is used by the server to tell the client about some configuration parameters.

```
Header:          SS01
Command:         /*conf*/
Parameter1:     Stat=n
Parameter2:     DumpConf=1
Parameter3:     <line1 of client conf file>
Parameter4:     <line2 of client conf file>
Parametern:     <line<n> of client conf file>
```

Stat=n is the number of seconds the server is expecting stat message from the client. Currently used by the Windows and Unix client for the general metric values. It is ignored by the ExternalSource plugins.

DumpConf=0 or DumpConf=1 is the server asking for basic configuration information which the client should send when the first ever connection is made. This is usually to help diagnose and identify client setup and host machine. It is not mandatory for the client to answer the DumpConf request.

Telnet session sample:

```
telnet 192.168.7.101 9123
Trying 192.168.7.101...
Connected to 192.168.7.101.
Escape character is '^]'.
SC01  init      Computer1
SS01  conf      Stat=10 DumpConf=1
SC01  stat      20080101.112233 PROC_CPU_perl=32.99      PROC_MEM_perl=4.56
SC01  stat      20080101.112234 PROC_CPU_perl=31.89      PROC_MEM_perl=4.76
SC01  dumpconf   HostConf          Linux machine with 4 cpu

### Previous line has spaces but not tab  last line

SC01  stat      20080101.112235 PROC_CPU_perl=32.77      PROC_MEM_perl=5.02
^]q

telnet> q
Connection closed.
```

B.2 SynchroTail

SynchroTail is a very useful tool to read application logs. It understands log file patterns, and can change files automatically when new files are created. SynchroTail can also be restarted to continue reading from the file it was reading when it was stopped.

Here are some of the parameters of SynchroTail -

```
java -jar SynchroTail.jar "FilePattern=<regex>" [FirstFile=<filename>]
[SortBy=time|filename] [FileTimeout=<nb seconds>] [EOFTimeout=<nb seconds>]
[BaseDir=<dirname to search>] [ZTail=yes|no]
```

All parameters are optional except for FilePattern.

FilePattern	Needs to be a valid regular expression (not shell match).
FirstFile	Allow skipping of files (as in a restart mode) – files will be skipped until the FirstFile.
SortBy	“time” or “filename” (case sensitive)
FileTimeout	Is the number of seconds that the program will wait for the EOF before concluding that the file is completed. Default is five (5) seconds.
EOFTimeout	Is the number of seconds that the program will wait for new files before concluding that no additional files are to be processed. Default is five (5) seconds. The program waits FileTimeout on a particular file, and then waits EOFTimeout to see if other files are to be processed.If no file appears, the program stops.
RestartFileName=filename	Is the name of the file where SynchroTail will store the information needed to restart.
Restart	Is the parameter used to request SynchroTail to restart at the offset in the files (where it was when it was stopped).

Other parameters to control flow:

lps=XXX	Lines Per Second is the maximum number of lines per second that can be read in a file by SynchroTail.
delimiter=Char	Is the character to be used as a delimiter in place of end-of-line.
mbps=XXX	Mega Byte Per Second is the maximum number of bytes per second that can be read in a file by SynchroTail.

B.3 Integration Examples

This section presents example on how to integrate new ???????

Example – Perl Fusion IO Monitor

In this example, the user wants to monitor the number of FreeLebs in FusionIO drives. The script uses the server time instead of the client time.

```
#!/usr/bin/perl -w
BEGIN
{push @INC, '/home/synchro/FusionIOMonitor/';}

use Net::Telnet ();

$Server="SynchroServerHostName";
$Port=9123;

$ServerID = new Net::Telnet (Telnetmode => 0, Output_record_separator => "");
$ServerID->open(Host => $Server,Port => $Port);

$HOST=`hostname`;chomp($HOST);

$ServerID->print("SC01\tinit\tFIOMonitor\n");
$line = $ServerID->getline;
print ($line);
while (1){
    $X=`cat /proc/fusion/fio/fioa/data/free_lebs`;chomp($X);
    $Line1= "SC01\tmstat\tserver.time\tFIOMonitor\_FIO\_FreeLebs\_ $HOST-fioa\=$X\t";
    ServerID->print("$Line1\n");
    sleep (1);
}
```

Example – Bash Latency Monitor

In this example, the user wants to monitor the latency between a Linux server and some hosts list contained in a file (HostList.txt). There is one host name per line. The user uses the host where the script is running time instead of the SynchroServer's time.

```
FILE CONTAINING HOST LIST:
synchro@affa2:~$ cat HostList.txt
HostName1
HostName2
synchro@affa2:~$

SCRIPT:
#!/bin/bash

while (true); do
    if [ "$Init" = "1" ]; then
        Time=`date +%Y%m%d.%H%M%S`;
        echo -en "SC01 mstat $Time\t"
        for Host in `cat HostList.txt`; do
            Latency=`ping -c 1 affa8|grep "time=" |cut -d= -f4|cut -d' ' -f1,1`;
        done
    fi
done
```

```

        echo -en "LatencyMonitor_Latency_Latency_${Host}=${Latency}\t"
        done; echo ""
    else
        echo -e "SC01\tinit\tLatencyMonitor";Init=1;
    fi; sleep 1;
done|telnet localhost 9123

```

Example – Application Monitor using SynchroTail

The following is an example of a log file adapter using the SynchroTail application. SynchroTail monitors the end of files (EOF), using file pattern, as they are created by the application and prints out on STDOUT the data added to the files in real time (it tails application log files).

In this example, a simulator creates an hourly log file representing transaction made by clients on products. The format of the log file name is as follow:

LogFile-YYYYMMDD-HH.log where

- YYYY is the Year, MM is the Month, DD is the Day
- HH is the Hour.

The format of the log is

- Date (YYYYMMDD);
- Hour (HHMMSS);
- ClientID (Integer);
- ProductID (Integer);
- Qty (Integer).

EXAMPLE OF FILE CONTENT

```

host:~/TailExample/bin$ head ../data/LogFile-20120927-01.log
20120927;010000;00001;00008;00677
20120927;010000;00002;00007;00956
20120927;010000;00001;00003;00021
20120927;010000;00005;00000;00253

```

See the simulator script `ExampleFileCreation.pl` below.

The perl script (`Injector-Tail-Example-01.pl`) that parses the data created by the application (a simulator in this case) uses SynchroTail and pipe the data it reads as the data becomes available.

The outcome is a second by second aggregation so the transaction per Client and per Product.

USAGE EXAMPLE

```

host:~/TailExample/bin$ ./Injector-Tail-Example-01.pl -component TailExample -server
192.168.0.98 -file LogFile-20120928\.*log -dir ../data

```

SCRIPT

```

host:~/TailExample/bin$ cat Injector-Tail-Example-01.pl
#!/usr/bin/perl -w

use English;
use Getopt::Long;

```

```

use IO::Handle;
use Net::Telnet ();

my $Version="Version 0.0.3";

my %opts = (
    "help"           => \$Help,
    "version"       => \$ShowVersion,
    "Dir=s"         => \$Dir,
    "Date=s"        => \$Date,
    "Server=s"      => \$Server,
    "Port=s"        => \$Port,
    "Component=s"   => \$Component,
    "Java=s"        => \$Java,
    "LogDir=s"      => \$LogDir,
    "LogFile=s"     => \$LogFile,
    "File=s"        => \$File,
    "Debug"         => \$Debug,
    "Restart"       => \$Restart,
    "SynchroTail=s" => \$SynchroTail,
);

GetOptions(%opts); &usage() if ($Help); die "$Version\n" if $ShowVersion;
sub Print;
sub Execute;
sub ProcessData;

die ("Need a Component name, use the -Component switch") unless defined($Component);
die ("Need a Server name to connect to, use the -Server switch") unless defined($Server);
$SynchroTail="./SynchroTail/lib/SynchroTail.jar" unless defined ($SynchroTail);
die "Cannot find SynchroTail.jar, use the -SynchroTail switch (default:$SynchroTail)" unless
(-e $SynchroTail);
$Java="java" unless defined ($Java);
#die "Cannot find java, use the -java switch (default:$Java)" unless (-e $Java);
die("Need file Pattern to proceed, use the -file switch.") unless defined ($File);
$Dir="." unless defined ($Dir);
$DateCmd="date" unless defined ($DateCmd);
$Date=`$DateCmd +%Y%m%d` unless defined ($Date);chomp($Date);

$LogDir="./log/" unless defined ($LogDir);
$LogFile="$LogDir/TailExample.$Date.log" unless defined $LogFile;
open ( LOGFILE, ">$LogFile" ) || die "Cannot open file : $LogFile";

$Port=9123 unless defined ($Port);
$ServerID = new Net::Telnet (Errmode=>"return",Timeout => 10, Telnetmode => 0,
Output_record_separator => "");
$ServerID->open(Host => $Server,Port => $Port);
my $OldTime=""; my $GlobalRate=0; my $MaxRate=30000000; $Delay=0.01; #100

sub MatchExpression;
my $TmpDate;
my $TmpProcess;
my $TmpTarget;
my $TmpMetricString;

$TmpString="$Java -jar $SynchroTail lps=50000000 synchroserver=$Server serverport=$Port
logfile=$LogDir/SynchroTail.log sortby=filename basedir=$Dir filepattern=\"$File\"
RestartFileName=restart.ini RestartAllowOverlap=yes|";
$TmpString="$Java -jar $SynchroTail restart.ini|" if defined ($Restart);
printf (LOGFILE "$TmpString\n") if defined($Debug);
open ( DATAFILE, $TmpString) || die "Cannot open $TmpString";

while ($Line=<DATAFILE>){
    chomp($Line);
    ($MyDate,$Time,$ClientID,$ProductID,$Qty)=split(/;/,$Line);

    Print($MyDate, $Time,$Component,"Client",$Quantity,$ClientID,$Qty) unless
(defined($NoSend));
    Print($MyDate, $Time,$Component,"Client",$Quantity,"ALL",$Qty) unless
(defined($NoSend));

```

```

    Print($MyDate, $Time,$Component,"Product","Quantity",$ProductID,$Qty) unless
    (defined($NoSend));
    Print($MyDate, $Time,$Component,"Product","Quantity","ALL",$Qty) unless
    (defined($NoSend));
}

sub Print {
    my ($MyDate, $MyTime,$Component, $Type,$Metric,$Target,$Value)=@_;
    $TmpStr=$MyDate.".".$MyTime;

    if (!$MyTime eq $OldTime){
        Send("SC01\tinit\t$Component\n") if ($OldTime eq "");
        printf (LOGFILE "#SENDING SC01\tinit\t$Component\n")if defined
    ($Debug);

        $MetricRate=0 if ($OldTime eq "");
        Send("\n") unless ($OldTime eq "");;
        select(undef, undef, undef, $Delay);
        Send("SC01\tmstat\t$TmpStr\t");
        printf (LOGFILE "#SENDING SC01\tmstat\t$TmpStr\t\n")if defined
    ($Debug);

        $OldTime=$MyTime;
        $GlobalRate=0;
    }
    $MetricRate++;
    if ($GlobalRate > $MaxRate){
        Send("\n");
        printf (LOGFILE "#SENDING \n")if defined ($Debug);
        select(undef, undef, undef, $Delay);
        Send("SC01\tmstat\t$TmpStr\t");
        printf (LOGFILE "#SENDING SC01\tmstat\t$TmpStr\t\n")if defined
    ($Debug);

        $GlobalRate=0;
    };
    $GlobalRate++;
    my $MyTmpTarget=substr($Target,0,30);
    $MyTmpTarget=~s/[\\=\\-\\,\\;+\\ \\_]/g;
    $TmpStr=$Component."_".$Type."_".$Metric."_".$MyTmpTarget."=".$Value;
    Send("$TmpStr\t");
    printf (LOGFILE "#SENDING $TmpStr\t\n")if defined ($Debug);
}

sub Send{
    my ($SendTmpString)=@_;
    $ServerID->print("$SendTmpString");
    $Status=$ServerID->errmsg;
    while (! ($Status eq "")){
        sleep(1);
        print ("CONNECTION ERROR -retrying...\n");
        printf (LOGFILE "CONNECTION ERROR -retrying...\n");
        $ServerID->close(Host => $Server,Port => $Port);
        $ServerID->open(Host => $Server,Port => $Port);
        $ServerID->print("SC01\tinit\t$Component\n");
        $Status=$ServerID->errmsg;
        if ($Status eq ""){
            print ("CONNECTION OK.\n");
            printf (LOGFILE "CONNECTION OK\n");
            $ServerID->print("$SendTmpString");
            $Status=$ServerID->errmsg;
        };
    }
}
}

```

Grouping of all clients and products activity also creates a second by second overall activity by clients and products id done by the expression below. Instead of having the script sending another transaction, this grouping could have also been done by the SynchroServer using “Time Series Mapping”.

```
Print($MyDate, $Time,$Component,"Product","Quantity","ALL",$Qty) unless (defined($NoSend));
```

SCRIPT

```
#!/usr/bin/perl -w
use Getopt::Long;

my %opts = (
    "Date=s"      => \$Date,
    "NBClient=s"  => \$NBClient,
    "NBProduct=s" => \$NBProduct,
    "MaxQty=s"    => \$MaxQty,
    "Rate=s"      => \$Rate,
);
0
GetOptions(%opts);

$DateCmd="date" unless defined ($DateCmd);
$Date=`$DateCmd +%Y%m%d` unless defined ($Date);chomp($Date);

$NBClient=10 unless defined ($NBClient);
$NBProduct=100 unless defined ($NBProduct);
$MaxQty=1000 unless defined ($MaxQty);
$Rate=1 unless defined ($Rate);
$Separator=",";

for ($Hour=0;$Hour<24;$Hour++){
$FileString=sprintf("LogFile-$Date-%02d.log", $Hour);
    open ( LOGFILE, ">$FileString" ) || die "Cannot open file : $FileString";

        for ($Min=0;$Min<60;$Min++){
            for ($Sec=0;$Sec<60;$Sec++){
                for ($It=0;$It<$Rate;$It++){
                    $HourStr = sprintf("%02d%02d%02d", $Hour, $Min, $Sec);
                    $ClientStr=sprintf("%05d", int(rand($NBClient)));
                    $ProductStr=sprintf("%05d", int(rand($NBProduct)));
                    $QtyStr=sprintf("%05d", int(rand($MaxQty)));
                    printf(LOGFILE
"$Date$Separator$HourStr$Separator$ClientStr$Separator$ProductStr$Separator$QtyStr\n");
                }
            }
        }
    close(LOGFILE);
}

./ExampleFileCreation.pl -rate 2 -nbclient 1 -nbprod 1000 -date 20120922
ls -tr *.log|head
LogFile-20120922-00.log
LogFile-20120922-01.log
LogFile-20120922-02.log
head LogFile-20120922-02.log
20120922;020000;00000;00588;00190
20120922;020000;00000;00151;00259
20120922;020001;00000;00754;00003
20120922;020001;00000;00310;00796
20120922;020002;00000;00050;00588

#!/usr/bin/perl -w

use English;
use Getopt::Long;
use IO::Handle;
use Net::Telnet ();

my $Version="Version 0.0.3";

my %opts = (
    "help"          => \$Help,
    "version"       => \$ShowVersion,
    "Dir=s"         => \$Dir,
    "Date=s"        => \$Date,
    "Server=s"      => \$Server,
    "Port=s"        => \$Port,
```



```

"Component=s"          => \$Component,
"Java=s"              => \$Java,
"LogDir=s"            => \$LogDir,
"LogFile=s"           => \$LogFile,
"File=s"              => \$File,
"Debug"               => \$Debug,
"Restart"             => \$Restart,
"SynchroTail=s"      => \$SynchroTail,

);

GetOptions(%opts); &usage() if ($Help); die "$Version\n" if $ShowVersion;
sub Print;
sub Execute;
sub ProcessData;

die ("Need a Component name, use the -Component switch") unless defined($Component);
die ("Need a Server name to connect to, use the -Server switch") unless defined($Server);
$SynchroTail="./SynchroTail/lib/SynchroTail.jar" unless defined ($SynchroTail);
die "Cannot find SynchroTail.jar, use the -SynchroTail switch (default:$SynchroTail)" unless
(-e $SynchroTail);
$Java="java" unless defined ($Java);
#die "Cannot find java, use the -java switch (default:$Java)" unless (-e $Java);
die("Need file Pattern to proceed, use the -file switch.") unless defined ($File);
$Dir="." unless defined ($Dir);
$DateCmd="date" unless defined ($DateCmd);
$Date=`$DateCmd +%Y%m%d` unless defined ($Date);chomp($Date);

$LogDir="./log/" unless defined ($LogDir);
$LogFile="$LogDir/TailExample.$Date.log" unless defined $LogFile;
open ( LOGFILE, ">$LogFile") || die "Cannot open file : $LogFile";

$Port=9123 unless defined ($Port);
$ServerID = new Net::Telnet (Errmode=>"return",Timeout => 10, Telnetmode => 0,
Output_record_separator => "");
$ServerID->open(Host => $Server,Port => $Port);
my $OldTime=""; my $GlobalRate=0; my $MaxRate=30000000; $

my $TmpDate;
my $TmpProcess;
my $TmpTarget;
my $TmpMetricString;

$TmpString="$Java -jar $SynchroTail lps=50000000 synchroserver=$Server serverport=$Port
logfile=$LogDir/SynchroTail.log sortby=filename basedir=$Dir filepattern=\"$File\"
RestartFileName=restart.ini RestartAllowOverlap=yes|";
$TmpString="$Java -jar $SynchroTail restart.ini|" if defined ($Restart);
printf (LOGFILE "$TmpString\n") if defined($Debug);
open ( DATAFILE, $TmpString) || die "Cannot open $TmpString";
print("$TmpString\n");

while ($Line=<DATAFILE>){
    chomp($Line);
    #Date(YYYYMMDD);Hour(HHMMSS);ClientID;ProductID;Qty
    ($MyDate,$Time,$ClientID,$ProductID,$Qty)=split(/;/,$Line);
    #printf( LOGFILE "###$MyDate,$Time,$ClientID,$ProductID;$Qty\n")if defined ($Debug);

    Print($MyDate, $Time,$Component,"Client","Quantity",$ClientID,$Qty) unless
(defined($NoSend));
    Print($MyDate, $Time,$Component,"Client","Quantity","ALL",$Qty) unless
(defined($NoSend));
    Print($MyDate, $Time,$Component,"Product","Quantity",$ProductID,$Qty) unless
(defined($NoSend));
    Print($MyDate, $Time,$Component,"Product","Quantity","ALL",$Qty) unless
(defined($NoSend));
}

sub Print {
    my ($MyDate, $MyTime,$Component, $Type,$Metric,$Target,$Value)=@_;
    $TmpStr=$MyDate.".".$MyTime;

```

```

        if (!(MyTime eq OldTime)){
            Send("SC01\tinit\t$Component\n") if (OldTime eq "");
            printf (LOGFILE "#SENDING SC01\tinit\t$Component\n")if
defined ($Debug);
            $MetricRate=0 if (OldTime eq "");
            Send("\n") unless (OldTime eq "");;
            select(undef, undef, undef, $Delay);
            Send("SC01\tmstat\t$TmpStr\t");
            printf (LOGFILE "#SENDING SC01\tmstat\t$TmpStr\t\n")if
defined ($Debug);
            OldTime=MyTime;
            GlobalRate=0;
        }
        $MetricRate++;
        if ($GlobalRate > $MaxRate){
            Send("\n");
            printf (LOGFILE "#SENDING \n")if defined ($Debug);
            select(undef, undef, undef, $Delay);
            Send("SC01\tmstat\t$TmpStr\t");
            printf (LOGFILE "#SENDING SC01\tmstat\t$TmpStr\t\n")if defined
($Debug);
            GlobalRate=0;
        };
        $GlobalRate++;
        my $MyTmpTarget=substr($Target,0,30);
        $MyTmpTarget=~s/[\=\\-\,\,;\+\ \_]/g;
        $TmpStr=$Component."_".$Type."_".$Metric."_".$MyTmpTarget."=".$Value;
        Send("$TmpStr\t");
        printf (LOGFILE "#SENDING $TmpStr\t\n")if defined ($Debug);
    }
    sub Send{
        my ($SendTmpString)=@_;
        $ServerID->print("$SendTmpString");
        $Status=$ServerID->errmsg;
        while (! ($Status eq "")){
            sleep(1);
            print ("CONNECTION ERROR -retrying...\n");
            printf (LOGFILE "CONNECTION ERROR -retrying...\n");
            $ServerID->close(Host => $Server,Port => $Port);
            $ServerID->open(Host => $Server,Port => $Port);
            $ServerID->print("SC01\tinit\t$Component\n");
            $Status=$ServerID->errmsg;
            if ($Status eq ""){
                print ("CONNECTION OK.\n");
                printf (LOGFILE "CONNECTION OK\n");
                $ServerID->print("$SendTmpString");
                $Status=$ServerID->errmsg;
            };
        }
    }
}

```

Appendix C. SynchroServer Performance Tuning

To improve SynchroServer performance, use the following approaches:

C.1 Increase Disk Performance

This approach has proven itself to be the most effective approach for improving SynchroServer performance.

Synchro stores raw data it receives in binary sparse files in the directory <SynchroServer/data/metrics>. These sparse files can be changed any time. Data can be inserted “out of order” from the time perspective.

Compression of data, runs daily to limit the amount of disk used by the raw sparse file. Once compressed, the data becomes read only (cannot be changed). These compressed files are stored in the directory <SynchroServer/data/metrics.gz>.

FOR LARGE SYNCHRO CONFIGURATIONS

Synchro does a significant amount of small IO in numerous sparse files that take a relatively small amount of disk space.

Note: It is a recommended practice to set these files on fast disks (SSD drives are perfect for this job).

The compressed data (read only) can take significantly more space, depending on the retention policy and is less IO intensive.

Note: It is a recommended practice to set this data on slower large disks infrastructure.

Synchro Administrators can easily change the location of the directories using symbolic links.

C.2 Increase CPU Usage

SynchroServer is a multi-threaded Java application. By default there are four (4) threads that process incoming statistics and four (4) threads that process incoming queries on the data. It is possible to configure the number of these threads to increase performance. Parameters available in “SynchroAdmin/Server/Internal Configuration” menu are:

Merge Engine Count	Number of internal engines that process incoming Statistics from clients and probes.
Result Engine Count	Number of internal engines that process incoming HTTP queries from SynchroWEB interfaces.

C.3 Increase Memory Available

The amount of memory available to SychroServer is fixed at start time. By default, SynchroServer will not use more than 60% of available physical memory available on the host running the server. This can be changed by fixing the parameter "Memory Allocation" in the "SynchroAdmin"/"Server/Server" menu. (Unit Megabyte)