POC – First weeks benefits

28.02.2016
By Tidhar Seifer

1. Scope

This document describes the benefits that ConicIT system gives during the first 3 weeks, before proactive alerts are activated.

2. Introduction

During the first few weeks of ConicIT installation, the system is not yet capable of producing proactive alerts (anomaly alerts) based on statistical models and algorithms, simply because there's not enough history to base the "normal-behavior" model upon.
This might bring us to the false conclusion that ConicIT benefits starts only 3 weeks post installation. After all the proactive alerts that are based on unique statistical algorithms is very powerful.
However, ConicIT actually does have important advantages that it provides from the very first moment after installation and configuration:

- ConicIT can aggregate data from different sources and present them in a combined way.
- ConicIT can produce calculated variables which are hard to calculate manually and can provide important information.
- ConicIT can send static-alerts, which are not based on statistics. So these alerts are not based on the ability of ConicIT to study the normal behavior, but they can still provide important alerts, and provide helpful information for analyzing the alerts.

We'll describe some of these advantages within this document.

3. Data aggregation and presentation

The first advantage that ConicIT provides is a centralized and efficient way to observe all data from all monitors on one web-site.

1. Data Aggregation

ConicIT can aggregate information for example from different CICSes, even from different monitors, and show the results on a single screen. ConicIT provides a single interface with single screen from which you can see information from different monitors and different type of data – from CICS, from MVS, from DB2, from TSO, from WebSphere, from storage, from files, and even from external sources like Splunk™ and from results of Linux-scripts that brings external information (e.g. from some database or web-service).

Summary table for Groups of CICSes (Comparables)

Systems	1CPU %	MIPS	Tran rate	LPAR
Dynamic_SYSB	139.57	1783	29724.6	SYSB
Static_SYSB	50.74	647	16149.4	SYSB
TOR_SYSB	8.77	111	10089.2	SYSB
ALL_SYSB_CICSes	199.08	2541	55963.2	SYSB
Dynamic_SYSG	155.04	2095	37461.5	SYSG
Static_SYSG	-	-	-	SYSG
TOR_SYSG	14.96	200	30846.4	SYSG
ALL_SYSG_CICSes	170	2295	68307.9	SYSG
Dynamic_ALL		3878	67186.1	**ALL**
Static_ALL		647	16149.4	**ALL**
TOR_ALL		311	40935.6	**ALL**
[AOR_ALL]		4525	83335.5	**ALL**
ALL_CICSes		4836	124271.1	**ALL**

Figure 1: Aggregating of CICS information from different monitors (and in this case also cumulating their values based on CICSes-groups).

Degredation Analysis - Average Resources %Wait Time

Resource Type_Name	TOR_ALL	Dynamic_ALL	TOR_SYSB	TOR_SYSG	Dynamic_SYSB	Dynamic_SYSG
TCLASS_TOTAL	6.75	-	6.75	6.75	-	-
USERWAIT_TOTAL	38.125	38.875	37.75	38.5	38.75	39
EKCWAIT_SINGLE	-	11	-	-	11	11
MQSeries_GETWAIT	6.5	-	6.5	6.5	-	-

Figure 2: Aggregation of %Wait time for resources-types, based on calculation of information read from many screens from different CICSes.

2. Concurrent 24/7 data collection

ConicIT is gathering thousands of metrics per minute, and they're all available for you within a single-click distance. You don't need to switch between monitors and do tedious login and to type long commands to get the performance information that you need. ConicIT is doing all of that for you, and the data is always available for you through your private ConicIT Web-site. It makes it much easier to see all relevant information within few seconds.

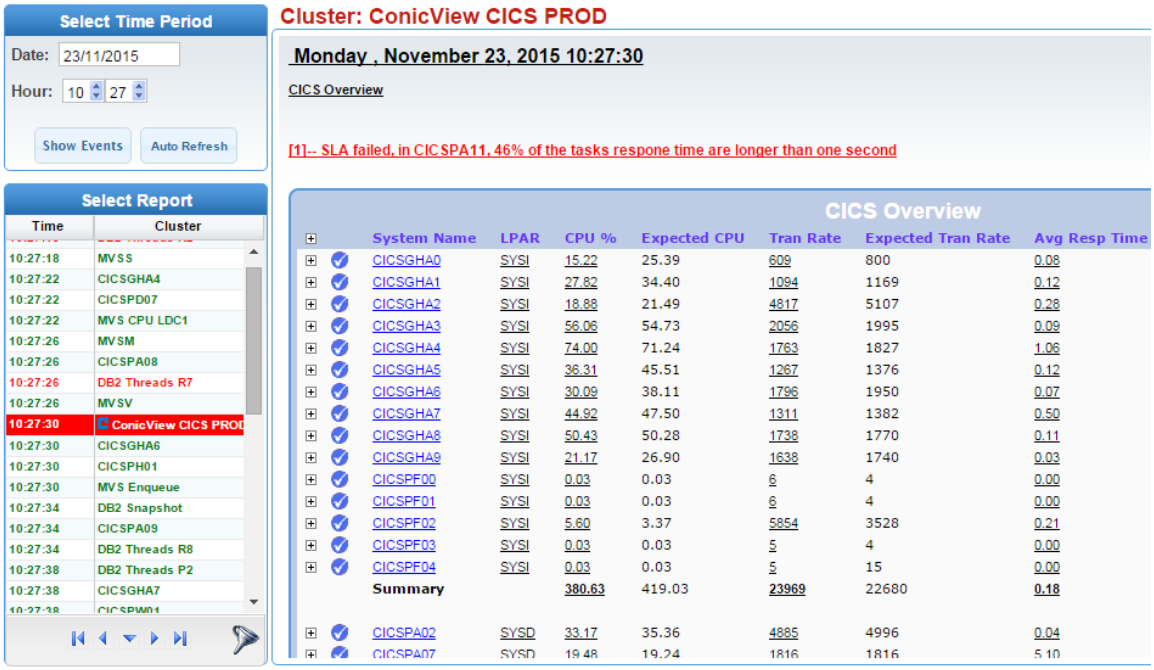


Figure 3: It takes a single click (on the left menu) to switch and view any type of information from any point in time

3. History and graphs

ConicIT is keeping two months of history for the detailed investigation of each screen, and keeping one year of graphs-data. It means that you can easily investigate trends, and track changes in your system. In addition, when investigating a performance problem, you can check the graphs and the investigations of the last few hours. It can help you to realize when the problem started and what which the first symptoms. Finding the first symptoms often helps a lot in identifying the source of the problem, and separating between cause and effects.

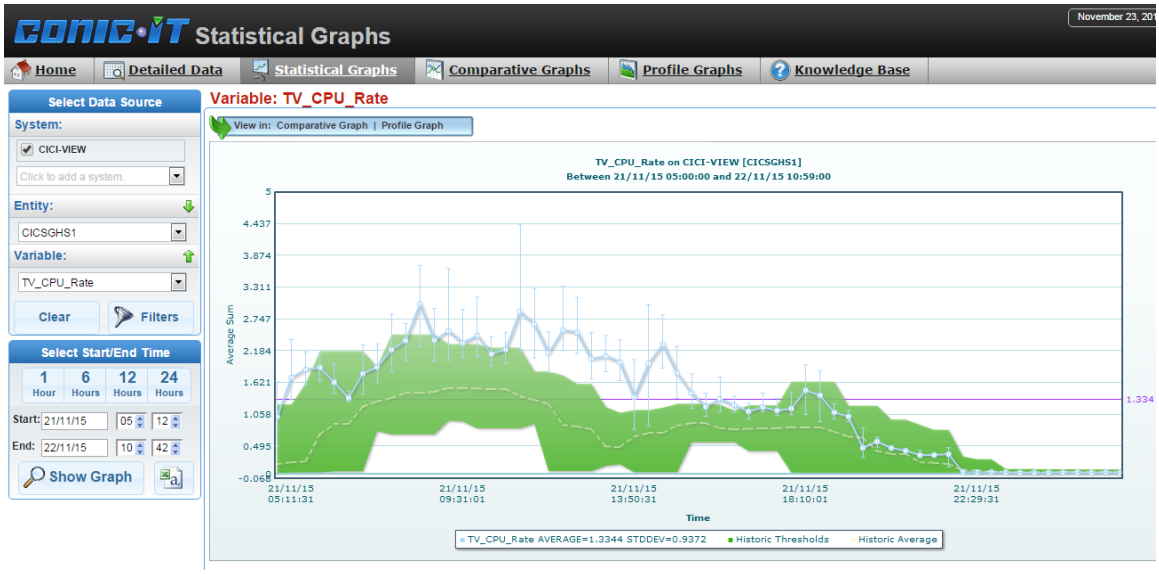


Figure 4: 30 hours graph

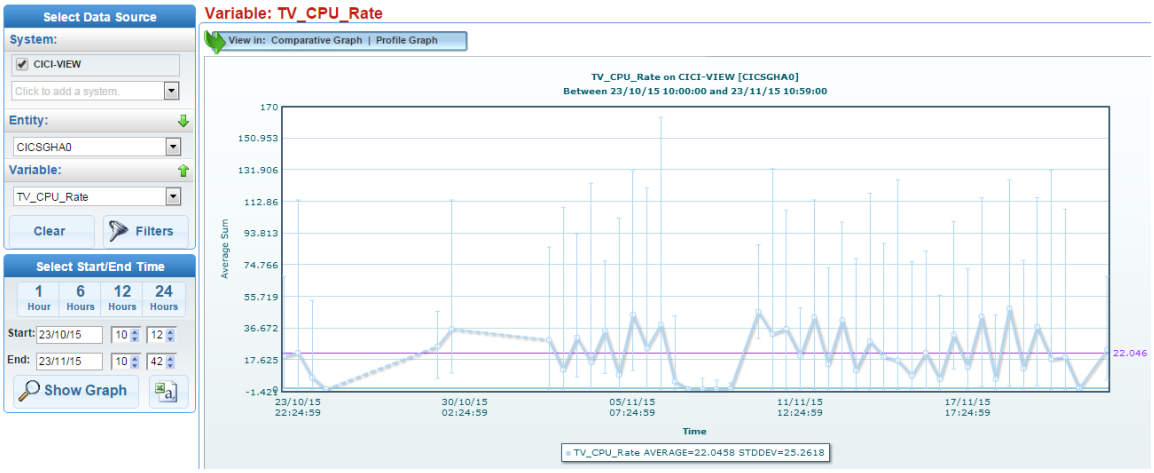
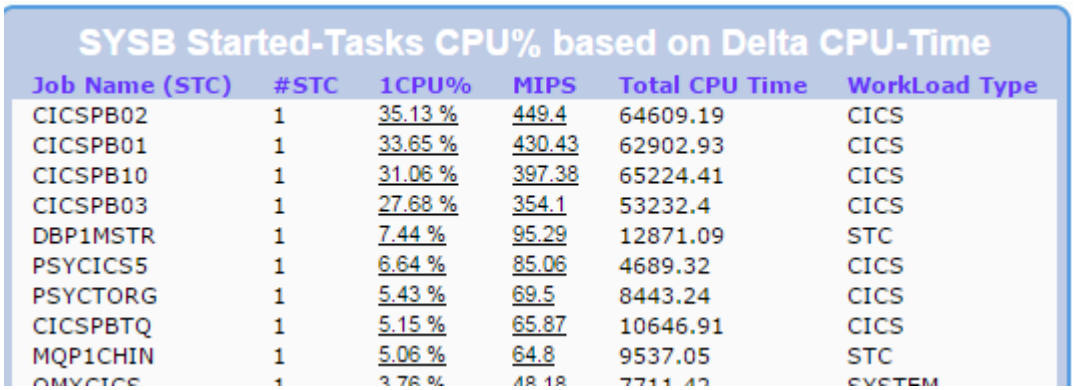


Figure 5: One month graph

4. Calculated Variables

One of the great advantages of ConicIT is the ability to easily define calculated variables, present it, and analyze it for alerts.
Calculated variables can be variables calculated based on different sources of information, or variables that are ratio between other variables, or variables that are delta of some variables between the iteration.
For example, the monitor provides us only the cumulating CPU-time of each started-task. So the only way to know the real-time CPU is by calculating the delta of the CPU-time between each minute and the one before, and thus getting the real-time CPU consumption:



SYSB Started-Tasks CPU% based on Delta CPU-Time

Job Name (STC)	#STC	1CPU%	MIPS	Total CPU Time	WorkLoad Type
CICSPB02	1	35.13 %	449.4	64609.19	CICS
CICSPB01	1	33.65 %	430.43	62902.93	CICS
CICSPB10	1	31.06 %	397.38	65224.41	CICS
CICSPB03	1	27.68 %	354.1	53232.4	CICS
DBP1MSTR	1	7.44 %	95.29	12871.09	STC
PSYCICS5	1	6.64 %	85.06	4689.32	CICS
PSYCTORG	1	5.43 %	69.5	8443.24	CICS
CICSPBTQ	1	5.15 %	65.87	10646.91	CICS
MQP1CHIN	1	5.06 %	64.8	9537.05	STC
OMXCICS	1	3.76 %	48.18	7711.42	SYSTEM

Figure 6: Real time CPU-consumption based on the delta CPU-time from previous minute, for each job-name. In addition we created a MIPS which is also a calculated variable, based on the calculated %CPU and the total MIPS of the CPU.

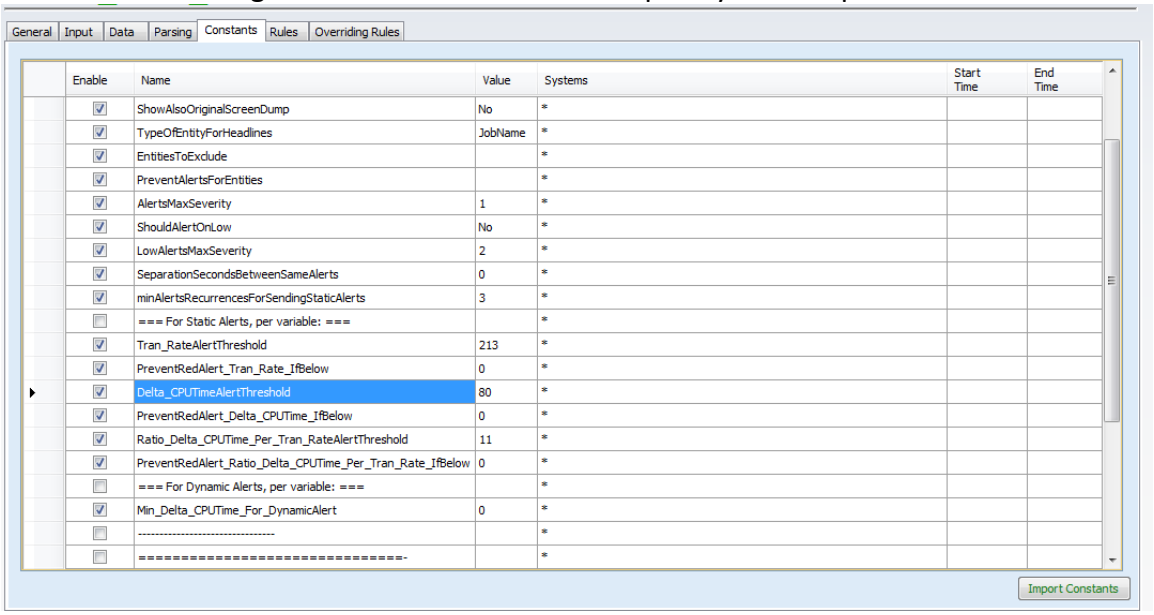
Another examples are in figure 1 and figure 2, where the presented values are the summary of few CICSes for each variable.

5. Static Alert

Even that during the first three week ConicIT is not generating dynamic (proactive) alerts, **it does send static-alerts**, based on static thresholds and based on non-statistical conditions. Static alerts can protect the system from extreme situations, based on the definitions of them. The most basic static alerts are just alerts that are sent if some variable is above some value. But ConicIT provides much more, even for static alerts, it allows:

- Setting different thresholds for different hours
- Setting different thresholds for different system for the same variable (e.g. for transaction-rates to set different thresholds for different CICSes)

- **Combining time and value**: it's possible to set alert only if some variable is too high for more than X minutes. This way it's possible to prevent false alerts for short peaks, and send the alert only if a bad situation exists for enough time.
- It's **possible to set alerts on calculated variables**, so for example to set alert on real-time CPU of started-task (even that the monitor doesn't have such variable, it only has total-CPU-time from IPL). Another example is to set static alert on the summary of the CPU of a cluster of CICSes that are working in PLEX.
- It's possible to define different severity of alerts – based on time and based on value
- It's **possible to create complex condition for alerts**, e.g. based on few conditions on different variables.
- It's possible to define the frequency of the alerts in case we have a continuous alert situation. E.g. to send the alert at max frequency of once per hour.



Enable	Name	Value	Systems	Start Time	End Time
☑	ShowAlsoOriginalScreenDump	No	*		
☑	TypeOfEntityForHeadlines	JobName	*		
☑	EntitiesToExclude		*		
☑	PreventAlertsForEntities		*		
☑	AlertsMaxSeverity	1	*		
☑	ShouldAlertOnLow	No	*		
☑	LowAlertsMaxSeverity	2	*		
☑	SeparationSecondsBetweenSameAlerts	0	*		
☑	minAlertsRecurrencesForSendingStaticAlerts	3	*		
☐	=== For Static Alerts, per variable: ===		*		
☑	Tran_RateAlertThreshold	213	*		
☑	PreventRedAlert_Tran_Rate_IfBelow	0	*		
☑	Delta_CPUTimeAlertThreshold	80	*		
☑	PreventRedAlert_Delta_CPUTime_IfBelow	0	*		
☑	Ratio_Delta_CPUTime_Per_Tran_RateAlertThreshold	11	*		
☑	PreventRedAlert_Ratio_Delta_CPUTime_Per_Tran_Rate_IfBelow	0	*		
☐	=== For Dynamic Alerts, per variable: ===		*		
☑	Min_Delta_CPUTime_For_DynamicAlert	0	*		
☐	----------		*		
☐	==========-		*		

Figure 7: The configuration screen for alerts behavior allows great flexibility.