Statistics and Machine Learning Techniques for Real End-User Experience

Würth Phoenix S.r.l.
Via Kravogl, 4
I-39100 Bolzano
Phone +39 0471 564111
E-Mail: info@wuerth-phoenix.com

Abstract: *Real End-User Experience (RUE) is a monitoring approach that aims to measure the end-user experience by providing information on availability, response time, and reliability of the real used IT services. The response time of each user transaction is measured by an analysis of the network communication flows. Several performance metrics get archived to monitor RUE over time.*
An abstract, generalized view of performance over time is of advantage before digging into data. We explored how advanced statistics and machine learning techniques can be used as effective tools to bring registered data to the desired level of abstraction. The resulting high-level visualizations help to get the big picture of what is happening within a network under investigation and improve our understanding of application performance across the network.

Keywords*: real end-user experience, statistics, machine learning, NPM, APM*

Real User Experience (RUE) is a monitoring approach that aims to measure the end-user experience by providing information on availability, response time, and reliability of IT services such as for example ERP, email, and applications. *Application performance monitoring* (APM) is often based on *performance metrics* measured by RUE such as *impacted users*, *client network latency*, *server network latency*, *application server latency*, *throughput* and others. They are used to identify problems that might negatively influence the way of how end users perceive application quality. Furthermore RUE plays an increasing role for *network performance monitoring* (NPM). For example wrong router configurations, wrong network card configurations or network infrastructure problems, just to mention a few, can be comfortably analyzed into detail by the RUE Monitoring tool. Common goal of APM and NPM is to ensure user satisfaction by detecting problems as soon as they occur to resolve any potential issues before the end-user gets aware of them.
While a very *detailed view* of as many ***key performance indicators*** (KPIs) as possible is an important step for getting towards the solution of a specific problem, a more *abstract, generalized view* of network or application performance and end user quality perception might be of advantage at an earlier stage before digging into data. ***Advanced statistics*** and ***machine learning techniques*** can be used as effective tools to bring registered data to the desired level of abstraction [1]. Let us start with a short introduction to probability densities and unsupervised learning.

A **probability density function** of a continuous random variable, is a function that describes the relative likelihood for this random variable to take on a given value [2, 3]. The probability of the random variable falling within a particular range of values is then given by the area under the density function between the lowest and greatest values of the range. The probability density function is nonnegative everywhere, and its integral over the entire space is equal to one. How can a probability density help to characterize network or application performance? Within an hour, a day, a week or even longer periods of time each performance metric reaches various values. Most of them are going to fall within certain ranges of values that characterize the base traffic related to the application or network or interest.
In the best-case scenario values that lie far away from those registered during base activity are related to non-standard activity and therefore irregular traffic or potential network problems. The estimation of a probability density function and definition of base regions for each performance

indicator might lead to the improvement of (critical) warning quality, if some of the performance metrics of the network or application under investigation do not have a distinct, single local maximum (Illustration 1, subnet 2).

Let us use the following example to make things clearer. Illustration 1 shows the estimated probability density function of the throughput of two different subnets. In case of subnet 1 the mean lies close to the only prominent local maximum. For this subnet (critical) warnings based on the data mean are reasonable.

The estimated probability density function of subnet 2 instead has three distinct local maxima. The data mean falls in the region between two of the maxima. In a worst case scenario this might lead to standard traffic producing false warnings and non standard traffic occurring without any warning. For such settings warnings based on the local maxima of the probability distribution might be a reasonable alternative to merely mean-based ones.

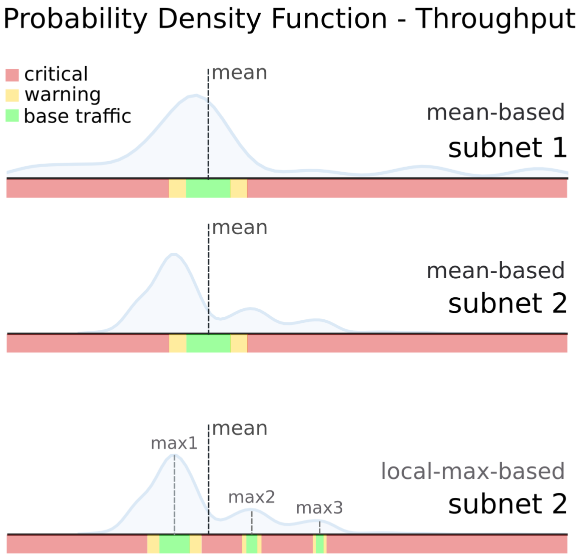


Illustration 1: Different concepts as basis for warnings

Unsupervised learning tries to solve the problem of finding hidden structure in unlabeled data [4, 5]. It is closely related to the problem of density estimation in statistics. A common unsupervised learning approach is cluster analysis, i.e. the task of grouping data in such a way that data in the same group are more similar to each other than to those in other groups. In density-based clustering those groups are defined as areas of higher density than the remainder of the data set. Queries from normal network activity can be assumed to be closer or more similar than queries from random activity or during network problems. For this region a density-based cluster analysis of network data might bring the advantage than normal, standard activity can be separated from irregular traffic such as activity caused by network problems. Multi-dimensional regions of *dense* traffic can be defined as *base regions*, while one might want to perform a detailed analysis of *sparse* traffic regions, especially where a sparse region shows bad values of the performance metrics.

The following example is meant to illustrate the advantage of multi-dimensional base regions:

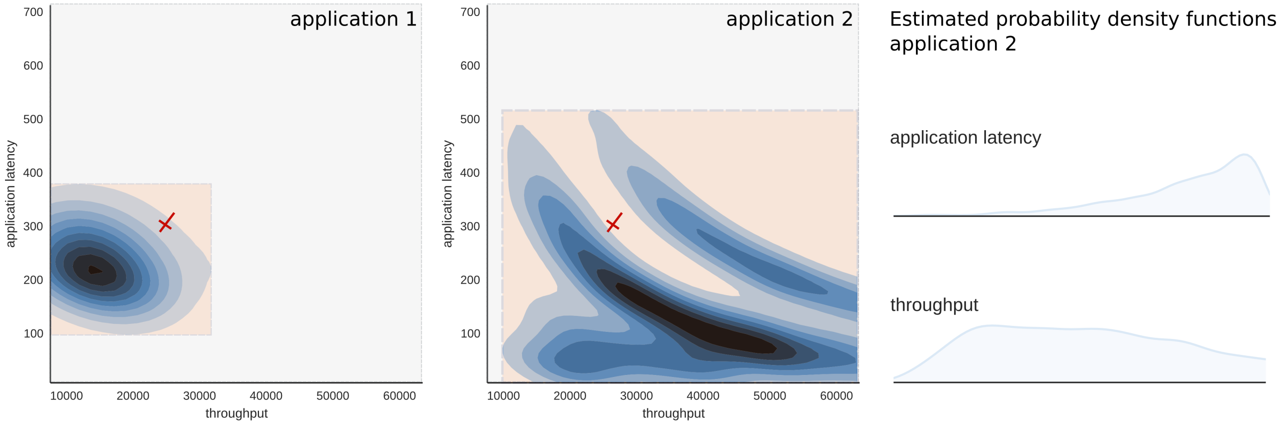


Illustration 2: Base regions in multi-dimensional performance metric space

For the sake of simplicity we limit us to two dimensions, namely the application latency and the throughput as performance metrics. Application 1 – on the left – has a well defined range in both dimensions. Throughput values fall into the interval between 0B/s and 30kB/s with a peak around

15 kB/s. The application latency lies between 100ms and 380ms with a single global maximum at 220ms. Warning and alerts based on the mean can be expected to work well for both performance indicators. This is not the case for application 2. Even the local maxima of the probability density are not sufficient in this case because there are no distinct ones. The regions where activity can be expected to be dense cannot be defined for each single performance metric, but they can be defined via density-based supervised learning algorithms in the multi-dimensional space that contains all performance metrics. For example imagine a query falling in the orange-shaded sparse activity region at the center of the density plot of application 2. It would be close the center of the range of common values for the throughput and close to the center of the range of common application latencies. Nonetheless the very combination of values for the throughput and application latency is not common at all, given that the region has not be detected as dense traffic. Warnings based on one-dimensional metric ranges have no way of marking this query as irregular traffic.

After this dive into the world of statistics and machine learning let us explore how the concepts introduced above can contribute to bring an enormous amount of data to the level of abstraction necessary for network or application performance control at high level and step by step drill downs. Networks with complex infrastructure and many users are not straight forward to analyze. For example when end-users report the network to be slow as the very first step, before any detailed analysis, one should control the current performance. The values of several performance metrics need therefore to be considered and in particular their change over time. The analysis of the probability density function of the registered data and significant changes to it over time are a comfortable way to check whether there are potential performance issues even visible at high level. Furthermore this kind of analysis can also be used to compare network performance before and after changes to network settings or the installation of new hardware. Also a significant change to the ratio between dense and sparse data can indicate network anomalies. A more detailed analysis of the queries falling into sparse regions brings us a step nearer towards the causes of such traffic.

We are currently working on a new graphical tool that allows us to visualize a high level ***performance trend*** (PT) which can be used for both APM and NPM. The PT uses the concepts described above to provide the owner of RUE Monitoring tool with highly informative contour plots that give a high level picture of several performance metrics and their changes within a specified time period. We simulated throughput values with three local maxima at 4 kB/s, 6 kB/s and 11 kB/s for three different periods. Period 1 – on the left – can be seen as a standard day of network activity. Most of the traffic has throughput values around the maximum at 6 kB/s. The traffic is stable for the entire period under investigation. Lighter color indicates a higher frequency of traffic at that particular value. Vertical areas form when traffic is constant over time. Less regular traffic produces dots and smaller regions instead.
The next simulation (period 2) depicts a case where network performance decreased with respect to period 1. More traffic has a smaller throughput around 4 kB/s and the traffic at 6 kB/s is still constant but with much lower frequency.
Period 3 on the right instead is the visualization of traffic that is less constant than the main traffic of period 1, but most of it has a notably higher throughput at 11 kB/s. On the one hand if period 1 corresponded to network traffic before changes to existing hardware and period 3 corresponded to network traffic after intervention, than both visualizations together could be used to show the effectiveness of the changes. On the other hand if period 2 was the visualization of network traffic after some changes to network parameters one could use the visualizations of period 1 and 2 to show that the new setting has a negative impact on network or application performance.

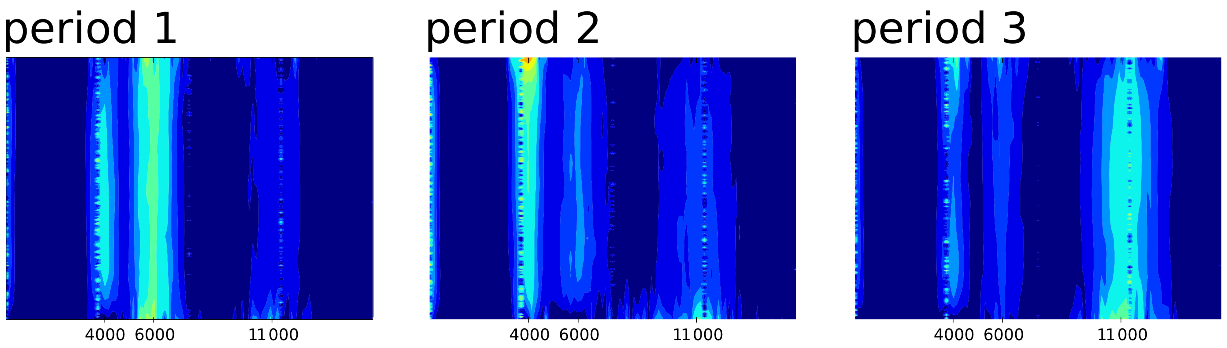
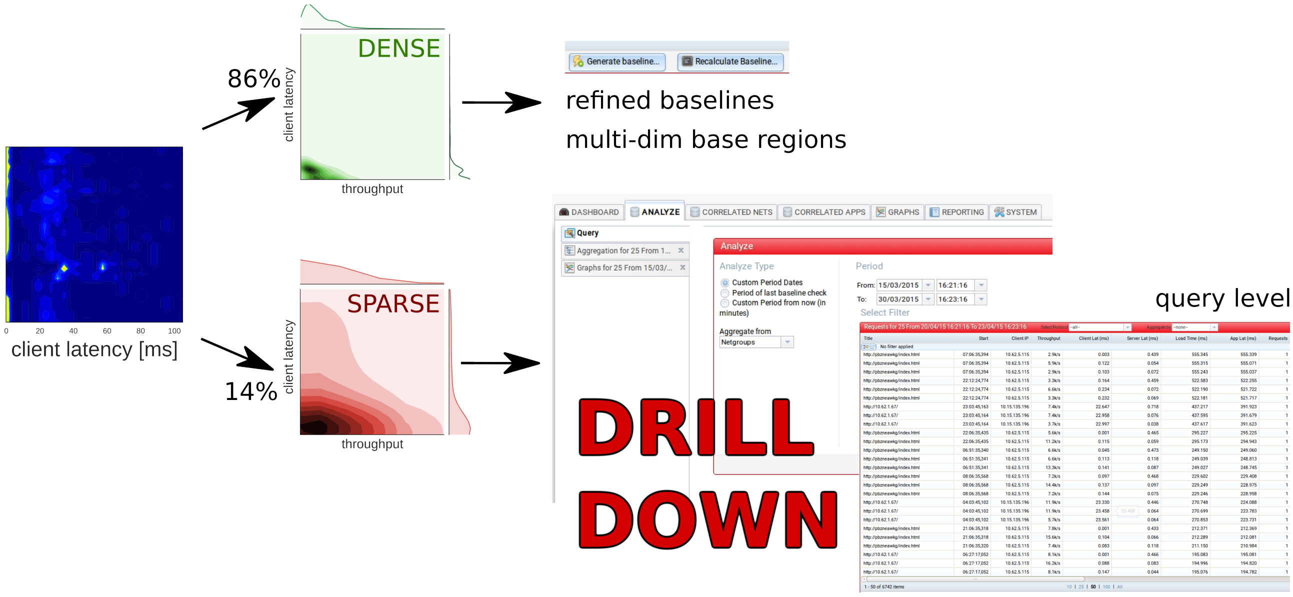


Illustration 3: Network performance over time: simulations

One possible way to use the new advanced statistics and machine learning features is explained in the following. For example there might be some users within a company complaining about abysmal performance of a specific application during a single working day. A fast network performance check of the traffic created by the application of interest during the working day in question might be able to show that there are no anomalies visible on high level beside a few blobs far from standard traffic for the client latency. After a quick sigh of relief one might want to go one step further and apply the density-based unsupervised learning analysis to find out that 86% of the queries of that day are densely concentrated within a small range while 14% of the queries have been detected as sparse traffic. The dense standard traffic might be used as basis for the calculation of refined baselines or even multi-dimensional base areas. The sparse traffic instead is the one that most probably contains most information about the potential problem and those queries responsible for the suspect blobs. For this reason a first dig into data might be restricted to an investigation of sparse traffic e.g. in the form of a drill down to query level. A closer study of the sparse traffic might reveal precious hints about the causes of the non-standard traffic. It is convenient to analyze the 14% of data that is most probably related to potential problems first, as some applications might have tens of thousands of queries a day.



Analyze Type

- ● Custom Period Dates
- ○ Period of last baseline check
- ○ Custom Period from now (in minutes)

Aggregate from: Netgroups

Period

From: 15/03/2015 16:21:16

To: 30/03/2015 16:23:16

Select Filter

Requests for 25 From 20/04/15 16:21:16 To 23/04/15 16:23:16 Select Protocol --all-- Aggregate by --none--

No filter applied

Title	Start	Client IP	Throughput	Client Lat (ms)	Server Lat (ms)	Load Time (ms)	App Lat (ms)	Requests
http://pbzneawkg/index.html	07:06:35,394	10.62.5.115	2.9k/s	0.003	0.439	555.345	555.339	1
http://pbzneawkg/index.html	07:06:35,394	10.62.5.115	5.9k/s	0.122	0.054	555.315	555.071	1
http://pbzneawkg/index.html	07:06:35,394	10.62.5.115	2.9k/s	0.103	0.072	555.243	555.037	1
http://pbzneawkg/index.html	22:12:24,774	10.62.5.115	3.3k/s	0.164	0.459	522.583	522.255	1
http://pbzneawkg/index.html	22:12:24,774	10.62.5.115	6.6k/s	0.234	0.072	522.190	521.722	1
http://pbzneawkg/index.html	22:12:24,774	10.62.5.115	3.3k/s	0.232	0.069	522.181	521.717	1
http://10.62.1.67/	23:03:45,163	10.15.135.196	7.4k/s	22.647	0.718	437.217	391.923	1
http://10.62.1.67/	23:03:45,164	10.15.135.196	7.4k/s	22.958	0.076	437.595	391.679	1
http://10.62.1.67/	23:03:45,164	10.15.135.196	3.7k/s	22.997	0.038	437.617	391.623	1
http://pbzneawkg/index.html	22:06:35,435	10.62.5.115	5.6k/s	0.001	0.465	295.227	295.225	1
http://pbzneawkg/index.html	22:06:35,435	10.62.5.115	11.2k/s	0.115	0.059	295.173	294.943	1
http://pbzneawkg/index.html	06:51:35,340	10.62.5.115	6.6k/s	0.045	0.473	249.150	249.060	1
http://pbzneawkg/index.html	06:51:35,341	10.62.5.115	6.6k/s	0.113	0.118	249.039	248.813	1
http://pbzneawkg/index.html	06:51:35,341	10.62.5.115	13.3k/s	0.141	0.087	249.027	248.745	1
http://pbzneawkg/index.html	08:06:35,568	10.62.5.115	7.2k/s	0.097	0.468	229.602	229.408	1
http://pbzneawkg/index.html	08:06:35,568	10.62.5.115	14.4k/s	0.137	0.097	229.249	228.975	1
http://pbzneawkg/index.html	08:06:35,568	10.62.5.115	7.2k/s	0.144	0.075	229.246	228.958	1
http://10.62.1.67/	04:03:45,102	10.15.135.196	11.9k/s	23.330	0.446	270.748	224.088	1
http://10.62.1.67/	04:03:45,102	10.15.135.196	11.9k/s	23.458	0.064	270.699	223.783	1
http://10.62.1.67/	04:03:45,102	10.15.135.196	5.7k/s	23.561	0.064	270.853	223.731	1
http://pbzneawkg/index.html	21:06:35,318	10.62.5.115	7.8k/s	0.001	0.433	212.371	212.369	1
http://pbzneawkg/index.html	21:06:35,318	10.62.5.115	15.6k/s	0.104	0.066	212.289	212.081	1
http://pbzneawkg/index.html	21:06:35,320	10.62.5.115	7.4k/s	0.083	0.118	211.150	210.984	1
http://10.62.1.67/	06:27:17,052	10.62.5.115	8.1k/s	0.001	0.466	195.083	195.081	1
http://10.62.1.67/	06:27:17,052	10.62.5.115	16.2k/s	0.088	0.083	194.996	194.820	1
http://10.62.1.67/	06:27:17,052	10.62.5.115	8.1k/s	0.147	0.044	195.076	194.782	1

1 - 50 of 6742 items 10 | 25 | 50 | 100 | All

Illustration 4: Concept of usage – performance trends and machine learning features

Does theory hold for real-data scenarios? First tests on real data make us optimistic that probability density over time is a useful addition to query-based analysis, as the abstract network traffic overview can give a better feeling for long term trends and precious hints on where to start a more detailed analysis. A detailed use case can be found in 🗎 **Use Case 1 - 2015**.