Lucierna
by SMARTBEAR

Lucierna Accepts No Compromises

Lucierna has taken a **step forward** in Application Performance Management.

Until now there have only been two approaches in developing APM solutions when deciding which parts of the code should or should not be monitored: the first and most widely-used approach consists in only monitoring what is stipulated, usually through configuration files. The second approach uses more sophisticated algorithms that dynamically instrument what is considered necessary based on heuristic properties.

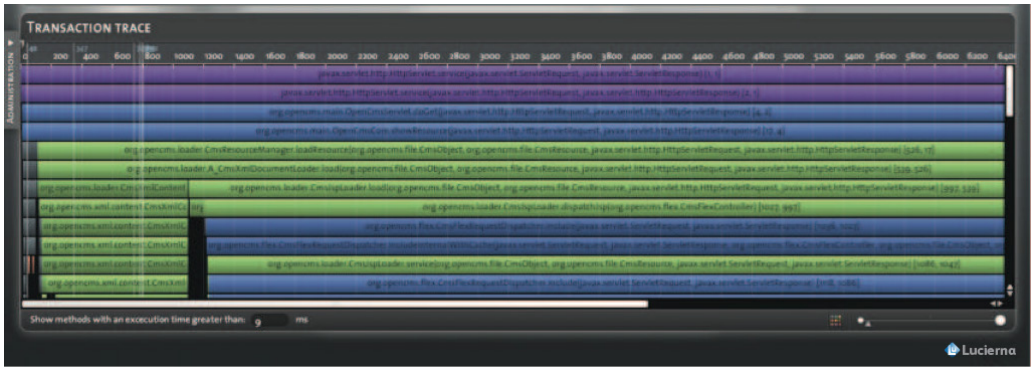
Thanks to the innovative technology that we have developed, lucierna is able to monitor all of the client's code with an overhead of less than 2%.

Features

100% of code instrumented — 100 % of the time at 2% overhead. The approach used by lucierna is to monitor 100% of user code and almost 100% of the rest of libraries and packets with no configuration. In doing this, lucierna can achieve the following:

1. Lower TOC and immediate ROI: as there is no need to include any configuration or indicate what to monitor and what not to monitor, lucierna may be installed very quickly and automatically tracks all changes occurring in your application.
2. Capacity to find problems that are impossible to diagnose with other solutions: by monitoring 100% of transactions, 100% of the time with the same visibility for 100% of the code.

We analysed the best solutions of our competitors in our laboratory* using OpenCMS as the application of reference. With the same overhead **lucierna monitors and reports 960 methods per transaction while the solutions of our competitors monitored 6** methods for the same transaction.



When your organization experiences problems, who would you want to have by your side?

*An internal study was carried out comparing lucierna with 5 of the most innovative and prestigious APM solutions on the market.

Smart Baseline

At lucierna we know that often, the problem does not reside in knowing where the defect is but in defining what defectuous means. Therefore, as well as the basic threshold mechanism for defining alarms, lucierna incorporates unique technology capable of learning how to deploy your application and its components, how your clients use it, the relationship between the two metrics in order to establish complex predictive models and issue warnings in the case where deviations occur.

For example, lucierna is able to warn you of a possible problem in the entrance Firewall today at 09:23 because, although today is a holiday and its load usually drops to 2 tps during the summer, today it is experiencing an entry rate of 1 tps.

Do you prefer to encounter problems or preempt them?

The Real Experience of Each User

The experience of each of your users is relevant, however there is only one way to determine it: by measuring it. Don't be deceived by far-fetched arguments – there is no other way. This is precisely why at lucierna we develop the technology that will enable you to measure the experience of your users – each and every one of them. It also collects additional information vital for diagnosing problems such as the network capacity of each client, geo-positioning, etc.

By automatically and transparently re-injecting small portions of JavaScript code, lucierna obtains the real experience of each of your users, measured where it really counts – in the client's system. In addition, lucierna also measures the user experience in mobile native applications (*Currently Android).

When your users complain would you prefer information measured as it exits your data center or information measured in your clients' systems?

Auto-discovery Topology

At lucierna we believe in simplicity and we know how important it is for your organization to be capable of isolating any problems that arise. Therefore, we have designed an animated topology providing a bird's eye view from where you can identify where the stalls occur as soon as they appear.

This topology automatically discovers and completely breaks away from the concept of vertical silos. Now you will have the information about your systems, not as a whole, but the individual parts that contribute to each of the applications and services.

How often have you experienced a scenario where your database as a whole had been functioning correctly yet was later found to be causing the malfunction of a particular service?

Cross Correlation

Applications are becoming more and more complicated with multiple layers. Portals that consume services exposed by integration buses which in turn act as proxies of other services exposed by other application servers are increasingly common.

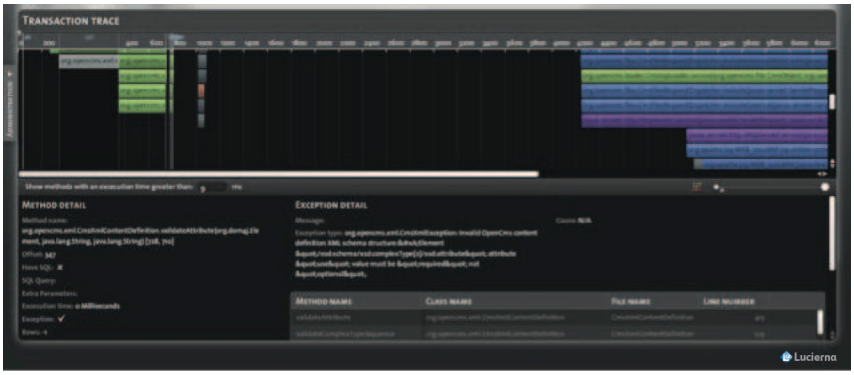
In this type of environment it is essential to track the transaction through each layer in order to identify bottle necks and points that need improving. Lucierna uses "transaction tagging" in order to assign synchronous and asynchronous protocols to this tracking, even between mixed Java and .NET platforms.

When analyzing complex problems and proposing improvements do you prefer the complete picture or isolated details?

100% of Exceptions

Other APM solutions only monitor certain types of exceptions or those that are thrown in very specific methods. Lucierna monitors all exceptions that pass through a method, including the message, the type of exception and their stack trace.

In a complex environment there are multiple stall points. Shouldn't they all be contemplated?



Advanced Backend Information

When other backends are accessed from an application server, as well as the access and process time, other relevant information may be retrieved, for example the parameters that are passed to the Prepared Statements, or the effective bandwidth for each backend call. At lucierna not only do we identify a large number of backends (SQL, NoSQL, LDAP, Documental DataBases, JMS, MQ, Mainframe, CICS, jboss remoting, EJBs, WS: Axis-1/2, JaxWS, JaxRPC, TibcoWS Stack) but we also provide the most relevant information, including the parameters that determine the queries to identify a backend and also extra parameters which are essential in order to gain a detailed description of how each of the backend elements behave.

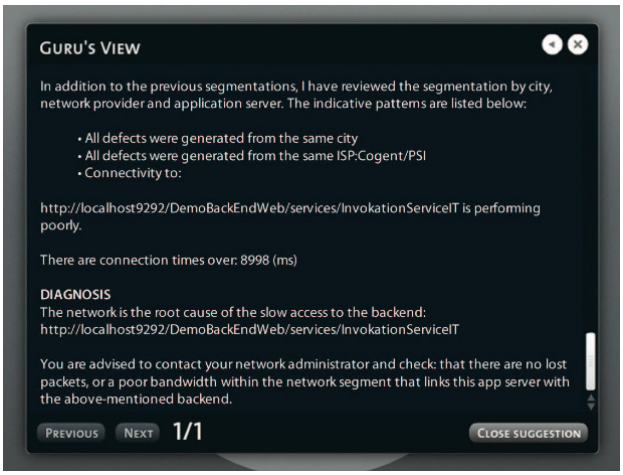
Have you ever complained to your DBA about a very slow query who responded by asking you about the parameters used in the query or by blaming the network?

Unassisted In-depth Diagnostics

At lucierna we call Unassisted In-depth Diagnostics the capacity to diagnose different kinds of problem using straightforward language and with no human intervention, including:

- Inefficient code
- Slow backend consultations
- Problems related to different versions
- Memory leaks
- Problems in the access network of the network that communicates with backends
- Problems related to specific ISPs
- Problems related only to specific cities or countries.

Based on a 360° view of the platform with thousands of metrics analysed per second, including bandwidths, specific JVM/CLR metrics and execution times of methods, and implementing state-of-the-art techniques in artificial intelligence and expert systems, the problems are diagnosed as clearly and simply as possible.

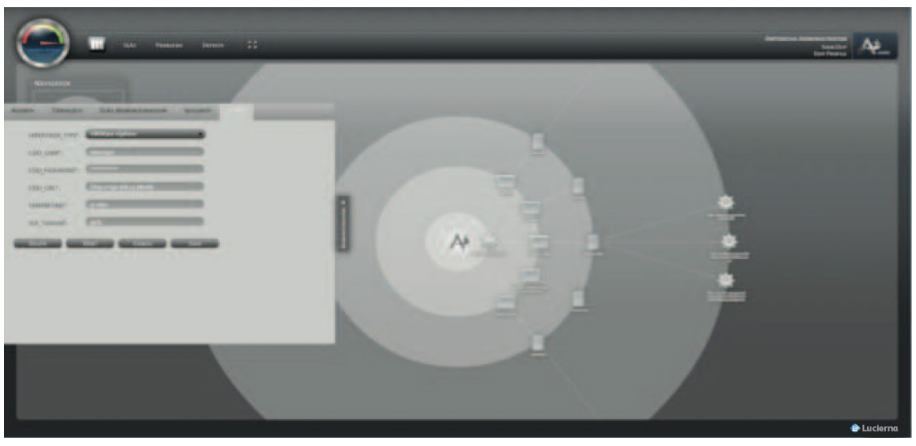


At 03:00 am, when everyone, except the support department is asleep, don't you think that a real expert should be on call?

Capacity on Demand

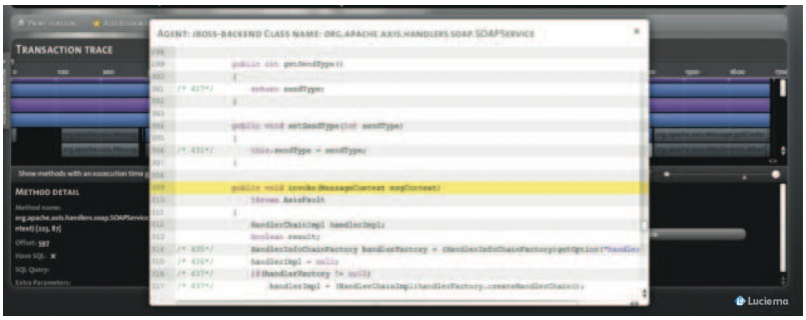
Using the service metrics and end user experience levels as an entry point, lucierna is the most reliable data source for making auto-scaling decisions. Its integration with suppliers of PaaS such as Cloudify and of IaaS such as StackOps enables it to easily create highly flexible and competitive environments, making maximum use of the available infrastructure while maintaining optimum user experience.

Why not pay for only what you really need?



Hot-Decompilation

During incident resolution processes it is sometimes necessary to access the source code in order to make a fast diagnosis. However, this is not always possible, either because the code belongs to a library that you did not develop or because the person conducting the analysis does not have access to it. Lucierna provides you with the code that was executed with your transaction because we know that it is fundamental for making fast diagnoses.



We call this functionality hot decompilation and it does what its name implies; it generates the bytecode of the executed method on demand, decompiles it, embellishes it and displays it on the screen. Therefore, when problems arise it can find them even in external libraries.

Why wait to be granted access to the code while you lose clients?

Integration Capacities

The principal asset that lucierna contributes to your company is not the tool in itself, but the information that it retrieves. This is why we make it accessible in the most appropriate way for each tool to ensure fast and smooth integration in any business environment.

Lucierna incorporates the following integration mechanisms:

1. **Two-way SNMP:** which enables integration with your fault and service management systems. Lucierna sends TRAPS to the fault managers and enables the metrics in each node of the topology to be consulted with GET.

   (Nagios, HP OpenView, IBM Tivoli, Infovista, etc.)

2. **Open databases:** that enables direct integration with your BI or reporting systems. Lucierna stores all the information in an open and documented model which can be connected with your favorite BI tool.

   (Pentaho, Spagho BI, Oracle BusinessObjects, microstrategy, jasperReports, CrystalReports, etc.)

3. **WS Facade:** which enables the automation and integration of any call that can be made from the user interface through WS.

Why make your operators or analysts learn how to use a new tool when they are comfortable using the one that they already have?

To sum up

Do you still **think** that there is a **better** APM solution?

Lucierna
by SMARTBEAR