INTRODUCTION: COMPOSITE APPLICATIONS ARE COMPLEX

CHALLENGES IN MANAGING COMPLEXITY

NORMAL VS. ABNORMAL

USING AUTOPILOT M6 TO MEASURE “NORMAL” VS. “ABNORMAL”

A SCENARIO – PAYMENT PROCESSING

VISIBILITY FOR THE BUSINESS

SUMMARY
Introduction: Composite Applications are Complex

Complexity is one word that describes today’s composite applications, and it’s growing. Here is the definition of complexity from Wikipedia: "A complex system is a system composed of interconnected parts that as a whole exhibit one or more properties (behavior among the possible properties) not obvious from the properties of the individual parts." The interesting part of this definition is "... parts that as a whole exhibit one or more properties (behavior among the possible properties) not obvious from the properties of the individual parts." One can conclude that even misbehavior in one or more components of a complex system may not necessarily compromise the system as a whole. The basic question arises: how does one manage the growing complexity of today’s composite applications that in turn depend on many other complex systems which are in themselves are prone to failure such as networks, servers, storage, security, firewalls, and even electricity? The other question is how do we know that our "complex system" is in fact misbehaving or acting outside of "normal" and what will the impact of this be to the business? These questions are much more difficult to answer since first, one needs to define what "normal" is for a given set of interconnected systems. Today’s management tools are pretty good at identifying faults such as server availability, network faults, resources utilization, etc. A typical data center receives thousands of alerts daily about all kinds of faults. Many of such alerts have nothing to do with system-wide outages, but are in fact part of the normal operation of complex systems.

Challenges in Managing Complexity:

Uncover problems before users are impacted...resolve them before the pain is felt...

Composite applications are prone to failure, healthy composite applications can tolerate and adapt to failures and still continue to function within acceptable limits. While fault management is a key part of managing complex IT systems and applications, a more effective approach is to understand "normal" vs. "abnormal" behavior. Faults have to be examined in the context of this behavior rather than each by itself. For example, a “server down” alert is clearly an indication of a failed system, but if the server is clustered and the failover succeeded without service interruption, the "server down" is part of normal operation of a complex system and should not raise a red flag – it is in fact normal. The same applies to the performance attributes of complex systems such as response time, volume and latency. Knowing normal ranges, deviations (dispersion) is key to understanding how complex systems behave. It is true that, normal does not always mean "acceptable." We may learn that the normal response time of an e-
commerce site is around 10 seconds, however; it’s not acceptable for end-users. A ten-second response time is too long. Here too, understanding normal needs to be examined in the context of "expectations."

The other key dimension of composite applications is transaction flow – the context and the flow of information from one part of the application to another. This is a circulatory system of a composite application and is one of most important sub-systems of any business service. Monitoring the flow of information, measuring “normal” vs. “abnormal” and comparing against business expectations is the most effective way to monitor composite applications. Understanding transaction flow and measuring normal vs. abnormal with an ability to compare against a set of expectations is key to managing complex systems and reducing the impact of failures and preventing the risk of cascading failures.

**Normal vs. Abnormal**

So how does one define normal for a given system? To understand normal behavior of a composite application we need to have:

- The inventory of all components, servers, applications and networks
- The interdependency of components and their topology
- The flow of information, meaning how components communicate and interact
- All relevant KPIs (key performance indicators) that describe the behavior of each component and the system as a whole (such as response time, latency, error rates, volume, transaction rate, etc)

Most advanced IT organizations already maintain the inventory, dependency maps of their server, networks and applications. The challenge is to understand how and what kind of information flows across different components, as well as the relevant indicators that describe the operational and performance health of each component and the system as a whole. Some of the common key performance indicators for a typical web based composite application such as a web portal are:

- Response time by server, by URL, by geography
- Latency by server, by URL, by location, by component
- Transaction/processing rate
Once we know all the components of a given composite application, its topology, transaction flow and the set of key performance indicators, we should be able to say something like “notify when one or more of our indicators within a given flow are “abnormal” i.e. “outside of normal.” While the English expression is rather simple, the practical implementation is tricky. To determine “abnormal,” one must find out what “normal” is.

Complex Event Processing (CEP) is a technology that can effectively weed out normal occurrences and zero in on only those that matter. What is CEP? Here is a definition from Wikipedia: “Complex Event Processing (CEP) consists in processing many events happening across all the layers of an organization, identifying the most meaningful events within the event cloud, analyzing their impact, and taking subsequent action in real time.” Combining CEP, transaction flow and analytics allows organizations to express multifaceted business conditions and apply them to a complex system, such as composite applications, business services or a business process. The value obtained from this includes correlation across multiple domains, problem prediction and prevention – resulting in greater visibility and better alignment between IT state and business impact.

**Using AutoPilot M6 to Measure “Normal” vs. “Abnormal”**

AutoPilot M6 is an application and transaction performance monitoring solution that combines Complex Event Processing with predictive analytics to distinguish normal vs. abnormal behavior and compare it against user expectations. AutoPilot M6 determines normal by the following:

1) Establishing a rolling based line of number of user-defined samples for a given set of KPIs

2) Computing statistical indicators that measure dispersion, momentum, rate of change as well as many other indicators

3) Comparing new samples to the established rolling based line to measure the % of normalcy – meaning how normal the sample is compared to the rolling baseline.
AutoPilot M6 combines several key technologies that are an absolute must for gauging “normal” vs. “abnormal behaviors”:

- **Auto discovery of applications, middleware and transactions** – the capability to automatically “find” and “catalog” all the applications of interest, the middleware such as messaging, buses and brokers that interconnect them and the transactions they invoke.

- **Event and Metric Streaming** – the ability to collect events and metrics KPIs from multiple sources in real time.

- **Transaction Analytics** – the ability to analyze transactions non-intrusively by observing interactions between various applications components and systems, across multiples tiers (including z/OS (mainframe)).

- **Complex Event Processing** – “is primarily an event processing concept that deals with the task of processing multiple events with the goal of identifying the meaningful events within the event cloud” – Wikipedia. Use CEP to identify patterns of activity that may be “normal” vs. “abnormal” for a given set of applications, systems, server or composite applications.

- **Statistical Analytics** – the ability to determine how each KPI behaves, including its rolling baseline, deviation, momentum, rate of change, advance and decline ratios, and many other factors.

- **State Modeling** – user defined state models that compare observed behavior with the desired outcome. State models are useful when detecting complex patterns and deviations from the user expected behavior.

- **Automated Actions** – the ability to trigger user defined actions, alerts, notifications, emails, scripts when user defined situations are detected. State models usually trigger actions in response to a pattern or activity.

- **Real Time Visibility** – real-time dashboard of applications, business services and transactions that compose a given business process.

- **Integration** – downstream and upstream integration with third party technologies, event correlation, problem management and other enterprise technologies.
Combining Transaction Discovery with Complex Event Processing and analytics can yield dramatic reduction in Mean Time to Problem Resolution (MTTR) and decreased Mean Time Between Failures (MTBF) by detecting dangerous patterns ahead of time and providing the necessary visibility to deal with current and potential threats that can degrade or disable a complex system, such as a billing or a payment processing application, trade settlement or trade clearance.

Some of the analytical instruments used by AutoPilot M6 to gauge trend and define “normal” vs. “abnormal” include the following:

- Moving and Exponential Moving Averages – used to measure long term trends
- Standard Deviation and Number of Deviation from the mean – measure dispersion
- Bollinger Bands (High/Low) – a way to gauge high and low for a specific metric or KPI
- Relative Strength Indicator – a measure of momentum for a specific KPI
- Rate of Change – measure of the % change in specific KPI relative to the base
- Average Gain and Loss – average gain and loss based for a specific interval
- Average Velocity – rate of gain or loss (usually measured in units/second)

AutoPilot M6 provides many other indicators that can help users understand the behavior of complex systems. All such indicators can be used in M6 CEP expressions to define what “normal” vs. “abnormal” as well as express the desired state for a given system, application or set of composite applications and KPIs.

**A Scenario – Payment Processing – Funds Transfer**

“Its 4 PM. Do you know where your payments are?” All organizations processing payments will most definitely answer yes. Payment processing applications, for the most part, have the necessary checks and balances to know what happened, was the payment issued, posted or cleared. However, most don’t really know where the “in-flight” payments are and if there is a problem during the payment processing. Most lack the visibility into the plumbing that actually transports, translates and orchestrates the payment processing. This layer collectively named “middleware,” which consists of the web, application, messaging and the database layers. The orchestration of the payment processing is accomplished in...
the black box beyond the reach of applications and BPM tools. Once the processing is handed off to the “middleware” which in most cases is comprised of many asynchronous processes that act on the individual payment or a fund transfer, any hick-up can potentially disrupt the processing of one or more payments. The results could be loss of revenue, currency exchange risk, penalties associated with missed SLAs, compliance risk. Tracking the lifecycle of an individual payment transaction as it makes its way through the “middleware” is an important part of managing risk associated with “inability” to process a single payment.

AutoPilot M6 allows users to take control of their most critical in-flight transactions by:
1) Discovering transactions and providing the necessary analytics around SLAs, response times, latency and failures and
2) Applying CEP and analytics to find patterns that can impact business services quality and avoid system wide outages.

AutoPilot M6 CEP engine can apply the following expression to determine whether payment processing time is exceeding normal behavior: “Monitor all payment transactions and for any Payment.Latency which exceeds Payment.Latency.BollingerBand-High than flag as a threat (Critical) and notify the appropriate personnel.” However, it does not take CEP expertise to take advantage of its benefits. M6 Policy Wizard is provided by AutoPilot with pull downs to make this process intuitive. Below is a screenshot that show how one can define such an expression without complex coding or scripting. The wizard simplifies what otherwise would have to be expressed in a SQL like EPL (Event Processing Language) (SELECT Latency FROM Payments WHERE Latency >= high (Latency)).
Figure 1: AutoPilot M6 Wizard to define CEP expressions, where %f0 refers to the set of all components of a composite application and Value and History-Band-High are derivative metrics for all such components calculated by the M6 CEP engine. This dialog box defines a situation that combines a complex expression with analytics as well as timing. It tests for: Payment transactions, Quantity > moving avg, Duration > moving avg and Time of day > 3 PM.

Pseudo code the above dialog box represents:

If any transaction_type=payment
   And transaction_type_qty > EMA (Exponentially Moving Avg)
   And average_duration > Bollinger_High_Mark (High Normal Range)
   And it is after 3 PM, then
Alert: Reconciliation in jeopardy AND Execute automated action to prevent impact.

Turning on the policy, which is a collection of CEP rules and conditions, allows users to track all payment transactions and automatically flag only those whose latency is approaching “abnormal” levels (see image below):
While the image above shows the real-time view of the payment service, AutoPilot M6 Transaction-Works also allows users to deep dive into specific payment transactions which are in flight, missed their SLAs, failed or completed. The diagram below depicts the topology of a payment transaction with the actual trace of the steps taken during its processing. In this example, a payment transaction originated in the application server (in this case, WebLogic), went through the messaging layer (WebSphere MQ), was processed by the WebSphere Broker ("dataflowengine") and a response was consumed by the same application server. The percentages are showing how much of the time was spent at each interaction. Based on this example, it is quite obvious that 90% of transaction processing was spent in the middleware layer. AutoPilot M6 uses a unique process called Transaction Stitching to relate synchronous and asynchronous interactions into a single transaction without disrupting user data.
Figure 3: Transaction Topology shows how an individual payment traverses the IT infrastructure. In this case it included WebSphere MQ messaging layer, WebSphere Message Broker, CICS as well as a J2EE WebSphere application server. The percentages indicate the % time spent in each layer, while arrows indicate direction of the flow. 90% of the processing time was spent in the middleware tier. The application highlighted is where the SLA was breached due to the message sitting in the queue too long.

AutoPilot M6 TransactionWorks also provides a broad view of how transactions flow through the environment and where missed SLAs or failures are clustered (see Figure 4). The example below shows a partial view of a composite payment application with SLA violation rate broken out by tier.
Figure 4: Transaction Activity by Tier shows the breakdown of transaction volume by application, server, resource and user as well at each point the % of transactions that either complied with or breached their SLA and the % of transactions that failed. Here we are showing a partial view illustrating SLA breakdown by resource type.
Figure 5: Transaction Performance by tier allocation used to analyze hot spots and plan for future capacity. Pie chunks indicate % of the time spent in a specific tier.

Below is a portion of the trace of all the steps taken within the middleware layer. In this case, it includes Application Servers, messaging, as well as the databases. The trace indicates all of the smaller steps that were executed as part of this payment transaction including: requests, method calls, SQL queries, JMS calls, WebSphere MQ interactions, message exchanges and message payload. Storing message payload provides the necessary context for problem determination since often times it is the data that causes problems in the application business logic. AutoPilot M6 also allows users to search for transactions based on the tags extracted from the message payload. This facility becomes indispensable when answering “Where is my transaction, message or payment?” Application Performance Management is not just about performance, but also about capturing business context in order to reduce the complexity of application support.

So what are some of the benefits of such a solution for your payment service?

- Find and fix problems before they impact your business (increase MTBF)
Isolate problems and performance degradation quicker (~95% reduction in MTTR, compared to manual efforts)

- Complete audit trail of your transactions for compliance
  - Search for a specific payment and produce audit trail
- Find root-cause of the problem, resolving the following questions:
  - What is causing the problem?
  - Where is the problem?
  - Why is the problem happening?
- Avoid the fallout associated with missed, failed or poorly executed transactions:
Loss of revenue, customer attrition

- Potential penalties associated with missed payments, transfer
- In case of payments – risk associated with currency fluctuations

Visibility for the Business

While AutoPilot M6 delivers actionable information for IT to proactively deal with risks associated with disruption in transaction flow, businesses must have a view into the overall quality of the business services being delivered by the organization. AutoPilot M6 delivers a Business Activity Dashboard that provides meaningful information to the business as well as actionable information for IT.

Figure 7: Business Activity Dashboard designed for Line of Business Managers to have a complete view of mission critical business and IT services.
Business Activity Dashboard presents a high level view of all critical business services, their health, as well as all key IT services that power business-critical activities. Business Activity Dashboard is a web portal and can consume information from other sources as well.

**Summary**

Managing growing complexity while improving service quality and reducing cost is one of the key challenges of today’s IT organizations. AutoPilot M6 features CEP and an analytics engine specifically designed for monitoring application and transaction performance. This key technology enabler lets organizations proactively monitor applications and transactions in the context of the business requirements. Nastel’s AutoPilot M6 solution allows organizations to tackle application complexity by providing complete transparency into the lifecycle of an organization’s most critical business transactions 100% of the time, 24x7, across all tiers. *M6 CEP (Complex Event Processing) capability further enhances business agility by proactively assessing behavior of composite applications against business requirements and objectives and enables organizations prevent problems before they impact the business.*

Nastel AutoPilot delivers meaningful information to the business while at the same time providing actionable information to IT.