White Paper

metagroup.com • 800-945-META [6382]

June 2005 Will Cappelli

Composite Applications and the Application Management Life Cycle

A META Group White Paper

METAGROUP

Contents

Introduction 2
The Varieties of Composite Applications 3
The Management Challenge of Composite Applications 4
Three-Dimensional Application Management 6
The Application Development Challenge of Composite Applications 7
Composite Applications as a Catalyst for App Dev/Ops Synthesis 8
From Technology Solution to Cultural Transformation 9

Executive Overview

Business processes are becoming increasingly dependent on composite applications, and it is essential for on demand businesses to maintain their availability and performance. At the same time IT organizations are finding that traditional systems management tools are unable to identify or diagnose the new types of performance problems that composite applications introduce to the environment. A new organizational challenge is also highlighted, which is the need for tight integration between IT operations and development teams in order to quickly resolve composite application problems.

As organizations evaluate composite application management solutions, there are three questions that should be asked. These questions will help organizations evaluate the capabilities around transaction tracking, deep-dive diagnostics and resource monitoring. All three capabilities are needed to be able to effectively manage composite applications.

Organizations that are embracing composite applications can realize substantial value in implementing an application management solution that is appropriate for their environment. These solutions can proactively recognize and prevent performance problems at the end user while improving IT staff productivity through simplified transaction management, deep-dive diagnostics and correlation.

Introduction

The structure and life cycle of applications are evolving and this evolution is requiring a fundamental change in the way in which applications need to be managed. In the past, the situation was relatively simple. On the one hand, applications were constituted by integrated and discreet blocks of code clearly segregated from the infrastructures on top of which they ran or leveraged underlying infrastructures. On the other hand, the collection of functionality the business user identified as an application serving his or her business needs more or less well corresponded pretty closely to the underlying entity developed by application developers and residing and running within the IT operational environment.

Now, however, applications are increasingly composed of multiple components. The boundaries between one application and another are frequently hard to discern and, even more importantly, many of the components, in fact, provide services that are ‘infrastructural’ in nature (e.g., load balancing, path optimization.) Furthermore, applications have become a lot more volatile in their constitution. Components can change without the overall functionality of the application in question being noticeably affected, even though over a long enough period of

time, the application can evolve to serve a completely different set of business purposes. Finally, a major disconnect has emerged between what a business user perceives as an integral application and the underlying reality of the IT event stream. The business user may see as one element what is, in fact, a shifting population of underlying components, superficially held together by a shallow portal construct. Alternatively, what appears to the user as a completely cordoned off array of functionalities available to him or her alone (e.g. via a portal or executive dashboard) is, in fact, part of a tightly woven function point fabric being simultaneously shared by users around the globe.

Traditional application management has been a discipline or set of processes associated with the maintenance of application health, availability, and performance in a run-time environment. How is that discipline transforming, in light of the change in the nature of applications?

We will call such volatile, possibly ephemeral, user perception driven collections of functional components, 'Composite Applications' and argue that successful Composite Application Management requires:

- First, a deep interweaving of three traditionally distinct ways of doing Application Management and,

- Second, an integration between processes usually associated with operationally oriented application management and those associated with application development post-deployment.

While the technical challenges associated with the former are greater than those associated with the latter, effecting the latter integration will undoubtedly prove to be a far more difficult task for technology provider and technology buyer alike. Application development and operational application management respectively reside in the midst of the two very different, and often antithetical IT cultures. Nonetheless, the cooperation between those two cultures is fundamental to ensuring IT/business alignment in an era of rapid business change.

The Varieties of Composite Applications

Composite applications are in essence combinations of more fundamental components of functionality. It is, however, important to keep in mind that such combination can occur in a number of dimensions. First, the combination can be horizontal. Multiple components, all of which interact directly with a user may be linked together in such a way that the user moves seamlessly from one component to the next. This is the kind of combination that one finds when an

organization integrates CRM and ERP functionality. Alternatively, the combination can be stacked vertically. Along this dimension, a single component may interface directly with the user but it stands on top of and transparently accesses the functionality of many underlying components that in turn rely on other components in a more or less hierarchical manner. This is the kind of combination that one typically finds in a Web-centric N-tier application e-commerce architecture.

In addition to this distinction between vertical and horizontal modes of combination, there is also a distinction with regard to whether the various components of a composite application are linked together as a matter of developer's design (whether 'binding' is early or late) or response to environmental necessities or whether their linkage is primarily a matter of user perception. It is important to note that any given composite application can be combined in both a vertical and horizontal manner. Also, certain components in any given composite application may be linked in virtue of developer's design while others are linked via user's perception. Nonetheless, as we shall see, different modes of combination require different types of management discipline and, hence, the best way of dealing with the entire composite management phenomenon is through a multi-faceted but necessarily deeply integrated approach to application management and application development.

The Management Challenge of Composite Applications

While there are many good reasons for the emergence of composite applications having to do with business requirements for agility on the one hand and reduced development costs on the other, there is a sense in which they are highly problematic constructs. Increasingly, business units are requiring that internal IS organizations and external service providers alike package the functionality which the business units consume as services, i.e., business intelligible flows of capability or transactions, delivered according to strict service level agreements which are defined by metrics that themselves make immediate sense to the business user. Supporting a catalogue of such services requires the IS organizations to not only treat the applications that support them as integral wholes but also to define their boundaries in a way that is congruent with the user's perception. The fact that many of the applications will be composite in one or more of the senses defined above makes the management task very difficult indeed.

In the past, application management was a comparatively simple task. In the course of processing, the code would generate numerous statistics which, if captured and analyzed, could signal an application's health or lack thereof in the run-time environment. Since applications were largely integral blocks of code,

statistics gathered at a relatively small number of places in the flow of logic could reveal quite a few symptoms affecting availability, performance, and resource consumption with reasonable probability. Furthermore, since there was a close correlation between the user's perception of the application's boundaries and the underlying business logic, these relatively sparse statistics were reasonable indicators of the user's 'end-to-end' experience.

Composite applications, however, change the rules of the game.

- First, since the application is now broken down into a possibly large number of components, the health of one component will, in general, tell the system administrator little about the health of other components. Hence, a larger number of statistics will have to be gathered all across the logic flow.

- Second, in many cases, the application will exist as an integrated entity only at the point where the user interacts with it. Statistics, even a rather comprehensive collection of statistics, will not reveal what, in fact, the user is experiencing. Hence, it will be critical to monitor user perception independently of the component statistic gathering process, for example by evaluating the response time of synthetically (robotically) generated transactions.

- Third, the transactions that flow through a composite application become rather abstract entities (especially in an SOA context.) In a more traditional, monolithic application, transaction flow was relatively easy to trace and integrity checks like two phase commit could be synchronized in a straight-forward manner with the processing sequence of the application's logic. In a composite application, however, a transaction needs to be monitored independently of the processing sequences taking place in the various components. In fact, there will frequently be points in the overall computation where the transaction 'dematerializes' upon exit from one component and 'rematerializes' upon entrance to another. Hence, transaction tracing emerges as an independent, and fundamental, application management process in a world of composite applications.

It is important to note that each of the three application management processes referred to above requires the other two in order to give the systems administrator a comprehensive view of the health of a composite application.

There is a temptation, unfortunately exploited by some technology providers, to glide from the requirement for end-to-end management and the recognition that, for many composite applications, an end-to-end picture is only available at the user interface, to the conclusion that end user experience monitoring is the only

functionality that really matters. While the temptation is understandable, particularly in an era of restricted IT budgets, it needs to be categorically stated that, while extremely valuable, end user monitoring only gives the systems administrator a superficial understanding of what is transpiring in the IT event stream. In the end, all one is getting to see are the symptoms of any underlying problem and not the causes of those symptoms. Furthermore, those symptoms will only show themselves long after the pathology has set into the underlying system. Since the maintenance of service levels is as much, if not more, about the proactive prevention of problems as it is about the rapid resolution of problems once they appear, composite application management requires all three processes to be running simultaneously.

So while monitoring the user experience is critical in terms of ensuring that applications adhere to service level objectives, deeper insight into the event stream becomes necessary to deconstruct these events and recognize patterns associated with performance degradation that directly impacts end users. Fortunately, tools are available today from vendors like IBM and Microsoft that look deep and wide over the various dimensions of application management to help detect and quickly triangulate failure modes that could be localized to one or more components. Point failures may also have cascading downstream effects, such that determining root cause becomes more arduous because multiple alarms or symptoms may occur simultaneously – the pressing question is where did the problem originate from?

In order to establish root cause, a filter must be applied to the alarm noise to quickly rule out all of the components that are not at issue, and narrow down triage paths to the one or two prime suspects, for example a flaw in the business logic such as a looping condition, or a resource constraint such as thread pool starvation. The key to success here is to leverage composite management tools to provide both the necessary monitoring points across the infrastructure in order to triangulate with precision, and then add automated correlation of events to guide the operator or subject matter expert towards the problem's origin in order to fix it or implement a workaround. A helpful way to visualize this process is to think in terms of three dimensions that define the space within which application management processes unfold.

Three-Dimensional Application Management

When evaluating a composite application management solution, organizations should look for the following three capabilities. All three are needed for effective application management. An IS organization needs to ask itself three questions.

- First, is the solution able to gather, correlate and analyze a large number of statistics across a large number of multiple application components? This includes the ability to perform deep-dive diagnostics at the application and middleware level. Drill-down capability and correlation should extend to legacy application subsystems.

- Second, is the solution able to monitor the user's end-to-end experience of the composite application's health? This includes being able to follow the transaction flows and isolate problems by component. The solution should be able to proactively identify performance problems at the end user and be able to prove the transaction service level that is being delivered at the end user.

- Third, is the solution able to monitor the health of resources? This would include application resource consumption analysis by application and visualizing workload trends to enable tuning the environment. Organizations can proactively prevent problems by accurately forecasting the resource requirements for new (or upgraded) applications.

A further line of questioning would ascertain how the solution handles vertically and horizontally composite applications on the one hand and the degree to which the solution assumes that an application's composition has been planned at the design stage, on the other.

The Application Development Challenge of Composite Applications

By their very nature, composite applications tend to be more volatile than traditional, monolithic applications. Elements are assembled from re-usable components that can be dynamically upgraded, added, or subtracted without requiring more than an incremental change in the user's experience. Put another way, composite applications can evolve gracefully over time. This fact has deep implications for the relationship between the application development community and the operational management community within an IS organization. A composite application is, in some sense, always a work in progress and hence the operational managers charged with ensuring the health of the application at run time will be interacting with application developers on a regular basis as the latter add, subtract, and improve components.

This continuous interaction is very different from what IS organizations have been used to in the past. As long as applications were monolithic blocks of code, it was easy, if not optimal, for developers to code in isolation from the operational management team and interact with them only at the point when a new (version of an) application was introduced to the operational environment. The net effect of such infrequent interaction was the fostering of two distinct IT subcultures, each with their own vocabularies, processes, and vendor communities.

Overcoming the barriers that exist between application development and operational management has proven difficult despite the fact that global 2000 CIOs typically concur that this separation is one of the major reasons why the IS organization as a whole is unable to deliver high service level functionality to its business customers. Nonetheless, the emergence of composite applications is, indeed, beginning to accomplish that heretofore elusive goal.

Interestingly, the vendor community has, in many ways, been more responsive to the need to graft application management and application development together than it has been to the requirement for the three dimensional approach to application management discussed above. For example, Mercury, Compuware, IBM, Microsoft, and Computer Associates have, although in very different ways, endorsed the concept of an Integrated Application Life Cycle Management process which contains elements of both application development and operational application management. By contrast, only IBM and HP have market offerings targeted at all three of the required application management disciplines. Furthermore, even among those vendors that have all three capabilities, it is only recently that an explicit understanding of their necessary interplay has received an explicit positioning.

Composite Applications as a Catalyst for App Dev/Ops Synthesis

The integration of application development and operational application management must, to be ultimately successful, take place along, in this case, four dimensions.

- First, application development processes and tools must explicitly support manageability as a design goal. This involves the ability to clearly interlace application logic with points of management oriented control along with guidelines as to where such points of control can be optimally inserted.

- Second, application development and operational management processes must use a common information model to describe the objects with which they are dealing. It should be noted that the near universal acceptance of UML as a basic metalanguage and XML as a means of communicating concepts takes the industry many steps down this path. Nonetheless, much hard work remains to be done to develop the actual UML object descriptions that will be acceptable on both sides of the divide.

- Third, testing environments must be developed that easily and accurately reflect the run-time environment.

- Fourth, and what is probably the most important, process description standards like ITIL must be extended to explicitly portray the tight choreography required between application development and operational management.

These factors become critical for service oriented architectures to become truly viable. The SOA approach explicitly combines the development of a functional interface for an application component with the delivery of a service level definition. While some vendors (e.g., Microsoft and IBM) are in a strong position to lead this convergence, the scope of the problem is so great, we expect vendor driven leadership will have to be supplemented in concert with standards bodies, consortia, and strategic partnerships in order to deliver robust solutions that gain industry-wide support.

From Technology Solution to Cultural Transformation

Even more fundamental, however, is the willingness of IT organizations to accept the culture change that the integration of application development and operational management will require. Processes that intimately link the activities of the two communities are a good start but even they will not be sufficient unless individuals in both communities fundamentally change their understanding about the nature of the work that they do.

The old model that accepted a hard and fast distinction between building, on the one hand, and running, on the other, that maintained a sharp distinction between infrastructure and application portfolio must be replaced by one that sees the entire IT asset base as a single ever evolving system, regularly transforming the functionality it delivers to the business in response to new market conditions.

Summary

With the advent of composite applications, organizations must take a new look at the tools that are being used by their IT operations and development teams. Their traditional systems management tools are not suited to ensure the availability or performance of composite applications, which are becoming increasingly important to the business.

When evaluating application management solutions, three key capabilities must be provided for effective management:

1. Gathering, correlating and analyzing a large amount of performance information across a heterogeneous environment, including legacy subsystems.
2. Tracking the user's end-to-end experience and quickly isolating performance bottlenecks.
3. Monitoring the health of resources to proactively prevent problems.

Organizations are realizing business value from utilizing composite applications, and implementing an application management solution can provide IT with the management tools needed to address the new challenges that are created by these applications. It also provides a unique opportunity to integrate the IT operations and development teams, which can result in increased IT staff productivity, along with faster problem resolution.