UK Retail Company Gains the Visibility to go Beyond Mere MQ Monitoring

THE PROBLEM

It's likely you've been working with WebSphere MQ (WMQ) for years developing, deploying, monitoring or all of the above. It's also likely that by now you have assembled a tool bag of items to support your implementation and ongoing operations. And you have no doubt become accustomed with dealing with problems within the MQ domain. But what if you could see more of the transactional journey on either side of MQ? Organizations doing this are finding **new ways to improve levels of service and avoid costly mistakes and upgrades.**

Initially just seeing the touch points, you know — the "puts" and the "gets" — along with MQ administration probably seemed entirely sufficient in order to manage WMQ. A common assumption was that as long as we saw messages coming and going, things were okay. Analogous to Archimedes' principle of water displacement, the health of MQ or any other middleware system could be assumed as long as both sides remained within a reasonable state of balance.

OVERVIEW

A Process to Solve Complex Issues

It didn't take very long at all to see that more was needed. The difficulty was that by the time **an imbalance was noticed**, too many problems had already been caused; it was obvious that an earlier warning was needed to avert problems or at least correct them sooner. The need to watch things like queue depth, channel status, or any of the other 40 standard events that MQ raises become clear and pretty much commonplace.

If you are still requiring more precise information to detect problems earlier and to see their impact on the applications and business processes, some of you may have had the experience of configuring your own customer events based on conditions with your unique MQ implementation. If you're really on top of the MQ management game, you've implemented automatic corrective actions, which launch the moment warning signs appear — preventing problems before they occur.

RESULTS

Despite your success implementing and managing WMQ, and at the risk of stating the obvious, your needs continue to chance due to some constant forces:

- **Increasing EAI complexity.** Even though most have implemented WMQ for a specific single project, the need to integrate it with other applications and systems continues to grow.
- **The need to do more with less.** Due to decreasing resource availability to manage the complexity with smarter tools and processes.
- **The demand for higher levels of service.** Not just to track and report on the level of service delivered, but also to resolve problems faster — in real-time before they impact the business.
- **The requirement to align IT services with business objectives.**

Delivering and supporting the service specifically to the needs of the business process. Amid a push to run faster and jump higher while carrying heavier loads, we commonly find this mistake: looking over the most practical solution. What is the most practical solution to these challenges? We've seen that one of the most straightforward and effective things you can do is to **expand your monitoring of middleware** to include more of the entire application infrastructure.

THE PROBLEM

If WMQ is the only middleware technology you have, you are unique. If you have any enterprise applications (commercial or homegrown) that do not also talk to Oracle, DB2, or SQL Server, then let's just say you are in a class of your own! Unless we've just described your environment, you also have a wide collection of management and monitoring tools for all these various platforms and systems – each of which lacks the ability to see or do much of anything beyond its own domain. The solution is to monitor and manage more of the overall application infrastructure. It may sound like the holy grail to track communications across all of your applications and through all of your middleware systems as a contiguous whole, while at the same time correlating the events generated by each platform. However, the fact is that it is being done – and it's much simpler to do than most expect.

We found this to be the case of this company, one of the United Kingdom's preminent retailers, and saw an opportunity to improve our performance.

Diagnosis of the problem wasn't the real issue – all the information we needed was there. But it was just taking too long to put it all together. We had been doing a good job of sorting out problems within the MQ space for some time. Our problem was that most of our time was spend traipsing through the API layers of our applications that were integrated into MQ in order to find problems. The difficulty was to see the whole picture. Think of it as pushing sausage meat along a sausage casing without any knots in it. We never knew quite where we were.

This is not an isolated problem.

Often "blind spots" in the round trip of messages make it difficult to see just where the hang-ups are. Even more troublesome is the cover that these blind-spots usually provide for the vendors involved to play the finger-pointing game while your valuable time is being wasted.