AVADA SOFTWARE
Infrared 360

Contain Costs and Mitigate Risks of Transactions on the Middleware Superhighway

Manage and Monitor the Middleware Superhighway While Containing Costs and Mitigating Risk

This article discusses management and monitoring of transactional environments by using a simple representation of the problem; a metaphor of highways as the network, traffic as the data, on ramps and off ramps as the intersection where the data meets the network.

I was initially attracted and volunteered to work on this strange technology called MQ back in 1996. As a person who was working on converting 3270 screen data into HTML via TN3270 sessions, I thought this was a novel way to get data from here to there, forget the screens, stick to just the data, and use any programming language on any platform I chose. Programmatically and architecturally, it made sense. At least more so than mapping 3270 characters to HTML!

As MQ matured, many companies were using it to assure transactions would get from point A to point B and, if they were lost or misplaced, could at least be located—most often in a default dead-end location. Fast forward to the current day, and transactions carry data across web services, rest, EDI, and enterprise message backbones. From afar, the transactions can be thought of as traffic going from point A to point B. Think of looking down from far above a set of highways; cars and trucks and buses travelling along, entering and leaving this data highway. There are many features besides the actual highway to think about: lanes, intersections, tolls/gateways, merging lanes and also different languages for signage.

In real world road travel, control devices are good enough to direct the traffic, but don't do much for configuring changes to the patterns or alerting problems either before they occur, or when a certain behavior or threshold is exceeded. Most of that is dependent upon the manual intervention of traffic, police or construction crew. Come to think of it, that reminds me of many IT organizations!

In business, 'traffic' usually contains important content to run the business—whether financial data, supply chain data, personal information, travel logistics, etc. If these transactions are lost or misplaced, it generally causes grief for the corporation responsible for the transaction, not to mention their customers; whether it is a monetary loss (trades, bank transactions), or information loss that prevents business to move forward, (such as airlines, hotels), supply chain orders whether retail, wholesale, B2B, parts, inventory, etc.

In order to prevent such scenarios, companies spend considerate amounts of money on staff to make sure they can manage the entire transactional environment beyond the already significant sums spent for logistics of hardware and software. This is done in order to be able to make appropriate configuration changes to prepare or react to that environment. Normally, this is accomplished via a buildup of scripts and process libraries in order to keep watch on these transactions and the environment in which they flow. But many companies realize that a specialized software product focused on these tasks is needed for this purpose. In almost every case, the operating cost of a commercial product is far less than the budget necessary to build, enhance, maintain and support all of these responsibilities in-house.

Advantages of Building ☑

- Complete control
- Tailored to unique business needs
- Ownership of the software code

Drawbacks to Building ☒

- Development Time
- Training and Support
- Staying Current
- Integration with Other Applications
- Competitive Functionality
- Validation for Regulated Organizations
- Reporting tools?
- Employee Turnover
- Back Door Access
- Deploy now vs. when?
- Total Cost of Ownership
- Budgetary flexibility
- Employee resources available
- Opportunity Cost (time to market)
- All platform development
- Ability to execute on all phases
- Standards based

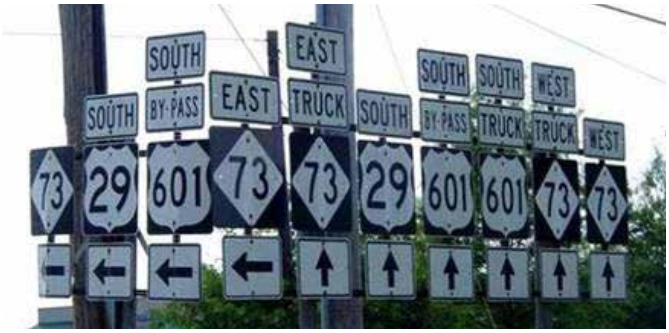
> *Unless you are operating a software company, software should not be central to the way you view your business. It's just a means to an end. And to be classed as truly successful, the means should be quietly efficient and as close to invisible as you can get.*
>
> —Robert X. Cringely, Inc Magazine
>
> Sources :The Standish Group, Robert X Cringely, 3C Software - To Build or To Buy?, Cincom - To Build or Buy?

Even in the case of specialized software products, some part of the IT staff is generally dedicated to managing those as well. This can range from entire departments for some of the heavier and surprisingly expensive solutions that require lots of scripting or customization, to just one or two people for more simple and intuitive solutions—even within large organizations. As a consumer of these products I always found some vendors a bit haughty in that they'd charge expensive fees to license products that in turn made *me* do tons of scripting and deployment. My philosophy as a consumer was that if I'm going to pay for it, I shouldn't be doing *all* the work.

Could you imagine hiring a contractor to build front porch steps onto your home and he says, "Ok, start measuring and figure out how many bricks per step and count them out for me?"

So let's go back to the traffic scenario. The advent of multi-platform then multi-delivery environments like web services, rest, and EDI in addition to messaging technologies has made the chore of managing those environments problematic because the work needs to be done on more than just the enterprise messaging backbone. This includes administration, configuration, and event monitoring. Typically when a variety of delivery systems and associated tools on each are utilized within the transactional environment, it makes correlation of problem events very difficult, isolating and identifying problems slow, and necessary problem resolution is often delayed.

Let us consider the enterprise message backbone as a main highway. The entry points of data can come from many interfaces to that highway: web services of many types, database queries, EDI interfaces, programs running in local or application server containers, transformation engines like Message Broker and IBM WebShere DataPower, etc.



These entry points are the on-ramps to the e-message highway. Data jumps onto the e-message highway to be delivered elsewhere via an exit point or off-ramp. Sometimes the data exits the highway, goes to a rest stop, and gets transformed. Sometimes the data is merged with other data onto the e-message highway. When this occurs it may need to be sequenced in order to find out which merged data belongs with which. This is similar to friends traveling in different cars trying to follow each other to a location, but other cars are interspersed between them. You'll notice they'll all get off on the same exit even if not directly in line with each other. As with any complex traffic, making it all flow is a multifaceted effort of preparation and adjustments.

In order to make sure this important data is not lost in this more complex environment, more than just the E-message highway needs to be managed, administered, and monitored. ALL of the on and off ramps, toll stations, rest stops, and weigh stations do as well, and in some cases, even the actual license plate number needs to be identified! Think about this simple analogy. If an exit (off-ramp) is closed, there'll likely be a backup on the highway approaching it. Conversely your highway may be traffic free. Is that good or bad? Well it depends upon the time of day and the normal pattern of activity on that highway. If it's rush hour and a stretch of road is empty it may be because a main feeder (on-ramp) is down. Think about tangent devices to middleware such as IBM WebSphere DataPower. If it's not feeding the transactions through to your middleware, then even a flawless middleware system won't help the transaction.

While the domain of transactions was originally with programmers, the fate of the programs controlling the actual handshake between systems has moved to platform software, originally via APPC, then EDI, then enterprise messaging, then web services—not solely for security but for the actual data exchange. Each performs their task admirably but differently.

Since the programmers themselves are forced to use quicker methods to keep up with the projects and timelines they are responsible for, it is not unusual that shortcuts are taken. These shortcuts can cause many issues. While developer talent is there, it is sometimes transient, usually understaffed and under-estimated for time requirements. Because of this, there is a higher risk to maintain transactional integrity. The characteristics that identify an MQ transaction are not the same characteristics that identify an application server transaction, and not the same as for a database transaction, and so on.

If the developers of these transactional systems don't plan for this cross-identification, then when an error occurs, there can be correlation issues when determining where the transaction went wrong. The effort involved in the synchronization of the characteristics that identify the error, capture and write it to a data store, and correlate the information about the error, can significantly decrease performance or require significant storage requirements in large volume transaction sites.

Given the effort and costs involved (time, storage, performance) most transactional environments do not manage or monitor their transactions using this method. Therefore it is imperative to become more proactive and isolate the points of failure in advance. If the points of failure can become known and corrected before a major transactional error occurs, then the above costs are mitigated.

In order to do so, some time is needed up-front to identify the on-ramps, off-ramps, and to identify the behavior on each of these that would suggest there may be a problem creeping up. After-the-fact problem-solving can be time consuming, resource consuming, frustrating, and fruitless. In this scenario, what is usually considered the most cost-effective solution is log analysis. Given different logs for different platforms are in different locations and formats, this is a problematic way to solve an issue as well. While there are some good software solutions for this type of forensic log analysis, sometimes this method doesn't allow for quick isolation, investigation, and action.

So why do companies spend so much on this infrastructure management? It's because the e-commerce infrastructure provides convenience and therefore customer satisfaction, and that is how you gain and retain customers. It is a quick way of gathering and disseminating business data, gaining faster time to market for new products and services, streamlining application business processes, and reducing operating costs. How much? When I worked for a typical large NYC bank in the 1980s, a paper bank transaction cost about $1.10. Voice response technology brought it down to 50 cents. Home banking software to 25 cents. Today, internet banking brings it down to just 13 cents. The positive, operating-cost impact is eye-opening when you consider the national and global reach of banks and the growing volume of daily transactions.

However, the reliance on e-commerce has a flip side. In today's IT world, the applications enabling the business processes are more distributed, more complex and more prone to transaction slowdowns or outright failure. Reliance on forensic problem analysis can be time consuming. This is not acceptable for any business process. The sheer volume and importance of these transactions make it essential to proactively manage and monitor that infrastructure in order to keep it continuously running.

The bottom line is that transactional systems and their associated infrastructure are essential for corporations to do business in today's world. How do you achieve this in a cost-effective manner and still provide agile, flexible, convenient services to customers or B2B partners?

The answer is clear: "Be proactive!"

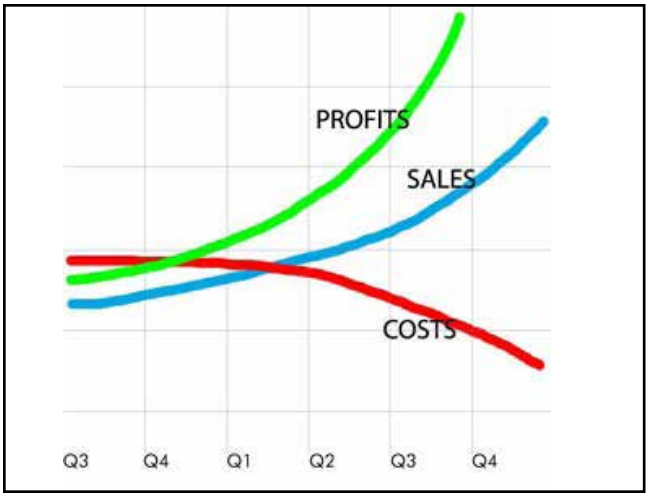
The following list contains rules of thumb to enable proactivity.

Use products or solutions that:

- Run on standards-based platforms and support standard software interfaces so that you do not paint yourself into a corner with proprietary systems that make change difficult and costly.
- Allow you to automate management procedures, to significantly increase your efficiency, versus having limited internal staff do everything in a reactive mode.
- Allow you to manage events at ALL the locations of the transactional middleware infrastructure, on the main highway, and the on-ramps and off-ramps.
- Provide an easy, intuitive interface, limit deployment time and reduce maintenance effort.

Operating **in**efficiency means wasted (cost) dollars, hurting the bottom line. Loss of productivity means even more wasted (revenue) dollars, hurting the top line. Identifying and deploying the most efficient and operationally cost-effective monitoring and management solution has been proven to increase business process profitability—a core goal of every organization.

White paper: "Managing & Monitoring Transactions on the Middleware Superhighway"
Author: Peter D'Agosta, Product Manager, Avada Software



Lessons learned from an IT Veteran

Perhaps it's due to lessons learned over 30+ years in IT development, support, administration, architecture, planning, and product management, or maybe it's because I gravitate toward new possibilities, but I have developed a core belief that simplifying IT is the best approach to get things accomplished. In a discipline notorious for making the complicated even more complicated, my goal is simply to remove the unnecessary complications from otherwise efficient processes.

While the acceptance of using open source has shifted methods and techniques quite a bit in recent years, most IT people, especially those who have been in the field for 20+ years, have a similar experience early in their career: While in the midst of either operationally or programmatically trying to solve an urgent problem, many IT brethren would rather watch you squirm than give you a simple syntax or a reference to suitable material that would lead to quicker resolution. I understand the 'teach them to fish' philosophy, but when Mrs. O'Leary's barn is burning you need one direct answer to extinguish the fire, not four to five cascading questions and a treasure map to get there.

Before I knew Unix, getting a Unix admin to give you the syntax of some arcane command (grep I ps –ef...) was like pulling teeth from an elephant. When I first learned zOS I was given a CMD line interface only to discover there was TSO (F-keys, Menus, short cuts). My first impression of IT people was similar to that of a fraternity; you had to do some crazy stuff and show you were worthy before they helped you out. After scrounging for vendor docs and creating lots of 'cheat sheets' to put all the commands at my cut/paste fingertips, I could finally concentrate on the problem at hand and not the syntax. Of course using a GUI would have been out of the question because it was for 'end users'!

Why am I bringing this up? Because it brings awareness to the fact that significantly improving productivity is more important than being a member of the IT "know-it-all" fraternity. At one time while I was responsible for instructional training, I realized the ability to "keep it simple" was crucial to the position. Another enlightening moment came when I was taking graduate courses and someone had asked me if I was a computer science major or an IT major. I didn't really know the distinction and like all creatures of both pursuits I sought out books and manuals, and periodicals. Sorry Yahoo and Google users, but we had to do it the old fashion way back then. Plus, I needed to do something in between submitting my 'batch job' to be processed!

I realized that I was not trying to solve equations for orbit trajectories, or figure out where the Fibonacci sequence hit seven digits. What I was essentially trying to do was move this data to another place, perhaps in a different format, and make sure it showed up there. That basic understanding of being an IT person (and not a computer science person) has led me happily around the globe and to interesting companies; initially with Pan Am, Volvo, Prodigy (yes, the email program I developed was new and innovative at the time, but essentially was still 'move this data from here to there, and make sure it arrives in a format we can all read'); then later to scores of F1000 companies as a consultant for technologies like MQ messaging, portal servers, web services, and basically any transactional delivery system environment.