

# How to Leverage Messaging to Achieve Consistently Low Latency

By: Mark Mahowald  
Copyright 2008, 29West, Inc.  
September 5, 2008

## Abstract

Of the messaging architectures available today, a pure application-to-application design typically results in the lowest latency messaging solution. However, there are many more issues to be considered to truly optimize end-to-end latency, such as the source of data, the design of the applications, the loss characteristics of the network, the ability of the system to withstand misbehaving software or hardware without triggering network storms, and even the division of data and applications according to their intrinsic latency requirements. 29West messaging can provide sub-50 microsecond one-way latencies, and provides you the tuning and control to ensure you get consistently low latency and jitter. Without these controls, design tradeoffs and network loss can have a large impact on actual application latency. This paper discusses these issues, offering advice on how to obtain consistently low latency in real-world environments.

## 1 Approaches in Use Today

Before 2004, the dominant solutions used in high performance messaging applications tended to be either server based, or daemon based designs. Starting in 2004 with the introduction of Latency Busters® Messaging (LBM) by 29West, a third model entered the market: a pure application-to-application design that did not rely upon servers or daemons as intermediaries to provide messaging functionality. In the last couple of years, additional vendors have adopted the application-to-application design model. Other vendors have brought hardware accelerated specialty message servers to the market. Each approach has performance and feature tradeoffs, and the network and application type can have a large impact on how well each model performs. It is not possible to cover each approach in great depth in this article, so only the key features and tradeoffs are summarized here.

### 1.1 The Client Daemon

Early high-speed messaging systems were developed in the mid-1980's as a part of the first digital trading floors - for example the Rich/Reuters TRIARCH system. These systems had to work with many constraints. The processing power, memory and networking capabilities were limited which led to the development of UDP-based broadcast protocols that used packet numbering to detect and recover lost data. The broadcast data was often constrained to a dedicated network, and all machines would see all traffic. To make this class of system work, each receiving machine needed a messaging daemon to discard the network traffic that was not of interest (not being subscribed) to any applications on that machine. This approach has worked well for years and is the most widely deployed legacy messaging model in the high performance, financial messaging market today. Due to the amount of data being broadcast/multicast, it is difficult to distribute data in this way across more complex networks. Typically, this problem was solved with gateway boxes being employed to isolate unwanted traffic and prevent it from crossing physical or logical network boundaries.

## 1.2 The Central Server Approach

The 1990's saw the creation of messaging systems built around a central message broker. Talarian introduced the SmartSockets model for messaging. The Java Messaging Service (JMS) was a standards-based publish/subscribe messaging model that came on the market in the mid 1990's as well. In these models all traffic would be sent to a central server (or a server cloud for load balancing) which would then forward traffic only to interested machines. Typically, all traffic would be sent point to point. This had a number of advantages and made it easy to distribute messages over complex topologies and through routers and firewalls (using TCP and unicast addressing). In addition, receiving machines would only receive traffic they were interested in, and the central server could provide value-added services such as store and forward capability, security, authentication, guaranteed messaging and "durable subscriptions." The disadvantage was that the server load increased as the number of subscribers, publishers, or message rate grew, and delivery of data to multiple receivers was not "fair" (i.e., one application would get the message first, and another would get it last). Besides the scaling issues, the routing of messages by a separate "message broker" or server adds considerable latency to the data path.

## 1.3 Application to Application Messaging

Application-to-Application messaging designs remove all components in the data path to minimize the latency between sender and receivers. 29West has successfully applied this model both to streaming messaging applications, where loss recovery is provided as long as the sender and receiver are still running, and to persistent applications – also known as "guaranteed messaging". Persistent applications must provide message delivery even in the event of sender or receiver failure or in the cases where receivers are not running when the message is originally sent. With durable subscriptions, delivery confirmation and a persistent store, 29West has been able to address this delivery model without imposing a store-and-forward model typical of other messaging solutions. By delivering the data to the receiver in parallel with delivery of the data to the persistent store, it is possible to provide a straight-through data path from the sender to the receivers while asynchronously processing stability notices from the store as messages become written to stable media.

## 1.4 The Server in Hardware

Specialty hardware messaging appliances have started to come to the market with the promise of greatly increasing the speed of a classical hub and spoke server design. By moving this server functionality into hardware, performance gains can be realized. By using specialty hardware to provide broader TCP fan out, these solutions offer the potential of broader, low-latency scale-out when compared to off-the-shelf servers using general purpose hardware and operating systems.

# 2 New Market Forces

A decade ago, completely automatic financial processes like electronic trading, black box trading, algorithmic trading, automatic execution engines, electronic exchanges, direct exchange feeds, crossing networks, and t+0 settlements were the dreams of financial industry technologists. Now these automated solutions are common.

Furthermore, the business volumes in terms of quotes, trades, and orders have been growing phenomenally, challenging market participants to increase computing power, messaging throughput, and network speeds. Similar challenges appear where ever completely automated financial systems connect directly with each other (i.e. no human response is involved in the process).

These automatic systems interact through high-speed messaging software. In fact, today a major limiting factor for many business processes is the capability of the underlying messaging infrastructure in terms of speed, latency, reach and reliability.

In addition, a successful messaging architecture must allow for run time tradeoffs between reliability guarantees, application size, and performance. These parameters may vary depending on the nature of the data, the needs of the applications as well as what networking environment the message data travels through. There are some streaming applications where potential data loss in peak conditions is preferable to re-sending data to allow recovery. For example it may be better to miss a foreign exchange price quote, if by doing so you ensure that a more current quote arrives with minimal latency.

The market growth has changed the view of low latency and high message rates. Where 50,000 messages per second and sub-5 milliseconds latency used to be very high performance messaging, today's fastest systems focus on sub-50 microsecond latency and millions of messages per second. This high-performance has also migrated from areas typically handled with multicast into more transactional or guaranteed message delivery applications. As network speeds increase and the cost of high performance end user devices drops, new classes of applications become practical. The recent Tabb group report on the low latency messaging market entitled "Faster than a speeding bullet, the new low latency messaging" documents this trend. For a free copy of this report, please visit <http://www.29west.com/products/tabb-report.pdf>

### **3 Key Concepts in 29West's Messaging Designs**

We are focused on the high speed, low latency delivery of very large amounts of message traffic. With this in mind, we have reduced multi-level architectures wherever possible. Where possible, we want to leverage the intelligence in the network to group traffic, identify available services and reduce the loads on individual receivers. The 29West designs remove the need for central choke points like message servers as well as extra data copying and processing by removing the need for separate messaging daemons to filter unwanted traffic. In addition, we want the sending application to be able to control data flow as well as protocol selection. For some applications, high performance and low latency are not critical and mass-market, legacy messaging solutions can perform adequately, but even in these applications, high performance messaging translate to a more efficient solution providing headroom for growth and reducing hardware costs.

29West Messaging products have a number of unique features including:

- Support for reliable UDP multicast, TCP/IP, latency bounded TCP/IP and reliable UDP unicast.
- Multi-layer group addressing constructs. This allows traffic to be divided via multicast groups as well as by topics. It also allows as many topics to be supported by the messaging layer as needed by the applications. The management and support of these topics is shared through the network, with no central server needed.
- Application-configurable delivery guarantees allow the sending applications to decide what should happen to slow receivers and how packet recovery should be handled.
- Completely symmetrical data flow: Any machine can be a sender, a receiver or both. The protocol scales across WAN boundaries and generates minimal recovery traffic.
- Support for persistent messaging models and features like delivery confirmation, durable subscriptions, late join support and flexible fail-over models, while still providing a direct application to application messaging model with typically, sub-100 microsecond one-way latency on commodity hardware.

- Ability to use rate controls to limit both recovery traffic and sending traffic. This can be used to provide stability for the network in the case of unexpectedly high peak data flows that could push a network without rate control into an unstable state caused by “NAK implosions” or “broadcast storms”.
- Support for latency-bounded TCP message delivery. LBM allows the sender to decide if it wants to wait for slower TCP receivers. For more information on latency issues with TCP, please see <http://www.29west.com/docs/THPM/>
- Automatic topic resolution. When a receiver subscribes to a topic, it receives all the information needed to receive the data (address, port, etc). At this point all message routing is done by the network hardware.
- Linking directly with your application. This provides a number of benefits, including:
  - Minimal data copies
  - Minimal “context switches” and number of processes involved in handling each message
  - No new entities to manage in the network, which results in less maintenance and upgrade headaches.
  - True application-to-application messaging; no extra processes or machines sitting between the applications.
- Detailed monitoring and traffic statistics accessible from the 29West messaging API as well as using standard network monitoring applications and the ability to tie this data into a customer’s existing monitoring framework via SNMP support.

The combination of many transport models allows the 29West messaging layer to provide the approach that best suits the application parameters and network capabilities. By providing extensive remote configuration support and management capabilities, we have reduced the time it takes to update an application and made it easy to scale LAN /WAN and other network boundaries.

## 4 How Do You Optimize True End To End Latency?

Messaging latency is a very interesting topic with many variables. True low latency is a system level concept, since to achieve it you need to make sure you have considered every step in the data flow and optimized it. The key to having a truly high performing system is to minimize the total latency from the point where data enters your enterprise to where data is being processed by the receiving applications. This latency ultimately affects your ability to execute as an enterprise on new market information. Some of the key links in the chain to be optimized are:

- Assurance that you have the lowest latency data when it enters the system. For example, taking direct market feeds instead of consolidated feeds means your firm gets to start its processing with fresher data.
- Efficient messaging layer.
- Efficient application use of the messaging layer.
- Ideally, a no-loss or a low-loss network
- Well designed receiving applications that can keep up with both expected and peak message flows. Some legacy systems can generate large amounts of extra recovery traffic pushing additional receivers into an overload situation.

- The ability to have a two-step delivery model if you have “low capability” receivers or want to use multicast on the backbone but also want machines utilizing TCP to be connected to the flow of data.

Without considering all these design points and having the flexibility to tune all the steps in the message delivery process, you cannot truly optimize your end to end latency. These items are not always considered in system design, and in many messaging designs they are not exposed and tunable for optimal performance. The goal of this section is to explore each of these points in more detail and explain the tradeoffs so end to end latency can be more effectively optimized. In all cases, the best step is to remove anything that is not critical to the data path. Once you have removed all the servers and daemons and extra processing, you get to the next set of design choices critical to a system performing consistently with very low latency and high performance

#### **4.1 Lowest Latency Data into Your System**

It makes no sense to optimize a messaging infrastructure to get rid of 1 millisecond of latency when the inbound data is 3 seconds behind the market. Direct feeds and careful design choices regarding where data is obtained pay big dividends here. Clearly minimizing the data latency into the system is important.

#### **4.2 Efficient Messaging Layer**

As was covered earlier, making sure there are no extra processes or servers between your sending application and the receiving applications pay huge benefits in messaging performance. Today there is some talk of hardware acceleration and specialty devices for message routing. We at 29West think these are great ideas if you in fact need a server. Go ahead and optimize it. But if you can deploy a system that does not need a server at all, you have effectively made the “server” latency zero and the throughput infinite, with zero cost - the best optimization possible. By getting rid of these devices we have provided the shortest data path giving you the lowest latency and best scaling in cases served well by multicast deployment.

#### **4.3 Efficient Application Use of the Messaging Layer**

Once you have removed the extra components in the data path, there are application design decisions that can impact latency. If you send every message as soon as it is ready, you are not using batching. This is great for low latency, but if your messages are small, you end up using many more network “packets” to send the same amount of data than if you used “batching” to group small messages together to fit more in one packet. This fine granularity might be perfect for low message rates, but at high rates the CPU and network overhead may become too high, requiring some form of batching. Similarly, if you choose to use rate controls on the sender to avoid sending too fast for receivers, when these rate limits are hit, you have purposely added latency to benefit the receivers who could not keep up. These are all design choices that a modern messaging system can expose and allow you to optimize for your environment.

If you have turned on batching so that small messages wait until a larger group (or batch) is ready to be sent to improve network utilization, a batching delay occurs for the first message that must wait for the batch to be ready. If your receiving application reads from a queue and it falls behind by many messages, you may see much larger sender-to-receiver latencies than the sub-50 microseconds the messaging system took to actually deliver the message. In many real deployment cases, the batching and queue delays may add tens or hundreds of milliseconds to the effective latency seen by your applications.

Frequently, application designers optimize for a one messaging layer design, and then plug a new messaging model underneath without considering how to optimize the overall performance with the new messaging layer. This approach can lead to some confusing and disappointing results and can explain why an integrated application has much worse performance than the messaging layer performance would suggest. We have seen this in performance results for integrated vendors using 29West messaging and exposing the integrated solution through abstraction layers. It is easy to inadvertently ‘de-tune’ performance, and if the goal is to make sure any one of many different messaging layers can be used, applications may not be able to take advantage of unique features of any one solution.

#### **4.4 No Loss or a Low Loss Network**

A network that has no loss will provide the best latency in any system. Recovery of lost packets takes time (hence adding latency) as a packet loss has to be detected, then recovered. With “in order” message delivery, all packets received after the loss are delayed until the lost packet is recovered. In addition, if you are not careful in your application design, you can have multiple bursty senders hitting a single switch port, causing over runs and loss in the switch, even when the network looks very lightly loaded. Monitoring plays a critical role here as lost packets are easily flagged in any monitoring system. At this point, the cause can be traced and if it is a design fault, the correct changes can be made. The beauty of modern messaging systems is they will faithfully recover loss and not complain. Without monitoring, you may have a high latency infrastructure and not realize it if all the traffic is being received and the machine and network loads are low. If you monitor your network and take steps to ensure you have low loss, you can be assured that your one way message transport latencies with 29West messaging will be under 50 microseconds on standard 1GigE LANs. 29West allows you to control the latency you experience in a recovery of a lost packet. These choices have tradeoffs as well and are covered in the next section.

#### **4.5 Intelligent Loss Recovery Design**

There are a couple of loss cases to consider and how you want to handle each one is very different. One case involves a capable network that can easily keep up with the message rate (spare network capacity). In this case, you might want to have every lost packet immediately generate a retransmission request. This setting delivers the recovery packet as soon as possible, maybe with 100 microseconds of recovery latency. This model can create extra load if the loss is spread across all receivers of a large receiver pool due to many retransmission requests for the same lost packet. This would be troublesome if the network was hitting an overload point. The extra traffic simply makes the loss problem worse. In this environment, delaying retransmission requests for a configurable interval (random across the defined time range) would increase the likelihood the generation of a small number of (maybe only one) retransmission requests for the whole group. The delay also means the retransmission comes a little later. Hopefully after the dissipation of the momentary traffic peak that might have caused the original loss. 29West provides these configurable parameters allowing you to get the behavior and latency you desire.

Again, careful monitoring and testing in your environment both in “normal” traffic and application patterns as well as in simulations of the “worst case” peaks can help you discover the balance for your network and your applications.

#### **4.6 Well Designed Receiving Applications**

No matter how well you have tuned your network and messaging layer, if the receivers fall behind and messages sit in long receiver queues, latency is added and the system can fall under pressure. If the receiving application falls far behind, it may overrun receiver buffers and suffer

loss. Again, making sure there are no weak links in the chain is the key design solution. 29West provides the ability to notify the application of queuing delays so that it can take “evasive action” or at least be aware of this additional source of latency.

#### **4.7 Two-Step Delivery Model**

In many systems there are high speed machines that can keep up with the backbone, and other receivers that can only connect via TCP but want to share data. We believe this is one spot in a well designed system that a message server is the right answer. Acting as a step-down box to provide TCP-based traffic to these machines and providing the conduit to exchange data with the backbone applications, this style of message server provides WAN or enterprise scaling while introducing minimal latency. Slower machines connect where they need to based on their capability and network topology.

### **5 Summary**

As applications evolve to require new data delivery models, higher performance and lower latency, the messaging layer must be adaptable and flexible in its delivery mechanism, and designed from the ground up for efficiency. Today, everyone is talking about low latency. There are a wide range of performance claims (including many by 29West). We feel the best way to cut through the clutter is to carefully understand the design goals, controls, cost and base performance of each solution. Once you have narrowed the solutions down, running a true pilot will provide you with the information on performance, support and reliability needed to make an informed decision. Once products are installed, careful monitoring can pay huge positive dividends in long term system health and performance. If you consider the data path end to end and balance tradeoffs between efficiency and latency, you can develop a system that performs best for your needs and design goals.

The team at 29West started with over 20 years of financial market data delivery experience, a clean sheet of paper and a single goal: to build the highest throughput, lowest latency, most flexible messaging products for the financial markets. In June of 2004, we announced LBM and our first customer, and have been setting the performance standard in the market ever since. With innovative design ideas, the absolute highest performance and customer focused integration expertise, 29West has seen strong market acceptance with over 120 firms worldwide running production deployments with 29West messaging. Our goal is to provide our customers with a solution that not only provides a strong performance boost, but also one more tightly coupled to the customer’s business and data delivery model.

We believe a ruthlessly efficient messaging design, combined with the control needed to optimize performance for your production environment, provides our customers with the best solution. The only way to be sure is to run a head to head comparison. In the last three years, 75% of the firms who performed an evaluation of 29West messaging software have purchased our software. We feel we have the best performance, and have clear design leadership. We encourage you to start a free evaluation of 29West messaging and see if our next generation design can provide a competitive advantage in your enterprise. Seeing is believing, and the only results that truly matter are the ones produced by your team with your applications on your network. We look forward to the opportunity to answer any questions you may have.

If you would like to learn more about 29West Messaging, please visit <http://www.29west.com/>, or contact us via e-mail at [info@29west.com](mailto:info@29west.com). We have offices in Chicago, New York, London and Tokyo and would welcome a chance to discuss your needs and see how 29West can help.