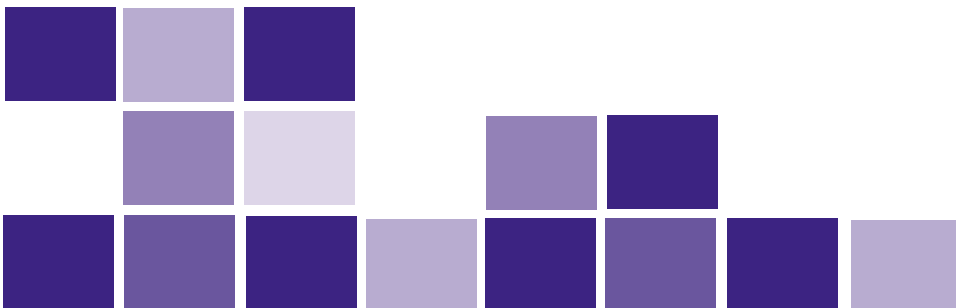




# WHITE PAPER

---

## *eCommerce Physical and Logical Connection Methods*



**Introduction**

We all know that eCommerce involves connecting systems for business communications methods. What we don't often consider is all of the methods, protocols, and communication agreements that must come together to

make it happen. It seems easy on the surface, but is very difficult in practice. Part of what makes it difficult is in the understanding of the various layers that must all come together appropriately for true communication to happen. Consider the following:

5	Content	We must all agree on common verbiage for the things we talk about.	Product numbers, designations for units of measure, numbering schemes, etc. Does 'Price' mean net after discounts or gross before discounts?
4	Topics	Conversations are useful only if both parties understand the topics of those conversations. Our respective systems can work only if we have a agreed upon set of topics.	Purchase Orders, Quotes, Changes to Orders, Acknowledgements, etc. Topics can be document?based ("This is a purchase order") or message?based ("Please acknowledge this order as received").
3	Format	We must agree on a common language, understood by both of us, in which to hold this conversation.	The actual format doesn't matter. Agreement and common definition does. XML, EDI, standardized flat file - all are formats for our communications.
2	Transportation	There must be some method by which we hold this conversation. Two people talk in person by vibrations in the air. They talk over the phone via electric pulses.	The internet has become the accepted transportation method, although private networks and VANs (Value Added Networks) are still very popular.
1	Connection	Somehow, the people holding a conversation must form a connection. Sometimes it is a shared connection (physical proximity). Sometimes the connection is through a network?we both plug our phones into the wall, but do not directly connect to each other.	There are a multitude of ways to connect physically to the internet and through that to an exchange. Generally speaking, eCommerce connections are not one-to-one, but rather individual connections to a centralized exchange.

Technically, there are actually more layers involved in a successful eCommerce implementation, but these five will certainly illustrate the complexity of the problem. Remember that all parties to a conversation must have at least compatibility, if not complete coordination, at all layers. This complexity is what has led to the growth of exchanges. In theory, a centralized exchange eliminates all of the coordination and negotiation that must take place at every layer for every participant. In practice, the exchanges available today are usually designed to tackle as little as two of these layers, forcing cooperation and coordination out to the individual participants about the transportation layer. As exchanges become more sophisticated, they are starting to tackle the problems about the transportation layers.

**Performance and the Concept of Real-Time**  
 Many implementations of eCommerce boast of real-time performance. In theory, this means that questions can be asked, results can be returned, and actions taken in an approximation of the timing that would be expected from a single, unified system. In other words, if you ask for a price, you get an answer in a timeframe compatible with your expectations. This is great in theory, but hard to pin down in practice.

First, you need to define what real-time really means. The web pages you see on your browser are being presented in real time, but that time is pretty darn frustrating if your connection or their server speed makes for a 20 minute refresh. Given all the ambiguities, the industry has generally settled on a definition that contrasts real-time and batch processing. With real-time, the expectation is

that two systems in conversation will react directly to each other's messages as fast as possible, with no predetermined timing delays in the system. Batch, on the other hand, denotes a predetermined timing system where one system wakes up and processes incoming messages on some preset schedule, then sleeps again until the next batch. Each is appropriate for a given problem. For example, real-time works well when a human is sitting at the other end of a given system, defining questions and waiting for answers. Batch processing works better when working with large amounts of data that are system-to-system based, with no human waiting for answers or results.

This discussion is important, as some of the decisions made in the above layers will have a direct impact on a system's ability to use real-time methods and the actual performance of that real-time system. But it is not just the decisions themselves that make a difference; it is also the consistency of those decisions between participants. This goes right to the heart of the purposes of an exchange.

In theory, two parties wishing to conduct eCommerce could get together, reach agreement on all of the above layers, and go about their merry way doing business electronically. This works really well as long as there are only two players. But, as the field grows and more people want to play, the coordination necessary to get everyone working exactly the same gets too complicated. This is where an exchange comes in. Common exchanges are, or should be, designed to accommodate all of the differences between the various participants at the various layers. If player A chooses a different format from

player B, it is the exchange's responsibility to transform the message from one format to another in the process of carrying the communication. This transformation, in theory, should be available at every single layer in the effort to let A talk to B. But all that transformation comes at a price. Each transformation carries with it a performance price. The more transformations, the heavier the price. So, if two parties really wish to converse very rapidly, with no performance delays, they should work together to reach the highest possible levels of compatibility at each layer, consequently minimizing the amount of transformation work necessary in the exchange.

### Performance and Layers 1, 2, and 3

The ability to get anywhere close to real-time conversations between two parties is highly dependent on the choices made at the very bottom layers of the conversational models. In particular, the choices made at these lower layers determine your ability to get close to real time. There are still decisions above these layers that will have an impact, but the wrong decisions here will ensure that you cannot handle real-time conversations.

To just complicate it a touch more, there tend to be natural affinities among the choices available in these lower layers. Part of this is history; part of it is technological compatibilities. There are no hard and fast rules, but the choices tend to either go with the grain of industry advances, or against that grain. These choices and affinities are detailed below:

Connection	Transportation	Format	Explanation
Java Message System (JMS)	Internet	XML	The most advanced combination; designed for real-time and compatible with future industry directions.
Hyper Text Transfer Protocol (HTTP)	Internet	XML, WML, or HTML	The method by which web pages work on the net. Designed for real-time but with a number of flaws. Better suited to display work than system-to-system message work.
Simple Mail Transfer Protocol (SMTP)	Internet or Private Network	Plain text (e-mail)	This is e-mail. It is very well suited to non-real-time (that was the original design), but can be adapted to real-time under controlled circumstances.
File Transfer Protocol (FTP)	Internet, VAN, Private Network	XML, EDI, Flat File	FTP is the most common connection method for EDI. It is well suited to batch work and occasional connections, but is not well suited to real-time requirements and full-time physical connections.
Java Database Connectivity (JDBC)	Varies	SQL	JDBC provides an ability to read or write directly to a database. This is useful between partners with a very high compatibility in systems and messages. It is not well suited to real-time.
Telnet	Varies	Keystroke replication and screen scraping	This method works by simulating a user interacting with the system through a normal terminal device. It can be well suited to real-time needs if the system being simulated is well suited to real-time interactions with the user.

Keep in mind that it is usually possible to mix and match almost all of the above. In theory, FTP can transfer almost anything. In practice, it is rarely used for XML-based messages. In theory, you can use Telnet to transport EDI documents, but in practice I've never seen it done.

### **A Quick Word on EDI**

People use the acronym EDI as a short-cut to describe an approach to all of the layers above. EDI (Electronic Data Interchange) is really a set of standards that have been implemented in some industries as a method of eCommerce. Generally speaking, EDI was implemented as an FTP connection method using value added networks (VANs) as transportation methods. Flat file layouts were defined and used for almost any message you could imagine between players. EDI has several virtues and several flaws, but is generally seen as a batch-oriented system for sending large transactions (large in size, not necessarily in volume) between participants. With EDI, agreement at the upper layers is generally left up to the individual participants. While this allowed EDI to flourish, it is also the characteristic that most frustrated smaller potential participants.

As business moves more toward real-time conversations and exchange approaches, they are tending to move away from EDI and toward other solutions.

### **Summary**

If the goal is to let everyone, regardless of technology sophistication and expertise, participate in our electronic community, then our exchange should support every possible choice at every possible layer. It should also, naturally, support conversion between approaches and content at every level. But all of this requires a more sophisticated exchange, and will demand a price both in implementation and its ability to deliver real-time performance.

If, on the other hand, the goal is to get as close to real-time performance as possible, then you should strive toward a limited set of choices to be adopted by all community players. This is true at every single layer of

the conversation. Every transformation at any layer will cause delays. The following guidelines will help:

- A full-time connection to the network mechanism is absolutely necessary. The most common network mechanism today is the internet, so everyone in the community should have a high-speed (not 56K), full-time connection to the internet. This connection should be to the internal network, not to an individual workstation.
- FTP, SMTP, and JDBC do not work well for real-time. By nature, these connection systems are designed for occasional connections and batch work. Think about your e-mail - it is designed to be delivered when you connect, not immediately when sent; so stick to the more immediate connection methods if you need real-time response.
- XML is the most common format for real-time work. It was designed with this in mind, and there are parsers on the market designed to meet real-time needs. While technically any format can be used for real-time, EDI and other flat-file formats are generally thought of as batch methods with batch parsing routines.
- The telnet and scripting combination is designed specifically for integrating systems with no predefined integration methods. By simulating a user of the system, it avoids the complexity of interfaces into and out of the system. It works only with character interfaces and assumes that terminal screens exist to provide for the business functions being automated.
- Real-time eCommerce assumes real-time back office systems. If, for example, there is no real-time ability to process an order in the back office system, then there will be no real-time ability to process an order received by eCommerce. It seems obvious, but this has proven to be the Achilles' heel of many eCommerce implementations.
- Real-time assumes no human intervention on the part of the receiving system. In other words, all work needed to process a request or answer a question must be completely automated and available directly from the receiving (responding) system. If, for example, prices are not automated in the system, then it will be impossible to provide real-time responses to price and availability inquiries.

As a last piece of advice, keep in mind that not every problem must be solved using real-time techniques. The transference of purchase orders and advanced shipping notices has been done in batch for years quite successfully. But when it comes to price and availability inquiries and similar, real-time is very appropriate. Examine each problem in turn and pick the best tools and methods to get the job done. This is as true in eCommerce as it is with any problem in life. An eCommerce exchange serving a community should

allow and support seamless connections to all its members with different connection methods and needs. It should have a translation and transportation engine that accepts a document in one format (XML) from the sender and translates into another format based on the receivers needs (e?mail) or vice-versa. The exchange should be capable to maintain a directory of services by each of its members and each topic and their communication needs.

