# Project Venezia-Gondola (A Framework for P-Commerce)

*Raymond Gao*
*Editor-in-Chief, P2P Journal*
*mailto:raygao@comcast.net*
*(972) 530-4562*
*5609 Harbor Town Drive*
*Garland, TX 75044*

## Abstract

A novel project named Venezia-Gondola (Project V-G) was presented, which describes an application platform that enables the activities of Peer-to-Peer commerce (P-Commerce). A new pattern called the Inverted Model-View-Controller (IMVC) pattern was claimed that is suitable for P-Commerce. The author also explains the principles of the Project V-G and possible architecture for future development.

## 1 Introduction

E-Commerce as a business model allows people to make transactions via the web conveniently. Normally, a business model indicates the need to make money. However, it is expensive to maintain website infrastructures - the conduit for majority of E-Commerce activities. As a result, users will have to pay for services. Additionally, the complexity of the centralized website administration requires good content management services for serving both static and dynamic contents. E-Commerce extensively depends on web-browsers and HTML pages for the user interface.

Project Venezia-Gondola (Project V-G) proposes an alternative architecture – a decentralized commerce model. Project V-G cleverly leverages the P2P network as a conduit for commerce and greatly minimizes infrastructure costs, e.g. setting up and maintaining server farms, paying for high bandwidth pipes, installing applications, and administrating the website, etc. Project V-G also provides extended commerce capabilities, e.g. supports for "traditional online buying & selling" as well as "bartering" and "donation" of goods and services. It is highly customizable and can be used by any companies or individuals. It uses the open-source model. Users and developers can build simple plug-ins rather than developing the entire application from scratch. In fact, we encourage developers to write enhancements (pluggable modules) for the project. The unique thinking presented in the project may even cause a shift in the online commerce model from a "cost per transaction" model to a "subscription for services" model.

### *1.1 The Case for P-Commerce*

#### *1.1.1 The N-Tier Architecture*

The N-Tier architecture is widely accepted for building E-Commerce websites. The N-Tier architecture separates presentation, business logic, process management, persistence (database), and integration into separate layers. N-Tier architecture often have many components, e.g. application servers, web servers, portal servers, directory servers, workflow and integration engines, databases, etc. (See Figure 1) HTML/XML, CSS, Javascript, and DHTML technologies are staple technologies for providing front-end and graphic user interface (GUI) services. The business logic tier depends on specialized middlewares. The catalog is stored in the database. And, integration adaptors retrieve content from backend Enterprise Information Systems (EIS).

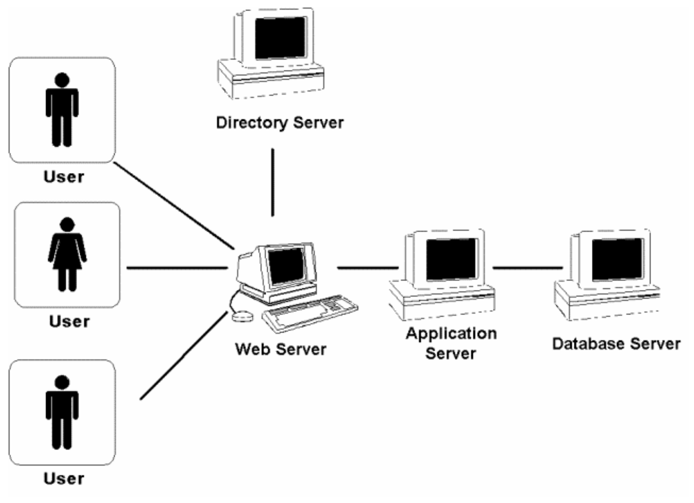

Figure 1 The tools for E-Commerce

### *1.1.2 Weakness of Centralized Models*

The current practice for building E-Commerce website depends on centralization. That practice emphasizes managing content and transaction logic from a single point. That practice can potentially lead to a "single point of failure" situation. Centralization may also mean that user's local settings are overridden by central policies. Sometimes those global policies are not appropriate to the local situation and cause inconvenience. Integration is another issue. It can become difficult to integrate two or more complex websites that were initially designed as stand-alone units. That situation is worsened by business' need to produce positive Return On Investment (ROI) on projects.

Online commerce requires reliability, availability, and scalability. When a customer has trouble with a down website, he/she may never come back. However, maintaining a high performance websites for large number of users can be an expensive and challenging task. The infrastructure alone can cost a lot of money for software, powerful servers, bandwidth, and a team of specialists, system administrators and support people. These large centralized systems also need to worry about computer viruses, professional/hobbyist hackers, worms, denial of service (DOS) attacks, and illegal snooping of theirs. Eventually, all those costs trickle into every business transaction. Users will have to pay for those centralized services.

Web-based applications are resource-intensive. For every click, a new page needs to be generated and sent to the user. Taking an online catalog application for an example, the catalog is re-loaded by every user and often multiple times. Even with caching of web pages, this is still a huge waste of bandwidth that must be supported in enterprise data-centers and equipments. All those waste translates into cost for the business who passes onto users. Now imagine a fellow trying to giveaway his used sofa or sell his car. That person must either build his own site or use someone's hosting service, which is not free by any means. And, that cost may cause some people to forgo those activities.

## *1.2 Why use P2P Instead of Client-Server?*

Centralized website can be very complex. And, complexity drives up the cost. Building a simple Business-to-Business (B2B) website can range from hundreds of thousands of dollars to millions. On the other hand, Peer-to-Peer (P2P) is very simple. Each peer behaves as both a client and server. Simplicity means less maintenance and consequently is cheaper.

P2P also minimizes the effect of DOS attack. Internet connects millions of homes and businesses. At each access point there maybe one or more computing devices. Each device has its own processor, storage, memory, and network connectivity. And, each device maybe running a slightly different operating system and has different virus scanners. In the event of a DOS attack, only a few computers will be affect. The rest can continue to function normally. P2P network is more resilient than the client-server architecture.

## *1.3 What is P-Commerce?*

Peer-to-Peer Commerce (P-Commerce) is an alternative model to centralized commerce. It is based on the ad-hoc relationship between individual participants. It uses the P2P network for infrastructure.

There is a saying, "the network is the computer." Project V-G utilizes the P2P network as a conduit for commerce. In effect, "**the network becomes the market**." (See Figure 2)

A transaction by nature is transient. It records an event-in-time between two or more participants. Since a transaction is a temporary phenomenon, wouldn't it be more cost effective to focus on that event than investing large sums of money on a centralized website? By giving people autonomy, participants will be in control instead of an arbitrary central authority. The business process becomes more efficient as well. Under the P-Commerce model, buyers and sellers rather than just the seller control a transaction. P-Commerce (P-C) becomes an equalizer for both parties.

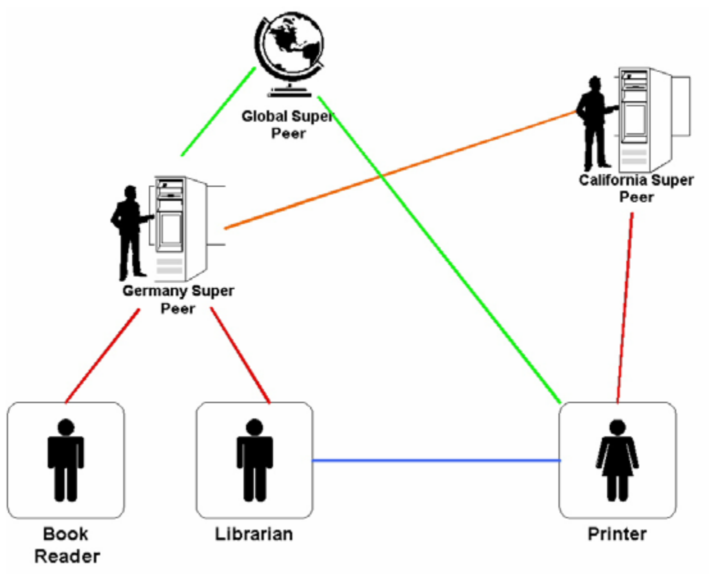

*Figure 2 P-Commerce Model*

P-Commerce has many other advantages.

- Users have choices on selecting which rich client for graphic user interface (GUI), which enables more fulfilling experience than with thin clients and browsers.
- Buyers and sellers can have real-time conversation before, during, and after the transaction.
- Multiple parties can engage in conversations and share information in both public and private venues.
- It is easier to implement agent-based computing, especially mobile agents. Mobile agents require tight security when done in a server. With P-C, the agent interaction is between the agent and the target.

### *1.3.1 XML and P-Commerce*

XML is a key component to P-Commerce. Because of the self-documenting nature and standard schemas available, XML enhances and eases compatibility between applications written by different vendors or for different purposes like the difference between a seller, a searcher, and a buyer. Imagine a seller that creates a schema for selling cars. The schema can contain information about options, performance, and the condition of the car. A buyer may not have a plug-in to examine the data from the seller, however they can look at the data with a standard viewer that converts the XML to a humanly readable form. The user can also use the tagged content in the XML to build search agents that can locate other sellers using the same or similar schemas. (Association)

### *1.3.2 Benefits*

P-Commerce empowers people. P-Commerce is more flexible for the seller, prospective buyers (searchers), and the buyer. Sellers can create rules similar to a centralized web-based system, but there is more flexibility to how you sell by using plug-ins that apply specifically to your product and style of sale. Searchers and buyers also have increased flexibility by creating agents for both searching and buying products or even bidding on products.

P-Commerce is very flexible and has several technical advantages. It allows direct interaction between participants. It more accurately models a transaction where peers at the network edge interact; the “transaction logic” and “content” are distributed. The P2P network can also scale better than the centralized architecture. P2P can circumvent “the single point of failure” problem in some cases. Individual peers can make decision on whether to use a thin or a full-feature application.

P-Commerce helps to cut down fees charged by the middleman in a business transaction. It allows direct interaction between sellers, buyers, and shoppers (a person who is looking and has not yet made his/her mind.) Because the true source of a product or service is exposed, there will be more opportunities and competition. Competition helps to drive down costs and to stimulate new innovations.

P-Commerce can also build a network of referrals (friends and partners) to help cross-sell goods and services. The recent success of "Friendster" and "Plaxo" further validates the value of social network. Using the market for pedigree dog as an example, smart buyers are interested in the quality of the dog, which includes the dog's lineage and awards the dog's parents have garnered at dog shows and competitions. A good buyer would reserve a puppy from a planned litter than going to puppy mills. Breeders also need to locate other dogs to add to their stock or to use as studs. In those situations, referral is a more powerful and reliable venue than direct marketing. In many cases, the logic behind those activities will be too complex for traditional E-tailers and must be addressed on an individual basis.

P-Commerce encourages creativity. It allows non-monetary based transactions, i.e. bartering (exchange & trade), goodwill (giving away merchandises and services based on certain criterions). It allows business to come up with imaginative new services and not feeling pressured by price wars. Quality and originality become more important bottom lines.

### 1.3.3 *Project V-G – A New Direction In Commerce*

Project V-G is the first P2P application framework aimed at a general-purpose commerce system. It adds "bartering" and "goodwill" functions (Figure 3) and additional creative business processes. It has three components: a P-Commerce network called "Venezia Network", a P-Commerce engine called "Venezia", and a graphic user interface call "Gondola". Project V-G is experimenting with a new computing model called the Inverted Model-View-Controller (MVC) pattern. (See section 2)

Project V-G is also an open-source project to promote its expansion into various markets and business models. Developers can write their own plug-ins that extend the system to add new product categories, methods of making a sale or bidding, and extend the way that a user searches for and buys products.

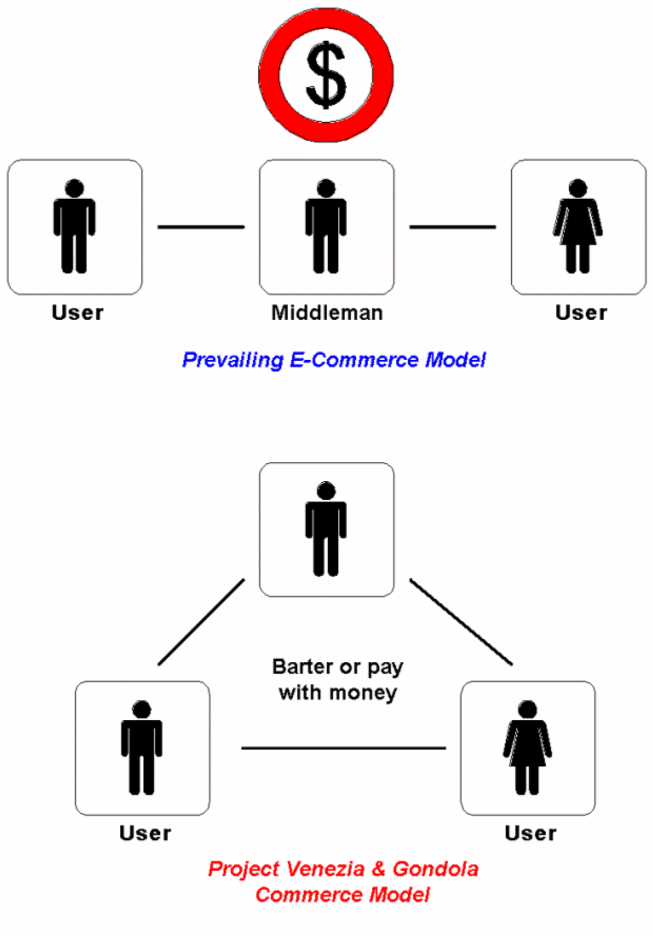
$
User
Middleman
User
Prevailing E-Commerce Model
Barter or pay with money
User
User
Project Venezia & Gondola Commerce Model

*Figure 3 Comparing Two Online Commerce Models*

## 2 P-Commerce, Model and Patterns

After analyzing several P2P networks and surveying general peers behaviors, we have decided to use the Inverted Model-View-Controller (IMVC) pattern for Project V-G.

### 2.1 *The MVC Pattern*

The Model-View-Controller (MVC) pattern has its roots extending back to IBM mainframe computing environments. The MVC pattern separates a complex application into three tiers. Those tiers are Model, View, and Controller. The Model represents the underlying data-store as well as access, update, manage, and delete functions. The View component displays those data in a useful format for the user. And, the Controller translates user actions and dispatches appropriate methods on the Model. The MVC pattern shields end-users from making system-level changes to the data-source, i.e. dropping tables, changing schema, shutting down database, etc.

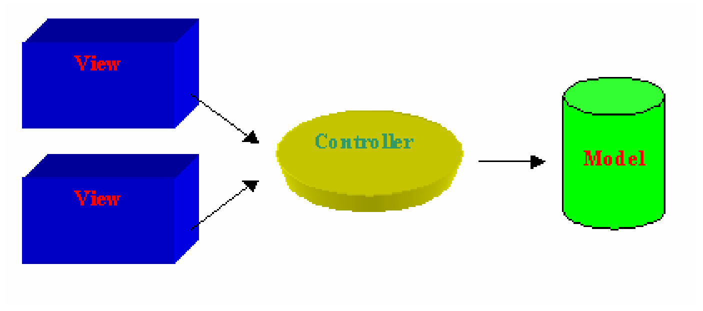

*Figure 4 The MVC Pattern*

### 2.2 *Patterns observed from P2P Applications*

The P2P network is based on ad-hoc relationships. Content is dispersed throughout the entire network and not aggregated on a single web-server. We reviewed articles about leading P2P applications, e.g. Gnutella, Napster, Limewire, and found some general patterns.

#### 2.2.1 *Reciprocal Relationship*

Peers are independent entities. Each peer is a self-contained unit that is both a client and a server. Peers make individual decisions of whether to consume content, to add new content, to route information, or to observe activities in the system. The interest in the system is similar to a publish and subscribe model where some peers are the creator of data, services, or transactions while others are the consumers of data, services, and the subject of a transaction.

#### 2.2.2 *Browsing vs. Providing Information*

The most popular activities on a P2P network are “searching”, “browsing”, and “downloading” content. Some studies have shown that only 10% of altruistic individuals actively add new material. There are some similarities in the online commerce situation. Most people are shoppers. It takes time, money, and effort to set up shops online. If a person only has one or two items for sale, giving them away (donation) maybe easier than trying to setup a shop. Merchants are people who frequently sell things online and manage to recoup initial costs and to make a profit. Entrance barriers cause there will be more buyers online than merchants. If we use the market-driven model, excess demand over supply will provide incentives for people to get involved, else the market has diminished value for producers.

### 2.3 *The Invert MVC Pattern*

In the P2P network, peers can freely add new content, provide comments, and summarize findings. If one imagines that the P2P network is a large data source, then one can think that peers provide contents for “tables”, peer-groups memberships are analogous to “database schemas” for grouping several peers together.

The Inverted MVC pattern is a derivative of the MVC pattern. A key difference between the MVC pattern and the IMVC pattern is the “access control”. The MVC patterns shields end-users from making direct changes on the data-source, e.g. shoppers cannot randomly add new products to a web-store’s catalog or delete product listings. In the P2P network, the data-source is exposed to the public. Anyone can add new content and even annotate meta-information, like indexes, to affect routing and items’ hit ratios. Because any peer can act as a server, occasionally, peers can provide conflicting data, i.e. Peer A advertises the availability of version 1.0 of a software, Peer B advertises the availability of version 1.0.1, and Peer C advertises version 1.0.1 with Beta service pack. Thus, a good P2P application architecture should plan for the inconsistency in the data model.

On a traditional website the web-site administrators are responsible for setting up its indexing engines and permissible filters. In a P2P network, there is no centralized system administration. Each user sets up his/her own filters and has a “window” into the “network data-source” via his/her local application. (See Figure 5) A peer sees and works with one “view” at any single point of time. Because the overall data-model can become inconsistent, two peers may get different results with an identical query.

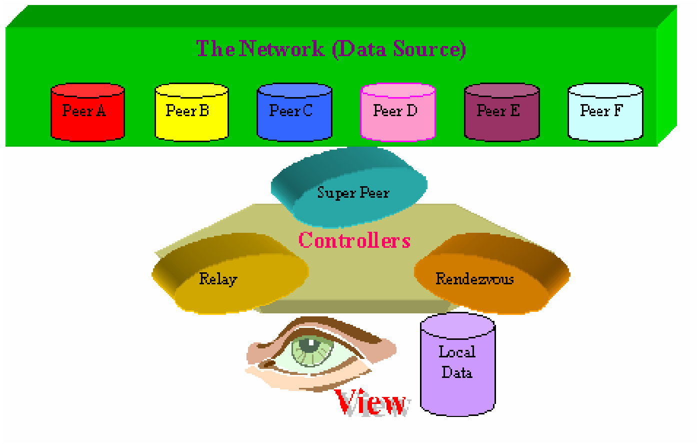

*Figure 5 The Inverted MVC Pattern*

Several P2P platforms have introduced the "super-peers" concepts, e.g. Rendezvous Peer for JXTA, reflector for Gnutella, and in FastTrack. Super-peers are "aristocrats" of a P2P network and provide basic peers with services, e.g. routing, proxy through firewalls, and caching service of frequently used items. The more clients a super-peer has, the more influence that super-peer has in a P2P network. Clients of that super-peers often develop similar "perception" even though that "perception" may or may not match the overall network data model which itself maybe inconsistent at times. The super-peer exerts "controls" by having many dependent clients. This shows a difference, in a P2P network the "controllers" are distributed; and in the client-server model that is centralized and is bundled with the "data model". The membership in peer-groups provides additionally form of filter and control.

| | **MVC** | **IMVC** |
|---|---|---|
| **Model** | Buffered from direct modifications | Externalized, anyone can affect the "model", i.e. add content and annotate meta-models |
| **Logic & Control** | Centrally managed | Decentralized and competing |
| **View** | Managed by the "controller", different "views" match the same "data- | Different "views" may reflect totally different "data-sets". |
| **Application** | The consistency of the "model" is absolutely essential. The "controller" reinforces the data-model consistency. | Inconsistency of the data-model should be planned. The "view" drives end-user behaviors. |

### *2.4 Suitability of the IMVC pattern*

The Inverted MVC pattern helps us to understand how P2P applications work and provides us a mental model that separates the data, the presentation, and the process flow. A model usually has two aspects, structure and behavior. The IMVC structure is loose on the data-model whereas the MVC model is tight on the data-model consistency. In the MVC pattern when the database is corrupted, the entire application is hosed. A P2P network is designed to handle cycles of disorder and restoration. Every time a new peer joins and leaves the network, the "data model" changes. In a large network with thousands of peers, cycles can fluctuate rapidly. The use of super-peers may help to smooth out fluctuations and bring stability to a P2P network and improve efficiency.

P-Commerce allows peers making transactions directly. When that happens, the transaction logic becomes decentralized. Under the MVC pattern, the transaction logic resides on a central website and is associated with the “model” and mediated by the “controller”. In a P2P transaction, that is no longer true. For example, two buyers are trying to purchase a same red Ferrari from a dealer. When the first buyer agrees to the transaction, the second buyer may not become aware of the situation until he/she receives a notification from the seller. However, the seller has no obligation to let the second buyer know until the second buyer asks. In a slightly different situation, the first buyer decides to abandon the transaction, does that leave the seller hanging and miss the opportunity to sell to the second buyer? In a third situation, when the first buyer withdraws from a bid, should the second buyer be allowed to lower his/her bid?

A possible solution to the above problem is to let the seller bear the communication responsibility with buyers. The seller can behave like a message queue and create two bi-directional pipes to two buyers. But, who is going to check the integrity of the seller? Alternatives one can use a super-peer to validate activities. However, it causes centralization of the network.

## 3 Project Venezia-Gondola’s Goals and the Design

### 3.1 *Solution Statements*

Project V-G is a P-Commerce application framework. It allows participants to conduct various decentralized business activities, e.g. buy, sell, haggle, barter, bid, and goodwill. Project V-G believes in that self-governing marketplaces are very efficient. Different products and services have different characteristic. Likewise, trading and/or bartering should have different guidelines. Having a single marketplace for trading everything may cause excess regulations and impede normal business activities.

Project V-G provides tools (network services and toolkits) and encourage establishing multiple P-Commerce marketplaces. Those marketplaces can have competing, cooperative, inter-dependent, and consolidated relationships, as well as be self-standing. It provides an alternative to the mono-marketplace approach because it is too complex to for a single website to write and monitor all business rules.

### 3.2 *Architecture Overview*

Project V-G three modules. The Gondola module functions as the Graphic User Interface (GUI), a façade to the underlying engine. The Venezia module is the core that enables P-Commerce. And, there is a third module for importing and parsing business rules.

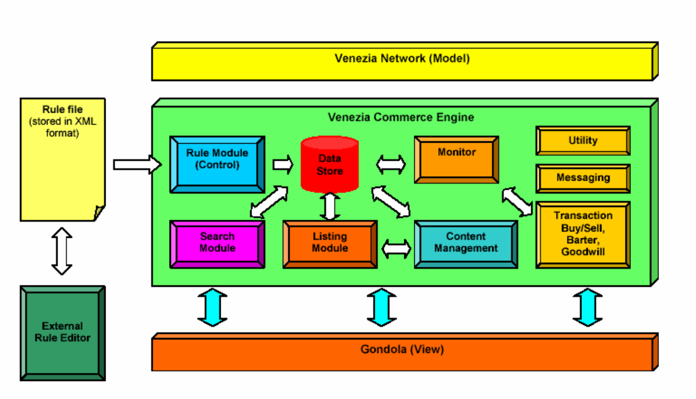

*Figure 6 Project Venezia-Gondola Technical Architecture Model*

The architecture follows the Inverted MVC pattern, whereby the “Model” is the Venezia P-Commerce Network. Many users will spend majority of their time browsing rather than adding new listings. The “View” is the Gondola module, which acts as the “windows” into the network. And, business rules are used to control the flow of information and guide the user’s interaction with the system. Business rules can be jointly edited and are stored in the XML format. Applications create “monitors” from importing business rules.

### 3.3 *Typical P-Commerce Processes*

Certain processes are recurring across most online commerce activities. Those processes can be grouped into following categories - Registration, Searching & Browsing, Listing, Transaction, and Feedback.

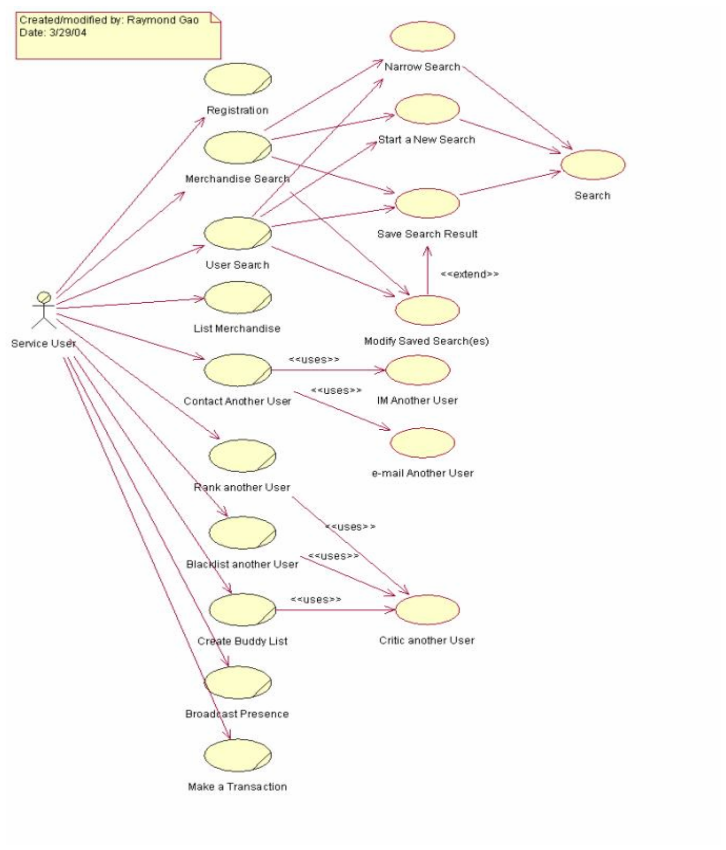

*Figure 7 Project Venezia-Gondola's Common Capabilities*

Additionally, there are utility functions to support above processes, e.g. Peer communication, Logging of transactions, Post-transaction critic and other miscellaneous activities.

### *3.3.1 The Registration Process*

Before a *User A* can participate in the Venezia Commerce Network, he/she must establish his/her identity. To register, *User A* needs to have Name (First-name plus Last-name), E-mail address, Instant Message ID, Phone number, Postal Address, and Zip Code. Currently, we are using e-mail address to identify each peer and that is set the first time the user runs the application. In the future, we intend to use pluggable module approach, allowing third party security-service provider add-ons.

### *3.3.2 The Searching Process*

There are two directories. One is a list of merchandises. The other is a list of users. There is also a "map", linking merchandise listings to owners. (See Data Model) Currently, we have implemented an area code based searching mechanism. We plan on adding geologic based searching features, i.e. longitude and latitude.

### *3.3.3 The Listing Process*

Adding a new listing involves generating a GUID for an item, confirming that ID, and propagating that listing in the network. The GUID creation is based on owner's UUID, item name, and a custom hash function. Each item will have following information.

**Product Specific**

1. Name
2. Category
3. Location (using area code, map segment, or zip code)
4. Keywords
5. Detailed description
6. Images & Sounds
7. Available Quantity
8. Price
9. Payment Options (Cash/Credit, Barter, Goodwill)

**Support Information**

1. Documentation, user guide, etc.
2. Shipping information
3. Contact information, e.g. phone number, e-mail, IM, etc.
4. Reference

### *3.3.4 The Transaction Process*

We are providing "straight purchase", "haggle", "auction", "barter", and "goodwill" functions in Project V-G. Only with P-Commerce, "giving away" and "exchanging" goods and services becomes valuable transactions model. Each transaction may have additional subtypes and be inter-dependent. For example, the auction process has the Dutch auction as a sub-category.

## 4 The Data Model

Project V-G currently uses HSQL for the local data-store. There are two primary tables. The “User” table stores the user specific information, such as “User ID”, “Name”, “E-mail”, “Interests”, etc. The “Item” table keeps track information about the merchandise. The “Item” table is linked to the “User” table via owner as a foreign key.

Secondary tables include “Review”, “Transaction”, and “SearchResult” tables. The “Review” table is used to keep track a user’s reputation. And, the “SearchResult” table stores searching information. Finally, the “Transaction” table is a temporary means to store the transaction information.

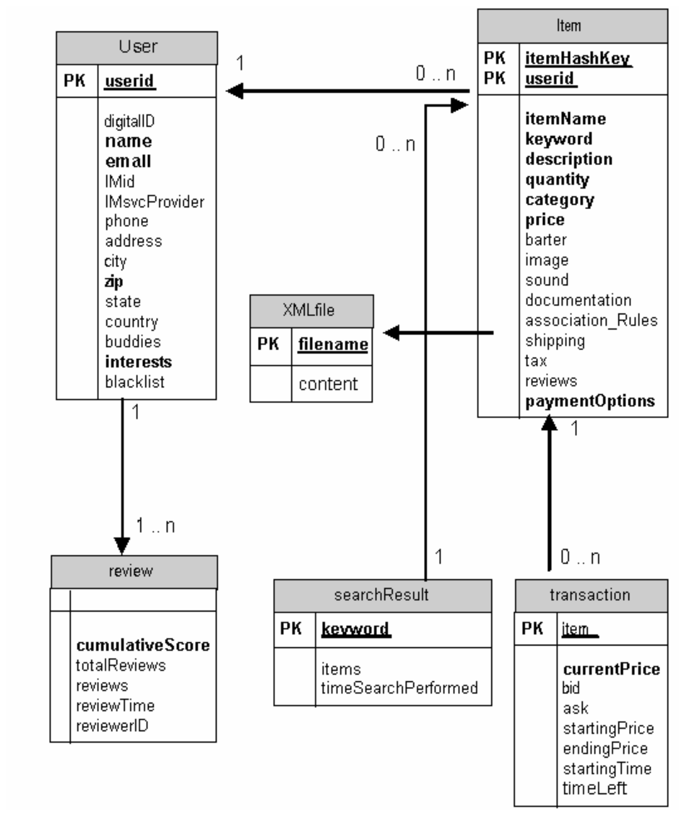

*Figure 8 Project V-G Data Model*

## 5 Observations

Project V-G is an ongoing project. During the implementation process, several observations were made.

1. Business processes for online commerce can become very complex. Section 3.4 has shown several typical P-Commerce processes. In reality, those processes barely meet the “tip of the iceberg”. There are many other types of processes and variations.

2. It is important to keep the client binary distribution to as small as possible. We are currently building this project on Sun’s JXTA platform. We have found that JXTA carries many dependent libraries (JAR and ZIP files). The size of the JXTA platform alone exceeds 6 megabytes. While broadband has become quite common nowadays, for international and dial-up users size remains an issue. We hope to strip down the JXTA platform and eliminate unused library files.

3. “Rendezvous pollution” poses a performance problem. Currently, the JXTA’s configuration utility allows any user to become a rendezvous host. Because a peer can disconnect anytime, it may leave its clients waiting for an extended time. Additionally, response from JXTA’s public Rendezvous hosts maybe slow at times. (It could take up to several minutes to resolve a connection.) To work around those issues, we have set up a private Rendezvous host that restricts access to a predefine peergroup. That private box runs BEA’s JRockit as its Java Virtual Machine on a Red Hat Linux 9.x platform. (In the upcoming release of JXTA, this problem will be fixed. But currently we can control this in our own network via our custom configuration.)

4. The conventional P2P searching method, broadcasting queries, is inefficient for a large network. It could potentially cause an avalanche of network activities. We are working on a referral model. Hopefully, that searching process will be more intelligent.

5. Security and trust are very important topics for P-Commerce and can become very complex. We hope that by using Pluggable Authentication Model (PAM) architecture we

can use third parties security modules with service provider interfaces (SPIs).

## 6 Summary

Project V-G is an ongoing experiment to demonstrate the validity of P-Commerce. It uses the IMVC model. Currently, we have two part-time developers working on this project. Given the complexity of P-commerce, we certainly feel that our project will benefit from community involvement. Perhaps, we can align our project with other open-source efforts.

We have a long wishing list for new functions. We hope that by opening up the source code, it will encourage people to get involved.

- Enhanced for the GUI, i.e. rich-client, skins
- Performance metrics tools
- Third party security modules and transaction validation services
- Interface with and save results to accounting software like Quicken
- Agents, i.e. automatically find and update new merchandise listings
- Management of buyers, sellers, and friends
- Pluggable product description modules vs. writing the description for each product from the scratch, for anything from cars to car parts to the pedigree of a pure bred dog.
- Pluggable payment system allowing COD, credit card, third party exchange, barter, or even non-monetary like airline miles or even community points.

**Special Thanks** to Daniel Brookshier and Johnson Gao for reviewing this article and giving me many useful inputs.

Raymond Gao is the editor-in-chief for P2P Journal. He founded P2P Journal in early 2003. He has a proven track record involving enterprise-level and distributed computing projects. He has spoken widely and is well published. He is currently a chief architect for Nokia's Business Infrastructure group. His other interests include music, oil painting, travel, and writing. He can be reached at raygao@comcast.net or (972) 530-4562.

## References

1. Project Venezia-Gondola's website (http://venezia-gondola.net)

2. "Project JXTA 2.0: Java TM Programmer's Guide", Sun Microsystems, Inc. May, 2003

3. Daniel Brookshier, Darren Govoni, Navaneeth Krishnan, "JXTA: JavaTM P2P Programming", SAMS Publishing, March 2002

4. Denis Urusov, "JOSE – A Java Open Source Exchange", P2P Journal, January, 2004

5. Luca Caviglione, "The "dark" side and the "force" of peer-to-peer computing saga", P2P Journal, January, 2004

6. Ben Strulo, "Middleware to Motivate Co-operation in Peer-to-Peer Systems", P2P Journal, March, 2004

7. Dimitrios Tsoumakos, Nick Roussopoulos, "Probabilistic Knowledge Discovery and Management for P2P Networks", P2P Journal, November, 2003

8. Joseph J. Bambara, Paul R. Allen, "J2EE Unleashed", SAMS publishing, 2002

9. White, K. Peterson, B. Lheureux, "New P2P Solutions Will Redefine the B2B Supply Chain", Gartner Research Note, 1 February 2003

10. R. Batchelder, "Peer Spaces: The Web Services Desktop", Gartner Research Note, 16 September 2002

11. Carl Lehmann, "P2P In B2B", Meta Group Research Report, 19, June, 2001

12. Raymond Gao, "To P2P or P2P Too: A discussion of Peer-to-Peer and Related Technologies", P2P Journal, July, 2003 (http://p2pjournal.com)

13. James W. Cooper "JavaTM Design Patterns", Addison-Wesley, 2001

14. Bo Leuf, "Peer to Peer: Collaboration and Sharing over the Internet", Addison-Wesley, June 2002

15. Mark Buchanan, "Nexus", W. W. Norton & Company, 2002

16. Sun Educational Services "Architecting and Designing J2EETM Applications", SL-425