JXTA technology for P2P: Project JXTA and its implications

Sudhindra Rao
11/03/03

Abstract

Peer-to-peer (P2P) systems are emerging as the latest alternative to perform distributed computing which includes file sharing, information exchange and service sharing. P2P technologies can adopt a network-based computing style that can improve the performance of information discovery, content delivery and information processing. P2P can also enhance the overall reliability and fault-tolerance of computing systems. P2P requires extensive services to enable the rapid construction of powerful applications. JXTA offers a framework to accomplish that. JXTA is a set of protocols that enable interoperability among distributed clients. The JXTA protocols do not dictate the implementation details such as programming language or operating system. Instead, they offer a high level map to guide implementations. JXTA details the standard services without limiting the implementation possibilities for a service.

1 Introduction

Broda [1] discusses six faces of the web. One of the most dynamic and promising among them is the device web where all the devices would be connected and would collaborate to use and provide services. Peer-to-peer networking is the basis of this web aptly termed as the pervasive computing web. The web is limited by its capabilities of indexing and storing information. Research has shown that two exabytes (2\times10^{18}) of information are produced every year, but only about 300 terabytes (3\times10^{12}) are published [2]. Moreover, Google claims that it can search only about 1.3\times10^{8} web pages. Thus finding information in real time is increasing difficult. Peer-to-Peer (P2P) technologies can adopt a network-based computing style that can improve the performance of information discovery, content delivery and information processing. P2P can also enhance the overall reliability and fault-tolerance of computing systems. A number of P2P technology based applications are available and become popular in a short time. Examples of such applications include ICQ, AOL, Napster, Gnutella, etc. which provide instant messaging or file sharing services. These applications have diverse characteristics and have been developed independently due to the lack of a common P2P infrastructure. The incompatibilities in these application have created vendor specific communities. Also lack of interoperability has stopped the collaboration between the communities. Project JXTA (Jxta for short) aims to bring a standard infrastructure to the P2P community to avoid duplication of effort and cause interoperability. Most of the P2P systems today offer services specific to networking protocols, operating systems and even depend on specific programming languages. JXTA technology is designed to develop such P2P services independent of operating systems, programming languages and development environments. Some important concepts pertaining to this model of P2P technology is presented here. A comparison with some known technologies is presented. A detailed discussion regarding each of the aspects discussed here can be found in [2, 3] and [4].

1.1 Related Work

With the discussion of JXTA a couple of other technologies need to be discussed to present the differences and uniqueness of each. A similar paradigm is presented with the Microsoft .NET technology. Microsoft .NET is an
integrated framework which aims and developing a rich set of applications, application interfaces that can be used as web services or P2P services. Microsoft .NET imposes a constraint on the P2P services offered requiring them to be operating on Microsoft Windows platform. As with the web services .NET does not provide interoperability with P2P services. Since .NET develops on the well known interfaces and data formats of the Microsoft platforms they are easier to adopt. Also .NET facilitates creating the services in any language of choice including Microsoft’s C#. As the data formats are based on XML and the use of SOAP as an information exchange protocol the approach is standardized. [4] provides more information on Microsoft’s .NET and its comparison with JXTA. Another such developer framework which needs to be mentioned here Mind Electric’s GLUE. GLUE is a Java based platform which allows a software developer to create web services from smaller parts. GLUE is a complete library for building and invoking distributed web services. It has a tiny foot print. GLUE is platform, protocol and transport-neutral. GLUE gives us a way of deploying P2P services using a web browser. As compared to .NET and JXTA GLUE is a software framework which can be used for development, whereas JXTA and .NET provide newer paradigms for providing distributed computing services using web or P2P. Krishnan [5] discusses the deployment of JXTA technology for providing P2P services. JXTA as a new paradigm in developing applications is detailed by Krishnan.

2 JXTA technology concepts

The JXTA technology which was a Sun Microsystems initiative, is now an open-source initiative and still evolving. Sun’s Project Jxta is a set of open protocols, that allow any device - from the toaster to a server - to connect, communicate and collaborate with any other device to provide, use or share resources. The resources can be anything from a file, message to a unique facility like printing or display. The Project Jxta(Jxta for short) pushes the capability of P2P technology from the well known messaging and file sharing to a resource level a lot of effort. A scenario with an existing infrastructure of Internet and systems administration is available easily. Jxta based tools can be used to discover new ways of utilizing the infrastructure to harness benefits without a lot of rework. [6] presents the characteristics of Jxta applications in situations where

- Centralization is not possible or not required
- Resilience is needed
- Massive scalability is needed
- Relationships are transient and ad-hoc
- Resources are highly distributed

A conceptual level of layering structure in Jxta can be shown as in Figure 1. Jxta is a framework which enables building any kind of Peer-to-Peer(P2P) services without being limited to file sharing.

At the bottom, the core layer deals with peer establishment, communication management such as routing, and other low-level activities. In the middle layer, a service layer handles higher-level concepts, such as indexing, searching, and file sharing. These services may not be commonly included as components in all P2P systems. At the top is the layer of applications which includes email, download, etc. Some features, such as security, manifest in all three layers and throughout a P2P system.

JXTA technology is designed to provide a layer on top of which services and applications are built. This layer is designed to be thin and small, while still offering powerful primitives for use by the services and applications. This helps maintain interoperability among competitive offerings from various P2P contributors and to providing maximum room for innovation.
JXTA technology is a set of protocols, each protocol being defined by one or more messages exchanged among participants. Like TCP/IP that links Internet nodes together, JXTA technology connects peer nodes with each other. Like TCP/IP, JXTA is also platform independent but it can use any transport protocol including TCP.

The following six protocols are currently defined:

- **Peer Discovery Protocol**: enables a peer to find advertisements on other peers, and can be used to find any of the peer, peer group, or advertisements. This protocol is the default discovery protocol for all peer groups, including the World Peer Group. Peer discovery can be done with or without specifying a name for either the peer to be located or the group to which peers belong. When no name is specified, all advertisements are returned.

- **Peer Resolver Protocol**: enables a peer to send and receive generic queries to search for peers, peer groups, pipes, and other information. Typically, this protocol is implemented only by those peers that have access to data repositories and offer advanced search capabilities.

- **Peer Information Protocol**: allows a peer to learn about the capabilities and status of other peers.

- **Peer Membership Protocol**: allows a peer to obtain group membership requirements, to apply for membership and receive a membership credential along with a full group advertisement, to update an existing membership or application credential, and to cancel a membership or an application credential.

- **Pipe Binding Protocol**: allows a peer to bind a pipe advertisement to a pipe endpoint, thus indicating where messages actually go over the pipe. In some sense, a pipe can be viewed as an abstract, named message queue that supports a number of abstract operations such as create, open, close, delete, send, and receive. Bind occurs during the open operation, whereas unbind occurs during the close operation.

- **Endpoint Routing Protocol**: allows a peer to ask a peer router for available routes for sending a message to a destination peer. Any peer can decide to become a peer router by implementing the Peer Endpoint Protocol.

JXTA technology defines a number of concepts: identifiers, advertisements, peers, peer groups, and pipes.
2.1 Identifiers

JXTA uses UUID, a 128-bit datum to refer to an entity (a peer, an advertisement, a service, and so on). The uniqueness of UUID in the local runtime environment can be easily guaranteed, but as the global state of the P2P system is not known uniqueness in the community is not guaranteed. One way to do so it to bind the UUID to the well known entity like the name or the network address. This binding has to be done during development. A server based or a serverless mechanism for registering these IDs can be developed.

2.2 Advertisements

JXTA technology defines a basic set of advertisements in XML documents that are used to publish the existence of a resource, such as a peer, peer group, service etc. This advertisement structure can be easily extended for publishing new services.

2.3 Peers

A peer is any entity that can speak the protocols required of a peer. As such, a peer can manifest in the form of a processor, process, machine, or user. Any peer does not need to understand all the protocols and can still perform at a reduced level if it does not support a protocol.

2.4 Messages

Messages are designed to be usable on top of asynchronous, unreliable, and unidirectional transport. Therefore, a message is designed as a datagram, containing an envelope and a stack of protocol headers with bodies. The envelope contains a header, a message digest, (optionally) the source endpoint, and the destination endpoint. An endpoint is a logical destination, given in the form of a URI, on any networking transport capable of sending and receiving datagram-style messages. Endpoints are typically mapped to physical addresses by a messaging layer. Such a message format is designed to support multiple transport standards. Each protocol body contains a variable number of bytes and one or more credentials to identify the sender to the receiver.

2.5 Peer Groups

A peer group is a virtual entity that speaks the set of peer group protocols. Typically, a peer group is a collection of cooperating peers providing a common set of services. The JXTA specification does not dictate when, where, or why to create a peer group, or the type of the group, or the membership of the group. It does not even define how to create a group. It does define how to discover peer groups using the Peer Discovery Protocol. There is a special group, called the World Peer Group, which includes all JXTA peers. Participation in the World Peer Group is by default.

2.6 Pipes

Pipes are communication channels for sending and receiving messages, and they are asynchronous. They are also unidirectional, so there are input pipes and output pipes. Pipes are also virtual, in that a pipes endpoint can be bound to one or more peer endpoints. A point-to-point pipe connects exactly two peer endpoints together. The pipe is an output pipe to the sender and input pipe to the receiver, with traffic going in one direction only from the sender to the receiver. A propagate pipe connects multiple peer endpoints together, from one output pipe to one or more input pipes. Accordingly, any message sent into the output pipe is sent to all input pipes. Any number of unicast and multicast protocols and algorithms, and their combinations, can be used.
3 Security Considerations

The security requirements of a P2P system are very similar to those of any other computer system. Given that JXTA is defined around the concepts of peers and peer groups, a security architecture could be envisioned in which peer IDs and group IDs are treated as low-level subjects (just like uid and gid), codats (code data) are treated as objects (just like files), and actions are specified operations on peers, peer groups, and codats. However, given that codats can have arbitrary forms and properties, it is unclear what sets of actions should be defined for them. It is quite likely that codats will carry along with them definitions of how they should be accessed. Developing a more concrete and precise security architecture is an active area of research. JXTA technology is a platform focused on mechanisms and not policy, hence it does provide not any security constructs but allows for their development with the applications. To allow maximum flexibility and avoid redundancy, JXTA technology typically does not force a particular implementation on developers. JXTA technology is neutral to cryptographic schemes or security algorithms. JXTA technology can sometimes satisfy security requirements at different levels of the system. As JXTA applications can be built over different infrastructures and protocols any proven techniques of providing security also can be used. One such implementation using Public Key Infrastructure is discussed in [7]. It also discusses a distributed trust model in P2P groups where the Peers collaborate acting as certificate authorities to establish trust between peers and groups.

More specifically, JXTA Version 1.0 attempts to provide the following security primitives:

- A simple crypto library supporting hash functions (such as MD5), symmetric encryption algorithms (such as RC4), and asymmetric crypto algorithms (Diffie-Hellman and RSA).
- An authentication framework that is modeled after PAM (Pluggable Authentication Module, first defined for the Unix platform and later adopted by the Java security architecture).
- A simple password-based login scheme that, like other authentication modules, can be plugged into the PAM framework.
- A simple access-control mechanism based on peer groups, where a member of a group is automatically granted access to all data offered by another member for sharing, whereas nonmembers cannot access such data.
- A transport security mechanism that is modeled after SSL/TLS, with the exception that the unidirectional pipe does not allow it to perform a handshake, a crypto-strength negotiation, or a two-way authentication on a single pipe.

The demonstration services called InstantP2P and CMS (content management service), which also make use of additional security features provided by the underlying Java platform.

3.1 Discovery Mechanisms

JXTA does not mandate exactly how discovery is done. It can be completely decentralized, completely centralized, or a hybrid of the two. In JXTA Version 1.0, it supports the following:

- LAN-based discovery: This is done via a local broadcast over the subset.
- Discovery through invitation: If a peer receives an invitation (either in-band or out-of-band), the peer information contained in the invitation can be used to discover a (perhaps remote) peer.
- Cascaded discovery: If a peer discovers a second peer, the first peer can, with the second peers permission, view its horizon, discovering new peers, groups, and services.
• Discovery via rendezvous points: A rendezvous point is a special peer that keeps information about the peers it knows about. A peer that can communicate via a rendezvous peer, perhaps via a pipe, can learn of the existence of other peers.

Rendezvous points are especially helpful to an isolated peer by quickly seeding it with lots of information. It is conceivable that some web sites or its equivalent will be devoted to providing information of well-known rendezvous points.

### 3.2 Propagation Scopes

JXTA does not mandate how messages are propagated. For example, when a peer sends out a peer discovery message, the Peer Discovery Protocol does not dictate if the message should be confined to the local area network only, or if it must be propagated to every corner of the world. The current implementation of JXTA uses the concept of a peer group as an implicit scope of all messages originated within the group. In theory, any scope can be realized with the formation of a corresponding peer group. For example, a peer in San Francisco looking to buy a used car is normally not interested in cars available outside the Bay Area. In this case, the peer would like to multicast a message to a subset of the default World Peer Group. A subgroup can be formed especially for this purpose, but it seems more convenient and efficient to perform the multicast without forming a new peer group. We can envision a number of approaches to solving this problem. For example, all messages can carry a special scope field that indicates the scope for which a message is intended. Any peer receiving this message can propagate it based on the scope indicator. Using this approach, a sending peer should be bootstrapped with some well-defined scopes. Further work is needed in this area.

### 3.3 XML

In theory, JXTA can be independent of any format used to encode advertisement documents and messages. In practice, it uses XML as the encoding format, mainly for its convenience in parsing and for its extensibility. Three points worth noting about the use of XML:

- JXTA can be modified to use any standard of data management and is not bound to XML.
- The use of XML does not imply that all peer nodes must be able to parse and create XML documents. For example, a cell phone with limited resources can be programmed to recognize and create certain canned XML messages, and still participate in a network of peers. To keep Version 1.0 small, we used a lightweight XML parser that supports a subset of XML.

### 3.4 NAT and Firewalls

The widespread use of network address translation (NAT) and firewalls severely affects the smooth operation of many P2P systems. It also affects the usability of JXTA. In particular, a peer outside a firewall or a NAT gateway cannot discover peers inside. In the absence of getting system administrators to let JXTA traffic through (say, by opening a special incoming port at the firewall or gateway), there are two rather obvious ideas to deal with this problem:

- Ask peers inside firewalls to initiate connections to peers outside the firewall.
- Set up peer nodes that operate like mailbox offices where traffic to a peer inside the firewall is queued to be picked up at a designated relay peer outside the firewall. The peer inside the firewall can initially reach outside the firewall, select a relay peer, and widely advertise this fact. Later, it can periodically contact the relay peer to retrieve messages. These are far from ideal solutions, and this is an active research area with lots of ongoing work.
3.5 Peer Monitoring and Metering

Peer monitoring means the capability to closely track a (local or remote) peers status, control its behavior, and respond to actions on its part. It is very useful when a peer network wants to offer premium services with properties such as reliability, scalability, and guaranteed response times. For example, a failure in the peer system must be detected as soon as possible so that corrective actions can be taken. It is sometimes better to shut down an erratic peer and transfer its responsibilities to another peer. Peer metering means the capability to accurately account for a peers activities, in particular its usage of valuable resources. Such a capability is essential if the network economy is to go beyond flat-rate services. Even providers offering flat-rate services can benefit from being able to collect data and analyze usage patterns. JXTA currently approaches monitoring and metering through the Peer Information Protocol, where a peer can query another peer for data such as up time and amount of data handled. Obviously, security is central to peer monitoring and metering. A peer may choose to authenticate any command it receives. It may also decide not to answer queries from suspect sources. A new project on monitoring and metering is set up on the Project JXTA website, and we expect to see lots of activities in this area in the very near future.

4 Recent improvements and implementations on JXTA

Project JXTA 2.0 specification defines some enhancements over 1.0. It defines the possibilities of building super-peer virtual networks. An enhancement over the Peer Membership Protocol is suggested in JXTA 2.0 specification (released recently) called as The Rendezvous Protocol (RVP). It is the protocol by which peers can subscribe or be a subscriber to a propagation service. Within a peergroup, peers can be rendezvous peers, or peers that are listening to rendezvous peers. RVP allows messages to be sent to all of the listeners of the service. RVP is used by the Peer Resolver Protocol in order to propagate messages. Rendezvous Walker [8] is an example of a virtual overlay network that can be built over JXTA. The virtual overlay network provides simple primitives to hide the complexity of the underlying physical network topology allowing peers to uniformly address any other peer on the network. Messages can be transparently routed, potentially traversing NATs and using different transport layer protocols. The JXTA network provides a default resolver service based on the rendezvous peers. The rendezvous peers maintain an index of advertisements published by edge peers via the Shared Resource Distributed Index (SRDI) service. The [9] describes the implementation of a loosely-connected DHT which is an improvement over developments like CHORD or CAN which require a rather stable peer infrastructure.

A list of active projects on JXTA can be found at [6]. An interesting approach in [10] shows how JXTA P2P technology can be used to perform distributed search on distributed networks. A collaborative network of JXTA search hubs that route relevant queries to relevant peers to retrieve data. Newer techniques of wide and deep search can be employed to deliver appropriate service/data for the queries. Possible applications of this JXTA search is foreseen in B2B networks, consumer websearch and extranet applications. The research in [7] shows how Public Key Infrastructure can be implemented over P2P networks for providing security. Also a technique for establishing a trust model between peers of a group or network are discussed in detail. The research discussed above indicates the JXTA is a flexible framework of protocols for implementing a network of distributed services and data. Also that the JXTA community is a developing one and enhancements to JXTA would be incorporated with experience.

5 Conclusion

JXTA provides a network-programming platform specifically designed to be the foundation for peer-to-peer systems. As a set protocols, the technology stays away from APIs and remains independent of programming languages. This means that heterogeneous devices with completely different software stacks can interoperate through JXTA protocols. JXTA technology is also independent of transport protocols. It can be implemented on top of TCP/IP, HTTP, Bluetooth, Home- PNA, and many other protocols. Like the Unix shell, the JXTA Shell helps users learn a
lot about the inner workings of JXTA during the process of writing scripts. The open sourcing of JXTA technology through http://www.jxta.org, on 25 April 2001, marked a significant turning point for Project JXTA. Many developers have already started working to advance JXTA and related technologies. Apart from these topics, there are many issues that need substantial research and development work.

References


