Project JXTA: A Loosely-Consistent DHT Rendezvous Walker

Bernard Traversat, Mohamed Abdelaziz, Eric Pouyoul
Bernard.Traversat@Sun.Com
Project JXTA
Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303, USA

Abstract

The open-source community Project JXTA defines an open set of standard protocols for ad hoc, pervasive, peer-to-peer (P2P) computing as a common platform for developing a wide variety of decentralized network applications. The following paper describes a loosely-consistent DHT walker approach for searching advertisements and routing queries in the JXTA rendezvous network. The loosely-consistent DHT walker uses an hybrid approach that combines the use of a DHT to index and locate contents, with a limited range walker to resolve inconsistency of the DHT within the dynamic rendezvous network. This proposed DHT approach does not require maintaining consistency across the rendezvous network, a stable super-peer infrastructure, and is well adapted to ad hoc P2P network with high peer churn rate.

1. Introduction

Project JXTA[1,2,3,4] is an open-source project (*www.jxta.org*) originally conceived by Sun Microsystems, Inc. and designed with the participation of a growing number of experts from academic institutions and industry. Project JXTA defines a common set of protocols for building Peer-to-Peer (P2P) applications to address the recurrent problem with existing P2P systems of creating incompatible protocols. The main goal of Project JXTA is to define a generic P2P network overlay usable to implement a wide variety of P2P applications and services. The Project JXTA platform provides core building blocks (IDs, advertisements, peergroups, pipes) and a default set of core policies. Developers can replace any of the default policies by plugging their own policies. For instance, the platform can be customized to use new routing, membership or search policies.

Section 2. provides a quick overview of the JXTA virtual network abstractions. Section 3. introduces the Resolver service and rendezvous peers as uniform resource locator on the JXTA network. Section 4. describes and discusses the loosely-consistent DHT, and the limited-range rendezvous walker. Finally, Section 5. covers implementation status and future directions.

2. Project JXTA Virtual Network

The Project JXTA protocols establish a *virtual network* overlay on top of the existing physical network infrastructure (Figure 1.). The Project JXTA virtual network provides simple primitives to hide the complexity of the underlying physical network topology (firewalls, NATs, mobile IP and non-IP proximity networks [5]) allowing any peer to uniformly address any other peer on the network. Every network resource is uniquely identified[1] and addressed independently of its network location creating a virtual indirection between the logical address and the physical address of network resources. Messages are transparently routed, potentially traversing firewalls, NATs, and using different transport protocols (Bluetooth, IrDA, TCP/IP, HTTP) to reach their final destinations. The protocols standardize the manner in which peers discover each other, self-organize into peergroups, advertise and discover network resources, communicate and monitor each other.

2.1 Message Routing

Project JXTA assumes an ad-hoc, multi-hop adaptive network. Connections may be transient. Peers may come and go at any time (high churn rate). Message routing is nondeterministic. Routes may be unidirectional (NAT, firewalls), and may change rapidly as the network topology changes.

2.2 PeerGroups

Peers in the Project JXTA network self-organize into *peergroups*. A peergroup represents an ad hoc set of peers that have a common set of interests, and have agreed upon a common set of policies (membership,

1. Each resource is assigned a unique ID. The reference implementation is using 128-bit UUIDs.

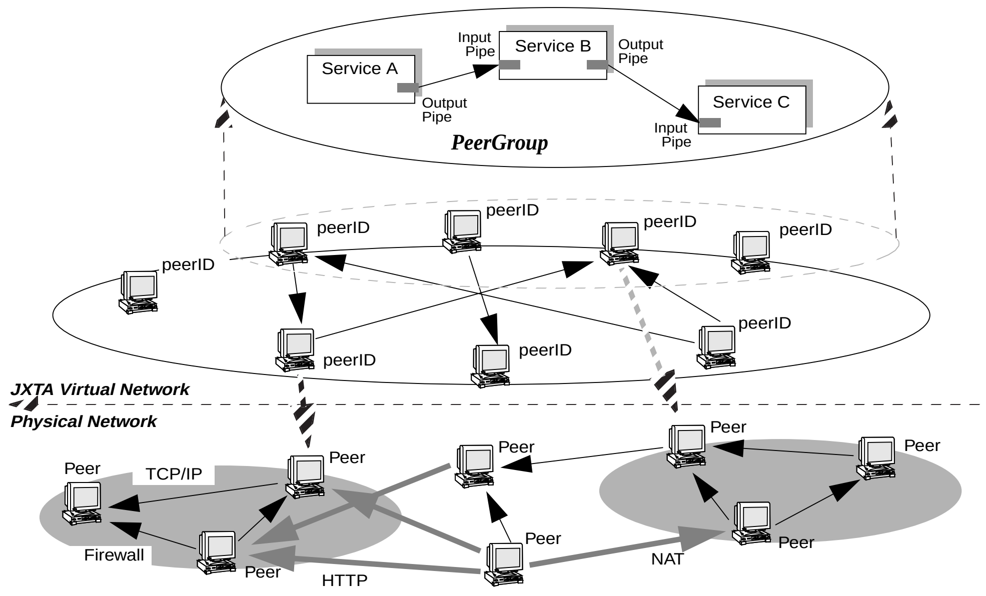


Figure 1. The Project JXTA Virtual Network

routing, searching, etc). The JXTA protocols only describe how a peergroup is created, published, discovered, and how different policies can be plugged into a peergroup. Peergroups form logical regions whose boundaries do not necessarily reflect the underlying physical network topology, such as those imposed by routers or firewall domains (see Figure 1.). Peergroups enable subdividing the JXTA network into virtual regions, providing scoping and scaling mechanisms for restricting the propagation of discovery or search requests. A peer can belong to as many peergroups as it wishes, and creates as many peergroups as it needs. Peergroup creators can create peergroups to match their specific requirements (centralized versus decentralized, deterministic versus non-deterministic, etc.).

2.3 Advertisements

All network resources in the Project JXTA network, such as peers, peergroups, routes, and services are represented by *advertisements*. Advertisements are language-neutral metadata structures represented as XML documents. Peers cache, publish, and exchange advertisements to discover and locate available resources. Peers discover resources by searching for their corresponding advertisements. All advertisements are published with an expiration *lifetime*.

3. Resolver: Universal Resource Binding

The Project JXTA network uses a universal resource binding service called the *resolver* to perform all resolution operations found in traditional distributed systems, such as resolving a peer name into an IP address (DNS), binding an IP socket to a port. In Project JXTA, all binding operations are unified under the search of one or more advertisements.

The Project JXTA network provides a default resolver service based on *rendezvous* peers. Rendezvous peers are peers that have agreed to index other peer advertisements to facilitate the discovery of resources in a peergroup. A peergroup can have as many rendezvous peers as needed. Each peergroup has its own set of rendezvous peers for scoping purposes. This ensures that only rendezvous that are members of the peergroup will see peergroup specific search requests. Any peer can potentially become a rendezvous peer. Secure peergroups may restrict which peers can act as rendezvous. Rendezvous maintain an index of advertisements published by edge peers via the *Shared-Resource Distributed Index (SRDI)* service. Edge peers use SRDI to push advertisement indices to their rendezvous when new advertisements are published. The rendezvous-edge peer hierarchy allows resolver queries to be propagated between rendezvous only, significantly reducing the amount of peers that needs to be searched when looking for an advertisement.

3.1 Rendezvous Network

The Project JXTA rendezvous service assumes that rendezvous organize into a loosely-coupled unstructured network. This is to reflect the high churn rate predicted in an ad hoc P2P network. In such fluctuating network, it is difficult to maintain a consistent view of all rendezvous in a peergroup without assuming a small peergroup or some tightly-coupled membership structure. This becomes even more difficult as the size of the peergroup and the number of rendezvous increases [4]. While enterprise P2P networks may show more stability, consumer and small devices (pda, phones) P2P networks are likely to be more unpredictable as peers will have a high churn rate. Our assumptions differ in this way with DHT approaches such as (Brocade[8], CHORD[7] or CAN[6]) that assume a relatively stable peer infrastructure where a distributed consistent view can be maintained with minimum overhead. Our assumptions are closer to the random walker approach for unstructured network proposed in [9]. One should also not forget the economic factor of assuming a stable infrastructure network. When deploying a P2P network not all entities may be willing to pay for the extra costs of deploying infrastructure peers (superpeers) to support their network. This economical factor and the need to enable P2P networks that are self-supporting made us thrive toward a more ad hoc and unstructured solution.

4. A Loosely-Consistent DHT

In a highly fluctuating and unpredictable environment, the cost of maintaining a consistent distributed index is likely to outweigth the advantages of having one, as we may spend most of our time updating indices (i.e index trashing). One can separate the cost of a DHT solution into index maintenance, and data access. A DHT approach provides the most efficient mechanism to access data (potentially a single hop access[1]). However, there is a maintenance cost associated with it which typically grows exponentially as peer churn rate increases. On the other hand, not having a DHT means that an expensive exhaustive traversal needs to be performed leading to network flooding. While network crawling is expensive, it does not have any maintenance cost.

Project JXTA proposes a hybrid approach that combines the use of a *loosely-consistent DHT* with a limited-range rendezvous walker. Rendezvous peers are not required to maintain a consistent view of the distributed hash index leading to the term *loosely-consistent* DHT. If the rendezvous churn rate happens to be very low so the RPV remains in sync, the loosely-consistent DHT will be synchronized and achieve optimum DHT performance.

4.1 Rendezvous Peer View (RPV)

Each rendezvous maintains its own *Rendezvous Peer View (*ordered list of known rendezvous in the peergroup by their peer IDs). No strong consistency mechanism is used to enforce the consistency of the RPV across all rendezvous. Rendezvous may have temporarily or permanently an inconsistent RPV, or may not know about all other rendezvous. A loosely-coupled algorithm is used for converging local RPV. Rendezvous periodically select a given random number of rendezvous from their local RPV, and send them a random list of their known rendezvous. Rendezvous purge non-responding rendezvous from their RPV. In addition, rendezvous may retrieve rendezvous info from a predefined set of bootstrapping *seeding* rendezvous. Each peergroup has the ability to define its own set of seeding rendezvous. Any peer can act as a seeding rendezvous. Seeding rendezvous are useful as they permit to accelerate the RPV convergence, as all rendezvous should know about all seeding rendezvous of a peergroup. It is important to point out that seeding rendezvous are only used as the last resort when a rendezvous cannot find any other rendezvous, or to bootstrap the initial booting of a rendezvous. This is to limit dependencies on seeding rendezvous. Seeding rendezvous will also be involved via the previously mentioned random selected rendezvous exchange. Beside the initial seeding, and after a rendezvous has learned about other rendezvous, seeding rendezvous are treated as any other rendezvous. If the rendezvous network is stable, RPVs quickly converge creating a consistent rendezvous view across all rendezvous.

Partitioning of the RPV may occur. However, RPV partitions will start to merge as soon as each partition reaches one of the seeding rendezvous, or a rendezvous in both partitions seeds the intersection. This mechanism is robust enough to enable partition merging even in the case of failures of seeding peers. However, we may have cases where an advertisement will not be found due to temporarily inconsistent RPVs. This inconsistency will be resolved as rendezvous continue to randomly exchange information.

Figure 2.1 shows how a newly published advertisement is indexed on the rendezvous network. Peer P1 publishes a new advertisement on its rendezvous R2 via the SDRI service. Each advertisement is indexed by SRDI using a predefined number of keys such as the advertisement name, or the advertisement ID (PeerID). It is important to point out that only indices of advertisements are pushed to a rendezvous by SRDI. This is to minimize the amount of data that

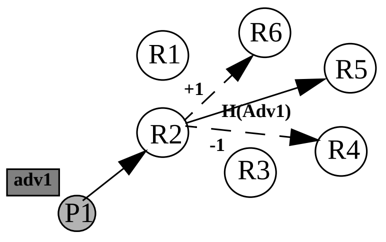
1. It is important to point out that the virtual route to reach the logical hop may in fact involve multiple physical hops due to the underlying network topology.

need to be maintained on a rendezvous. R2 uses the DHT function (H(adv1)) to map the index to a rendezvous in its local RPV map. The RPV on R2 contains the R1 to R6 rendezvous. Let's suppose that the DHT function returns R5. R2 will push the index to R5. To increase the probability to retrieve the index in the vicinity of R5 and address the potential disappearance of R5, the index is also replicated to the RPV neighbors of R5 (+1 and -1 in the RPV ordered list). In our example, the index is also replicated on R4 and R6. It is important to note that index proximity in the RPV does not necessarily means physical network proximity. R5 and R4 may be on opposite sides of the hemisphere. However, replicating the index around R5 means that we are creating a logical region in the RPV where a specific index can be located.

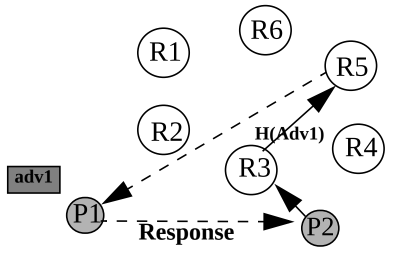
Now, let's assume that an edge peer P2 is looking for advertisement adv1 (see Figure 2.2.a). P2 will issue a resolver query to its rendezvous R3. The SRDI service on R3 will compute the DHT function (H(adv1)) using R3 local RPV. If the RPV on R2 and R3 are the same, the DHT function will return the same rendezvous R5. R3 can forward the request to R5 that will forward it to P1 for responding to P2. This works as long as the RPV is the same on R2 and R3.

Suppose that R5 is down, and R3 updated its RPV to reflect the fact that R5 disappeared (see Figure 2.2.b). R3 will find out that R5 is down either through a RPV map update or when it tries to send a message to R5. In this scenario, R3 has a new RPV that contains rendezvous R1 to R5. R5 now points to the rendezvous R6 of our previous RPV map. One missing rendezvous means that the RPV shifted by one position.

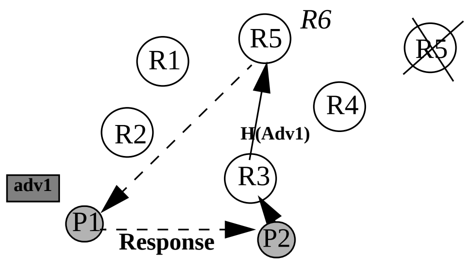
Upon receiving the resolver request from P2, R3 will compute the DHT function that will return the new R5 rendezvous. Since we also published the index on R6 as part of our replicated index strategy, we will find the index on the new R5. This example demonstrates, how even with an inconsistent RPV we able to successfully use the DHT. As long as the RPV shift is within the DHT replicated distance (+1,-1), we can guarantee the index will be found. Replicating the index in the vicinity of the initial rendezvous target increased our ability to retrieve the index even with a shifted RPV. It is important to point that no distributed maintenance of the RPV is required here. All RPV maintenance is done locally by having rendezvous exchanging part of their view with a random set of known rendezvous. The replication distance can be increased (+2 or +3) for large RPV maps.



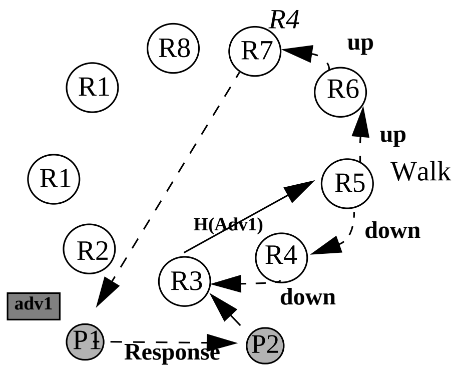
1) Publish Adv1



2.a) Search Adv1 (DHT consistent view)



2.b) Search Adv1 (DHT inconsistent view)



2.c) Search Adv1 (Limited-Range Walker)

Figure 2. Loosely-consistent DHT

Let's look at a more chaotic scenario where the RPV is going through massive changes (see Figure 2.2.c). The RPV on R3 is now composed of 8 rendezvous (R1-R8). R7 corresponds to the original R4. When R3 receives the query request from P2 it will compute the DHT function that maps the index to R5. Since the RPV has drastically changed, the index will not be found on R5. In this case, an alternative mechanism is used to *walk* the rendezvous peer view to continue searching. A default *limited-range walker* is used to walk the rendezvous from the initial DHT target rendezvous. The walker will proceed in both *up* and *down* directions (see Figure 2.2.c). The intent of the limited range walker is to look in the vicinity of the initial target rendezvous for a rendezvous that may have the index. The limited-range walker takes advantage of the increase probability to find the index in that region of the RPV due to the DHT neighbor replication scheme. In our example, R5 forwards the request to both R4 (down) and R6 (up). A hop count is used to specify the maximum number of times the request can be forwarded. If R4 does not have the index, it will forward the request to R3 (down). R6 forwards the request to R7 (up). When the index is found on R7, the query is forwarded to P1 and the walk stops in the up direction. The walk on the down direction will continue until the hop count is reached, or there are no more rendezvous in that direction. At that point, a continue walk request will be sent to P2, to ask P2 if the walk should continue. The response to the initial P2 request may have reached P2, before the continue request is sent. P2 can then decide to either continue or stop the walk.

Walking the RPV in both directions reduces query latency. It is also important to point out that since the RPV is ordered by peer IDs, each rendezvous is only visited once even if RPVs are inconsistent. The limited-range walker provides a fall-back mechanism to the DHT lookup. The combination of both makes the use of a DHT more practical for ad hoc unstructured P2P networks. If the rendezvous infrastructure is stable then we will take full advantages of the DHT, and rarely have to use the more expensive limited-range walker.

5. Conclusions

The open-source Project JXTA defines a generic P2P virtual network overlay usable to implement a wide variety of P2P applications and services. Project JXTA defines core building blocks while allowing developers to customize and plug in their own policies. This paper describes a loosely-consistent DHT that combines a DHT approach with a limited range walker to search for advertisements in the JXTA rendezvous network. The loosely-consistent DHT is used as the default pluggable resolver policy of the JXTA platform. This hybrid DHT approach has the advantages of not requiring a strong-consistency DHT maintenance, and is well adapted to ad hoc unstructured P2P networks.

An implemention of the loosely-consistent DHT has been completed and will be released as part of the upcoming scalability release of the JXTA Platform (platform.jxta.org).

6. Acknowledgments

We would like to thank Bill Joy, Mike Clary, Ingrid Van Den Hoogen, Bill Yeager, Juan Soto, Matt Reid, Mike Duigou, J-C. Hugly, Carl Haywood, Ahkil Ahora and Kuldeep Pabla from Sun, and the entire Project JXTA community for helping us refine the Project JXTA protocols.

7. References

[1] Project JXTA, *www.jxta.org*.

[2] B. Traversat and al., *The Project JXTA Virtual Network*, *www.jxta.org/docs/JXTAprotocols.pdf.*

[3] S. Oaks, B. Traversat, L. Gong, *JXTA in a Nutshell*, O'Reilly Press, 0-596-00236-X, Sept. 2002.

[4] B. Traversat and al., *Project JXTA-C: Enabling a Web of Things*, to be published in the proceedings of the HICSS-36 Conference, Jan. 2003.

[5] LongWork Network, http://www.echelon.com/products/Core/protocol/Default.html.

[6] S. Ratnasamy and al., *A Scalable Content Addressable Network*, ACM SIGCOM, 2001.

[7] F. Dabek and al., *Building Peer-to-Peer Systems with Chord, a Distributed Lookup Service*, 2001.

[8] B. Zhao and al., *Brocade: Landmark Routing on Overlay Networks*, in Proceedings of the First International Workshop on Peer-to-Peer Systems (IPTPS 2002), Mar 2002.

[9] Q. Lv, and al., *Search and Replication in Unstructured Peer-to-Peer Networks*, www.cs.princeton.edu/~qlv/download searchp2p_full.pdf