Design for a phase 0 network

Gemini Preprint #86

Kim Gillies
Gemini Observatory, Tucson Project Office,
950 North Cherry, Tucson, AZ 85719

Shane Walker
Gemini Observatory, Southern Operations Center,
c/o AURA, Casilla 603, La Serena, Chile

Design for a phase 0 network

Kim Gillies*[a], Shane Walker[b]

[a]Gemini Observatory, Tucson Project Office, 950 North Cherry, Tucson, AZ 85719
[b]Gemini Observatory, Southern Operations Center, c/o AURA, Casilla 603, La Serena, Chile

ABSTRACT

Today's astronomers may use the telescopes and instruments of many observatories to execute their science observations. Discovering the distributed resources that are available is time consuming and error prone because astronomers must manually take facility information and match it to the needs of their science observations. While Phase 1 and Phase 2 of the proposal process are well supported by a wide variety of software tools, the initial phase of discovering what resources are available, Phase 0, suffers from a lack of software support. This paper describes and proposes the creation of a Phase 0 Network to fill this void. The network is built upon peer-to-peer (P2P) technology, showing that this new approach to distributed computing has viable uses in astronomy.

Keywords: Phase 0, peer-to-peer, P2P, proposals, distributed computing

1. INTRODUCTION

Astronomers write proposals for observing time on telescopes during Phase1 and complete their proposals during Phase 2. Software tools such as Gemini's Phase 1 Tool[1] and Observing Tool[2] provide the astronomer with assistance during these steps. However, determining if a science experiment can be executed on the set of instruments and telescopes spread around the world can be a time consuming, error-prone process because it must be done manually, by inspecting manuals and web sites. Making the correct choices is challenging given the sophistication of today's instruments and facilities.

Phase 0, the first step in the proposal process, is the process of discovering what facilities and instruments, if any, are available that would allow a proposed science observation to be accomplished. There may be several facilities with instruments that could allow a particular observation to be executed; some better than others. It might be difficult for one astronomer to be knowledgeable about all the facilities available in the world. It is even more difficult to decide if those facilities can enable a science observation.

Phase 0 software tools assist the astronomer with the task of finding out whether or not an observation can be executed at a facility. The Phase 0 Network, proposed in this paper, is a distributed software system that provides this discovery service. The design demonstrates how a peer-to-peer (P2P) infrastructure can be used to allow loosely coupled observatories to provide the service with very little contact.

An astronomer's science problem is described in an Extensible Markup Language (XML) document and submitted to the phase 0 P2P network. Participating phase 0 peers evaluate the science description as a query and return a result stating how well they can match the proposed observation along with a description of the facility and instrument configuration that can be included in a Phase 1 proposal. This description of resources is also an XML file.

This paper describes the Phase 0 Network (P0N) design and presents a reference implementation written in Java utilizing a freely available P2P system. The XML representation of science observations is presented along with the interfaces that allow a new site to join the Phase 0 Network. The results of tests of the demonstration implementation are shown.

2. P2P DISTRIBUTED COMPUTING

P2P is dubiously known through the public attention given to the Napster file sharing application and its successors Gnutella, Freenet, and AudioGalaxy. Because of this association, P2P has some credibility issues and may not be viewed as a viable, useful approach to distributed computing. In the science area, the SETI@home[3] and Folding@Home[4] projects have shown that P2P distributed resource sharing can be a very useful way of solving some distributed computing prob-

* contact author K.Gillies at kgillies@gemini.edu. contact author S.Walker at swalker@gemini.edu.

lems. In both of these projects, the excess compute cycles of the computers of volunteers work on parts of a science problem.

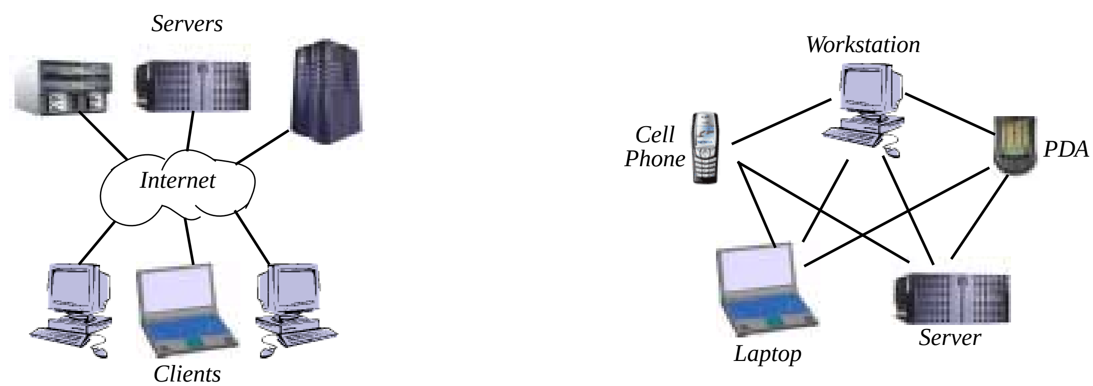
A peer is a single participant in a P2P system. It can be any connected device. Ideally, any peer can communicate directly with any other peer bypassing centralized web application servers or web servers, but there are a wide variety of architectures that can be called P2P. The thing that makes an application P2P is that the peers are active participants in the application, not passive clients. A peer provides a service for the benefit of other peers in the group. For instance, SETI@home peers rely upon a central server but are active participants in the application.

The achilles heel of Napster was the central server that facilitated the exchange of files between peers. Newer file sharing applications are completely decentralized. This is one of the significant ways P2P computing is different from the more traditional client-server computing. The goal of P2P is to move away from traditional hierarchical arrangements of computing resources and take advantage of the vast computing resources on the “edge” of the network. P2P computing provides high availability and reliability by duplicating services across multiple peers in the group.

2.1. P2P versus web services

The idea of web services is currently very popular as an architecture for providing web-based functionality. Web services is a phrase given to the combination of XML, Simple Object Access Protocol (SOAP), Universal Description Discovery and Integration (UDDI), and Web Services Description Language (WSDL). SOAP is a simple XML messaging protocol for exchanging information. UDDI is a building block protocol that allows potential clients to locate and learn how to interact with web-based services. WSDL defines the XML grammar used to specify a service as communication endpoints.

Potential clients need to locate WSDLs in order to use the web services. Fundamental to the web services architecture is the idea of a public UDDI Business Registry where service providers register their WSDL information and potential clients can go to find information need to contact and use services. IBM, Microsoft, and Hewlett Packard are currently providing public UDDI registries at no charge.



Web-based or client server system　　　*P2P computing with direct communication*

Figure 1. Web-based/client server systems versus P2P direct communication.

The web services architecture is just an extension of current client-server solutions. Web services are centralized and hosted by traditional servers. Clients locate services which they use passively. Although there are infrastructure peers in most P2P system, the peers themselves provide the services as equals. The intelligence for locating services is distributed throughout the network of peers. Figure 1 shows the architectural difference between web services and P2P pointing out the different communication pattern in the two.

In many ways P2P type applications are ideal for the loosely organized astronomy community. P2P applications do not require the up front agreement of all participants in order to go forward. Observatories or groups who are interested can participate by joining the P2P application as a peer and will be discovered by the others with no work or active cooperation. Individuals or organizations who wish to provide information can join without getting approval or registering. For instance, in many ways P2P is an ideal architecture for sharing personal astronomy data or large archives of data as discussed by various virtual observatory groups[5,6].

3. THE PROJECT JXTA INFRASTRUCTURE

The P0N described in this paper is a P2P application built upon Project JXTA Technology[7]. Project JXTA is a network computing infrastructure designed for P2P computing. It is an open project, meaning that the protocols and source code are available to all without cost and restrictive licensing. JXTA started its life as a research project at Sun Microsystems under the guidance of Bill Joy and Mike Clary. The goal was to produce a system that included the basic building blocks and services that most P2P applications need and to run on any connected device be it a large server, PC, or cell phone. So by design JXTA is not tailored to any specific application such as file sharing or searching.

JXTA is not a library for Java or C programmers; rather, it is set of protocols that define a thin network layer and mechanisms for needed functionality. The protocols allow applications to be platform independent. Applications can be written in different languages and will still interoperate. Currently JXTA applications can be programmed in Java, Perl, Python, Ruby, and C[8].

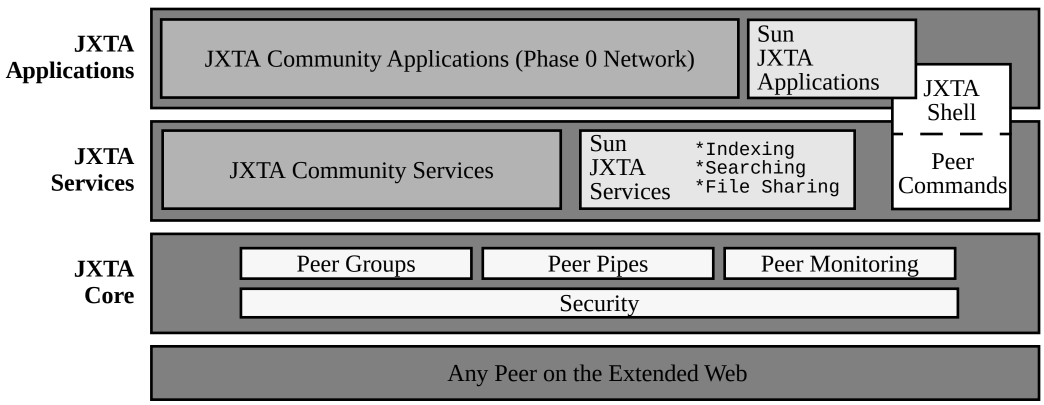


Figure 2. JXTA Software Layering[9].

Figure 2 shows the layering of the JXTA software infrastructure. At the lowest level capabilities exist to create and delete peer groups, to advertise the groups to potential members, to enable peers to find the groups, and to join or leave groups. At the next layer the core functionality is used to create a set of peer services, including indexing, searching, and file sharing. Peer applications are built upon the core functionality and built-in services. The Phase 0 Network exists at the JXTA Applications layer. The next sections briefly describe a few needed P2P and JXTA concepts.

3.1. JXTA Technology Concepts

This section introduces the core concepts found in the JXTA platform. Since JXTA provides the building blocks for P2P applications and not the applications themselves, the number of concepts is kept intentionally small. This facilitates their implementation in the various language bindings and gives more freedom to the developer building services and applications using JXTA.

Peers are the central concept of the JXTA infrastructure. A peer is an entity that provides or uses a service, exchanges information, and generally interacts with other peers. A peer is not tied to a specific IP address or machine, but rather associated with a unique peer ID that stays the same no matter where the peer is located. Furthermore, no requirements are placed on the low level network technology that two peers use to exchange messages. Instead, the communication medium is abstracted into the concept of a pipe. Pipes are unidirectional communication channels that can be bound to one or more interested peers. A pipe can be implemented on top of any transport using any number of protocols.

Taken together, peers and the pipes that connect them form a “virtual network”, a network that is independent of the physical hardware and network technologies on top of which it is running at a given time. A given JXTA implementation is responsible for mapping the virtual network onto the collection of physical networks that connect the peers. For example, network realities such as Network Address Translation and firewalls are transparent to the programmer writing a JXTA application. It is the responsibility of the implementation to utilize proxy services, etc. for traversing firewalls.

Peers exist to provide services to other peers and/or use the services of other peers. Without a scoping and organizing mechanism of some sort, the collection of all the peers in the world would become unwieldy. The JXTA peer group supplies the organizing concept. A peer group is a collection of peers that use the same services and share the same types of information. Membership in a group can be guarded, if desired, to form secure communities of interacting peers.

All of the concepts mentioned here—peers, pipes, peer groups, and services—are represented in the JXTA framework by advertisements. An advertisement is an XML description of an item. A pipe advertisement, for example, describes a communication channel. Using the advertisement, a peer can create a pipe object in the particular language binding in which the peer is implemented. A peer publishes its advertisements so that other interested peers can find them, and conversely can ask other peers to send advertisements of interest.

3.2. JXTA Protocols

In addition to the basic concepts mentioned above, a collection of agreed upon communication protocols is needed to form a working system. In JXTA, a protocol is simply one or more messages in a specified format. Recall that the actual delivery mechanism of the messages is not bound to any particular network transport.

There are six JXTA protocols:

- Peer Discovery Protocol - used to locate advertisements for peers, peer groups, pipes, and services
- Peer Resolver Protocol - provides a query and response mechanism for searching peers
- Peer Information Protocol - allows a peer to fetch status information from another peer
- Pipe Binding Protocol - used to establish a communication channel with another peer.
- Endpoint Routing Protocol - provides a means by which a peer can locate the route to another peer for the purpose of sending a message; attempts to provide resiliency in the face of partial network failures
- Rendezvous Protocol -used to propagate messages within a peer group by means of “rendezvous” peers

The last protocol is worthy of a bit more discussion. A rendezvous peer is a peer like any other, but performs some additional duties on the part of the “normal” peers it supports. Like any peer, a rendezvous peer caches the advertisements it knows about. In addition, it forwards discovery requests it receives to other known rendezvous peers and to the collection of normal peers it supports.

4. DESCRIBING SCIENCE OBJECTIVES IN XML

A basic assumption of the Phase 0 Network is that an observation’s science objectives can be described abstractly such that they are independent of a specific instrument or telescope specification. Of course this is possible, astronomers do this whenever they decide whether or not an observation is possible on some telescope and instrument. But the process for making this description is not trivial. It requires the astronomer to have a good understanding of the science and the ability to relate the goals of the science to technical requirements. Most Phase 1 proposals include a technical justification section that allows the astronomer to explain in words how an experiment can be done with a specific telescope/instrument setup. The success or failure of a proposal often rests on the quality of this section.

From the point of view of the Phase 0 Network, the level of detail in the description must be adequate to represent the aspects of the science that are pertinent to the job of matching science goals to facility configurations. Some parts of the description are factual and either the facility can meet them or it can’t. An example would be a need for a near-IR imager with a broadband J filter. Matching other aspects of the query can require extensive research on the part of the facility staff or even an Integration Time Calculator. Returning the required integration time for an observation is something that falls into this category.

This paper introduces an XML-based description of a science task. To arrive at the XML description of a science task, a small group of Gemini astronomers were interviewed and asked to discuss the process they use when considering the use of an instrument and telescope. Then some basic science observations were considered to devise the approach described here. The specification given here is not complete and only covers a subset of observation types and modes that astronomers use. What we've found is that creating an XML description of science objectives is challenging and more work is needed in this area. Our hope is that by publishing this early effort, interested collaborators will join the effort to refine and expand the science description and the P0N.

A P0N client peer will request information from the user that will be used to construct an Extensible Markup Language (XML) document, which will then be submitted to the P0N network. Each P0N facility peer has the opportunity to formulate a response in the form of an XML document and return it to the client. The following discussion assumes a basic understanding of XML terminology[10].

4.1. Design of the phase 0 query and response

The element `phase0Query` is the root of the XML document that is sent to a P0N service peer. It can contain up to three elements. The `wavelength` element is used to specify the wavelength regime of the observation. It is a required element since nothing much can be determined without knowing the wavelength of the observation. The optional `targets` element is present to specify where in the sky the targets are located. The final element describes the type of observation. The possibilities currently are `imaging` and `spectroscopy`.

The optional elements that can be included in each of the top level elements are considered constraints. The guideline of the design is to allow the astronomer to constrain the information in the element as much or as little as is appropriate. Elements exist to provide a full range of detail. It is more work for the user to specify tighter constraints, but the result is generally a response that is a more useful description of the required configuration. For instance, within the `targets` element the `north` and or `south` constraints can be included to loosely state the location of the targets in the sky. To be more specific, a target can be indicated with a specific right ascension and declination or even a range of positions. Generally, it's also true that a more constrained description is harder for the PON facility to match.

The contents of the `phase0Response` is more complex. The first element, `result`, indicates whether or not the telescope and facility can match the query. This is indicated with success or failure and a score; ranging from 0 to 100 for a perfect match allowing clients to rank responses. Minimally, the response will include a `resourceSet` element that describes the configuration of the telescope and instrument that was used to satisfy the query. This information is provided for inclusion in a Phase 1 program.

The `phase0Result` contains detail that complements the depth of information provided in the `phase0Query`. When the query specifies the observation in sufficient detail, the response contains parameters such as signal to noise and expected integration time. The observing conditions under which the observation must be made can be included along with the frequency those conditions occur.

4.2. Phase 0 query and result examples

The specifics of how the `phase0Query` and `phase0Result` are structured are specified in a schema described elsewhere[11]. This section explains more about the Phase 0 query and response by showing some examples. The examples are based upon the instruments available on the Gemini North Telescope. Information on Gemini North instruments comes from the Gemini web site[12].

The simplest query and response is shown in Figure 3. Lines 2 and 3 (and 8 and 9) show the required XML declaration and the phase 0 namespace declaration. The only required top level element is `wavelength` which is specified with the loose constraint `optical`. The result is shown in lines 8-14. Line 10 indicates the query was matched with 100% certainty. The returned `resourceSet` shows the Gemini North Telescope and its one optical instrument. This type of query can be used to browse a peer facility's instrument ensemble.

Figure 4 demonstrates a query that results in a failed response. In this case, the Gemini North service has been asked to list all optical instruments, but the query also includes the `south` element within the `targets` section. Without more specific information, the Gemini North service must indicate failure as is shown in line 13. The reason for failure can be provided as in line 14 to help the user improve the query.

```
<!-- Simplest query will display available instruments. --->
<?xml version="1.0" encoding="UTF-8">
<phase0Query xmlns="http://ftp.gemini.edu/Support/phase0/2002">
  <wavelength>
    <optical/>
  </wavelength>
</phase0Query>

<!-- Result returns the optical instruments of Gemini North -->
<phase0Response xmlns="http://ftp.gemini.edu/Support/phase0/2002">
  <result match="success" score="100"/>
  <resourceSet category="telescope">Gemini North
    <resource category="instrument">GMOS-North</resource>
  </resourceSet>
<phase0Response>
```

Figure 3. A simple query to show available instruments.

```
<!-- This query requests all optical instruments in the south. -->
<phase0Query xmlns="http://ftp.gemini.edu/Support/phase0/2002">
  <targets>
    <south>
  </targets>
  <wavelength>
    <optical/>
  </wavelength>
</phase0Query>

<!-- This time the response is empty demonstrating a failed query. -->
<phase0Response xmlns="http://ftp.gemini.edu/Support/phase0/2002">
  <result match="failure" score="0">
    <reason>The targets are out of range for this telescope, please
                refine your query.</reason>
  </result>
<phase0Response>
```

Figure 4. A query that results in a failure.

The query shown in Figure 5 fixes this problem by further constraining the target declinations using the `range` element within the `targets`. The `range` element is used throughout the `phase0Query` and `phase0Result` in a number of elements to indicate a range of values for some parameter. The `type` of the specified value and its `units` are added as attributes in the `range` element. There can be more than one `range` element to show a set of disjoint ranges. Similar to range, one or more `specific` elements can be used to constrain a parameter to one or more specific cases. It uses a `type` and `units` attribute like `range.`

```
<!-- Query further qualifies and constrains the location of targets. -->
<phase0Query xmlns="http://ftp.gemini.edu/Support/phase0/2002">
  <targets>
    <south>
    <north>
    <range type="dec" units="degrees">
      <lower>-45.0</lower><upper>50.0</upper>
    </range>
  </targets>
  <wavelength>
    <optical/>
  </wavelength>
</phase0Query>
```

Figure 5. Constraining the declination of targets.

Figure 6 shows a more involved case. The astronomer is interested in attempting to detect a point source using Near-IR imaging. The `wavelength` element shows that the `nearir` element is constrained with a `filter` element set to broad-band J. The `imaging` element follows and now includes several constraints. Each constraint should be viewed as the *worst* value that is acceptable for the parameter. The service must meet or exceed the constraint to completely match the query. The `fov` element indicates the need for a field of view of at least 30 arcseconds. The `sn` element must be at least 4. The observations should be made at an airmass of 2 or less. The type of object can be `extendedObject` or `pointSource`. The contained `brightness` element indicates the observer's best guess at the object's magnitude in J.

Finally, the type of experimental result is declared with the `detection` element. The XML description supports three types of experiments: `detection`, `resolve`, and `photometry`. A `detection` experiment attempts to determine whether or not a target is present. A `resolve` experiment attempts to resolve two or more targets. A `photometry` experiment attempts to measure the brightness of a source. Photometry is typically a more demanding experiment than detection. Each of the experiments can contain elements that constrain it. This detection is required to be sampled in at least 3 pixels.

```
1.  <!-- This examplee adds detail on the needed science goals. -->
2.  <phase0Query xmlns="http://ftp.gemini.edu/Support/phase0/2002">
3.    <targets>
4.      <north>
5.    </targets>
6.    <wavelength>
7.      <nearir/>
8.      <filter type="broadBand">J</filter>
9.    </wavelength>
10.   <imaging>
11.     <fov units="arcsec">30</fov>
12.     <sn>4</sn>
13.     <airmass>2</airmass>
14.     <pointSource>
15.       <brightness>
16.         <specific units="mag">22.2</specific>
17.       </brightness>
18.     </pointSource>
19.     <detection>
20.       <inPixels>3</inPixels>
21.     </detection>
22.   </imaging>
23. </phase0Query>
24.
25. <!-- This time the response returns NIRI along with more specific exposure information-->
26. <phase0Response xmlns="http://ftp.gemini.edu/Support/phase0/2002">
27.   <result match="success" score="100"/>
28.   <obsConditions imageQuality="80"
29.           skyBackground="any" cloudCover="70" waterVapor="Any" frequency="45"/>
30.   <imaging>
31.     <pixelScale units="arcsec/pixel">.117</pixelScale>
32.     <fov units="arcsec">120</fov>
33.     <sn>5</sn>
34.     <airmass>1.5</airmass>
35.     <integrationTime units="seconds">3120.0</integrationTime>
36.   </imaging>
37.   <detection>
38.     <inPixels>9.4</inPixels>
39.   </detection>
40.   <resourceSet category="telescope">Gemini North
41.     <resourceSet category="instrument">NIRI</resource>
42.       <resource category="filter">J</resource>
43.       <resource category="camera">f/6</resource>
44.     </resourceSet>
45.   </resourceSet>
46. <phase0Response>
```

Figure 6. An experiment to detect a point source in the J filter.

The successful response to this query starts in line 26 of Figure 6. The structure of the response looks a lot like the query with a few extras. The facility configuration in the `resourceSet` specifies Gemini North, the Near-Infrared Imager (NIRI), the f/6 camera, and the J filter.

Very often the feasibility of an observation is highly dependent on the conditions of the site during the observation. In line 28, the `obsConditions` element describes the minimal conditions required to meet the science described along with the frequency the conditions (or better) occur. The conditions are represented by four properties and percentile values representing the frequency of occurrence of the properties. The four properties are: image quality, sky background, cloud cover, and water vapor. For instance, the image quality can be 20%, 50%, 80% or any. The best image quality is indicated with the 20% value. Some properties are determined through statistical analysis of measurements and some are based upon models. This approach to specifying the observing conditions is used at Gemini and is described on the Gemini web site[13]. It is a useful approach for phase 0 because it creates an abstraction of the conditions at any one site.

The percentiles must be tied to actual values which are unique for each site. For this experiment, at this site, the response suggests image quality is not too important since we are doing a detection, sky background is fairly unimportant at J, we aren't doing photometry so cloud cover can be patchy at 70%, and water vapor is not too important for a detection. Given these conditions it is stated that the observation has a 45% probability of being executed. Tighter observing condition constraints will reduce the likelihood of the conditions occurring. The choice between integration time and likelihood of execution is difficult to balance. A detection experiment can be carried out in non-ideal conditions.

Finally, given the loose observing conditions, the response indicates that the observation will require at most 3120 seconds with a signal to noise of at least 5 when observed at an airmass of 1.5 or less. The field of view of the detector is present since it was a constraint in the query. The `pixelScale` element in included to indicate the size of the target on the detector. The `inPixels` value of 9.4 surpasses the required value of 3.

5. THE JXTA-BASED PHASE 0 NETWORK

This section describes the design of the JXTA-based P0N reference implementation. There are many advantages with choosing to use an open, standards-based product as the foundation of a system. With the right choice, the greatest advantage is that the hard problems will be solved, most of the code will be written, and the programmer can focus on what makes his application unique. Fortunately it is true in this case. JXTA provides the framework that allows us to focus on the contents of the message and the implementation of the peers. Additionally, we chose Java as the implementation language, but nothing in the design requires Java. Any of the languages with JXTA libraries would do.

JXTA provides the peer concept and the idea of a virtual network. Each peer has a unique peer ID in the form of a peer advertisement that is posted to the JXTA network. JXTA peers cache advertisements. The abstraction of the virtual network and peer ID allows the peers to freely move and change IP addresses. This feature is not tremendously useful in P0N, but does allow a freedom to observatories they might otherwise not have. In the JXTA P0N, there are two types of peers: client peers and facility peers. Any peer can act in one or both roles.

JXTA supports secure peer groups. The P0N uses a peer group, called the Phase 0 Group, to facilitate discovery of facility peers and to provide security and isolation from the rest of the JXTA network. At the moment, membership in the peer group is by simple password, but all communications are encrypted ensuring the privacy of a user’s science query. It is assumed that facility peers are long-lived and will have an almost constant presence in the Phase 0 Group. Some facilities may also choose to provide public client peers that could be accessed through non-JXTA means. An example would be a CGI script behind a web server.

A JXTA peer in a group can provide network services to all the members of the group. Each P0N facility peer can provide and publish advertisements for one or more services, called facility/instrument services, that can accept and respond to the phase 0 query. An observatory programmer may choose to provide a single service instance that handles all instruments for the facility or he may provide one service for each facility/instrument pair. The peer providing the service creates advertisements and posts them to the JXTA network. A facility peer’s service advertisements are propagated throughout the Phase 0 Group with no maintenance required by anyone. By “publishing” a service advertisement, all the peers in the group find out about it and can use it.

Finally, the service advertisements include instructions that tell other peers how to communicate with the facility/instrument service. The JXTA abstraction for communication is the pipe and P0N uses a bidirectional pipe between the P0N client and the P0N facility/instrument service. JXTA provides mechanisms to deal with the problems of today’s Internet. Peers can be behind a firewall or can have dynamic NAT-based IP addresses. In this case, the pipe uses an HTTP-based transport and peers behind a firewall go to a public rendezvous peer acting as a post office to find their incoming queries.

To recap, the entire JXTA-based P0N is shown in Figure 7. The physical network is at the bottom of the figure where there are a three observatories. The peers within the grey observatory ovals are P0N facility peers. Two observatories provide TCP/IP access through their firewalls, and one uses the HTTP transport. In the bottom middle area are public rendezvous and relay peers. These peers are present on public, well-connected hosts. A User Workstation, representing an astronomer who wishes to use the P0N is also shown.

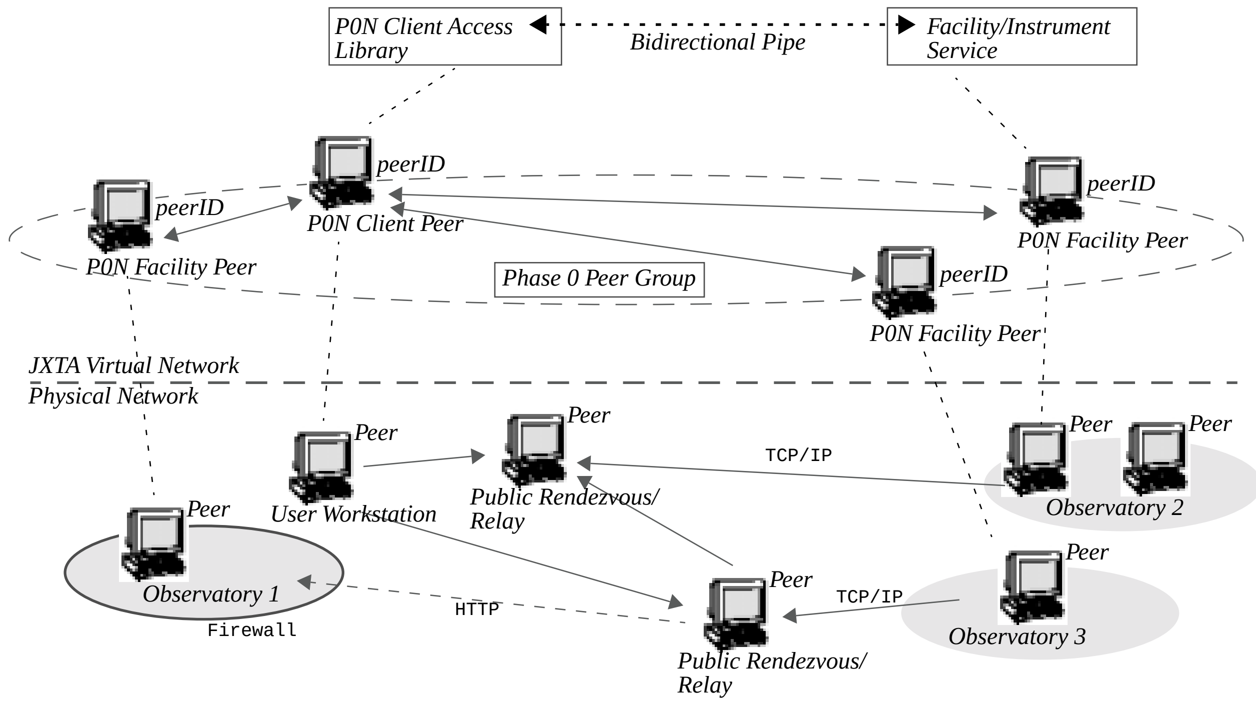


Figure 7. The JXTA-based phase 0 Network Design

The dashed line shows the break between the physical network of IP addresses and firewalls and the JXTA virtual network of peer IDs, peer groups, services, and pipes. The peers inside the dashed oval are members of the Phase 0 Group. The P0N client peer and three P0N facility peers are shown. Note that in the physical layer, communication between peers involves rendezvous and relay peers but in the JXTA virtual network, communication is peer to peer.

Some user interface, be it a GUI-based application or a browser, gathers the required information and builds a phase 0 query document for the user on the P0N client peer. The document is submitted to all facility/instrument services, which evaluate the query and respond directly to the requesting peer. The client peer returns the response to the user.

6. TESTS AND RESULTS

The Phase 0 Network is a research project in progress. The authors have implemented a Java-based library to facilitate the creation of P0N clients, facility peers, and facility/instrument services and to test JXTA. We have not yet implemented a full up facility/instrument service. Work continues on the definition of the XML description of science objectives.

In the future, we plan to investigate allowing any Java-based peer to host any observatory's facility/instrument services. Java allows code to be loaded from the network and run securely, and P2P provides better performance when more than one peer provides a specific service. Again, JXTA allows this with little new work on the part of the programmer.

7. CONCLUSIONS

This paper provides an introduction to the concept and design of a Phase 0 Network as a P2P application. The work is ongoing and the authors look forward to extending the work and to future collaboration with the community.

JXTA has a fairly steep learning curve because the concepts are quite different than traditional distributed computing. However, we have found the JXTA Java library to work as advertised and it seems to be stable enough to provide a reasonable platform for this project. The P2P approach to implementing the P0N is quite powerful and saves programming time.

Phase 0 is one of the last remaining parts of the proposal process that lack adequate software tool support. It is also the phase where tools can provide a great impact on the quality and efficiency of the observing process. Potentially, more astronomers will be impacted by phase 0 tools than at any other phase since more astronomers conceive of experiments than apply or are accepted at Phase 1.

The community of astronomers has much to gain through a Phase 0 Network that is fully supported by the international community of observatories. The P0N provides a platform for astronomers to ask, "Can this observation be done? Today?" For observatories, the P0N provides a way to expose the capabilities of their facilities to a wide, international audience.

The Phase 0 Network provides an opportunity to integrate current support web sites with integration time calculators. The peer services provide a focal point for observatories to locate the core knowledge of their instrument scientists and their best observing practices for the benefit of all.

ACKNOWLEDGEMENTS

The authors wish to thank Dr. Phil Puxley and Dr. Joe Jensen of the Gemini Observatory for discussing the process of planning an observation and the analysis of associated instrument specifications.

We acknowledge the support of the Gemini Observatory, which is operated by the Association of Universities for Research in Astronomy, Inc., on behalf of the international Gemini partnership of Argentina, Australia, Brazil, Canada, Chile, the United Kingdom, and the United States of America.

REFERENCES

1. Gemini Observatory, "The Gemini Phase 1 Proposal Tool," http://www.gemini.edu/sciops/P1help/p1Index.html.
2. Gemini Observatory, "The Gemini Observing Tool," http://www.gemini.edu/sciops/OThelp/otIndex.html.
3. "SETI@Home," http://setiathome.ssl.berkeley.edu.
4. "Folding@home," http://www.stanford.edu/group/pandegroup/Cosm.
5. US National Virtual Observatory Website, http://www.us-vo.org.
6. European Virtual Observatory Website, http://www.eso.org/avo.
7. Project JXTA Home Page, http://www.jxta.org.
8. JXTA Core Components Project, http://core.jxta.org/servlets/ProjectHome.
9. Image based upon one in, *Project JXTA: A Technology Overview*, Li Gong, Sun Microsystems, Inc., http://www.jxta.org.
10. W2C Organization, *Extensible Markup Language (XML) 1.0 (Second Edition)*, http://www.w3.org/TR/2000/REC-xml-20001006.
11. Phase 0 Network Home Page, http://ftp.gemini.edu/Support/phase0/p0Index.html.
12. Gemini Observatory, "Gemini Instrumentation," http://www.gemini.edu/sciops/instruments/instrumentIndex.html.
13. Gemini Observatory, "Observing Condition Constraints," http://www.gemini.edu/sciops/ObsProcess/ObsProcIndex.html.