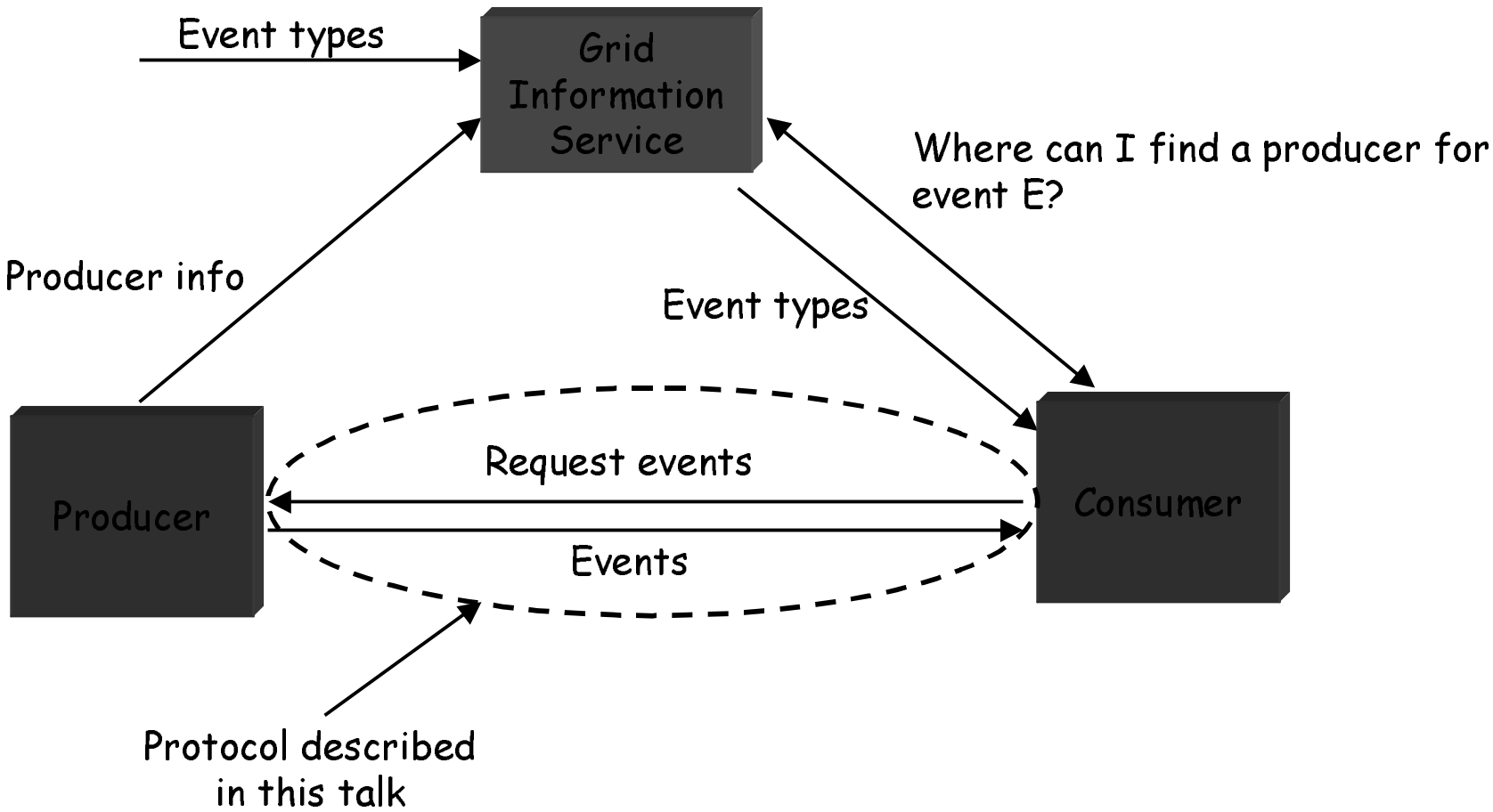
A Simple XML Producer-Consumer Protocol

Warren Smith (NASA Ames)
Dan Gunter (LBNL)
Darcy Quesnel (ANL)

Monitoring Architecture



Overview

- Define a relatively simple producer-consumer protocol
  - TCP
  - XML representation
- Implement the protocol
- Provide to the community
- Gain experience to specify better events, protocols, GIS objects

Events

- Simplification of what we've talked about before
- Consist of:
  - Type
  - A set of elements
    - Name
    - Value (string)
    - No units, no accuracy
  - Always have a timestamp
- We define 2 events

CPU Load Event

- HostName
  - The name of the host the measurement is made on.
- Load1
  - The 1 minute CPU load reported by uptime.
- Load5
  - The 5 minute CPU load reported by uptime.
- Load15
  - The 15 minute CPU load reported by uptime.
- TimeStamp
  - The time the measurement was made.

CPU Load Event in XML

```
<UptimeCPULoad>
    <Load1>1.5</Load1>
    <Load5>1.6</Load5>
    <Load15>1.3</Load15>
    <HostName>foo.nas.nasa.gov</HostName>
    <TimeStamp>2000-11-09T21:51:45Z</TimeStamp>
</UptimeCPULoad>
```

Round Trip Time Event

- SourceHostName
  - The name of the host performing the ping command.
- TargetHostName
  - The name of the host being pinged.
- RoundTripTime
  - The round trip time reported by the ping command in milliseconds.
- TimeStamp
  - The time the measurement was made.

Round Trip Time Event in XML

```
<Ping>
    <SourceHostName>foo.nas.nasa.gov</SourceHostName>
    <TargetHostName>bar.lbl.gov</TargetHostName>
    <RoundTripTime>7</RoundTripTime>
    <TimeStamp>2000-11-09T21:53:45Z</TimeStamp>
</Ping>
```

Protocol Interactions

- Consumer subscription
  - Consumer subscribes for events from a producer
  - Producer sends events to the consumer
  - Consumer unsubscribes for events
- Consumer query
  - Consumer asks the producer for an event and producer replies with the event

Assumptions

- TCP used for communication
- No authentication, authorization, or encryption
- XML is used to represent control and data messages
- One socket between a producer and a consumer for both control and data messages
- Multiple subscriptions active over a socket
- Multiple requests outstanding over a socket

Messages

- Subscribe request
- Subscribe reply
- Unsubscribe request
- Unsubscribe reply
- Event
- Query request
- Query reply

General Message Format

- Length (in bytes) as a 32-bit integer in network byte order
- XML tags that indicate the message type: <SubscribeRequest> … <SubscribeRequest>
- Message specific data inside the tags

Subscribe Request

- Consists of:
  - Event type (required)
  - Input parameters (optional)
  - requestID (required)

```
<SubscribeRequest requestID="2">
    <Ping>
        <Period>300</Period>
        <TargetHostName>bar.lbl.gov</TargetHostName>
    </Ping>
</SubscribeRequest>
```

Subscribe Reply

- Consists of:
  - Return (required)
  - ReturnDetail (optional)
  - SubscriptionID (optional)
  - requestID (required)

```
<SubscribeReply requestID="2">
    <Return>Success</Return>
    <SubscriptionID>1234</SubscriptionID>
</SubscribeReply>
```

```
<SubscribeReply requestID="2">
    <Return>Failure</Return>
    <ReturnDetail>The period is too small.</ReturnDetail>
</SubscribeReply>
```

Unsubscribe Request

- Consists of:
  - SubscriptionID (required)
  - requestID (required)

```
<UnsubscribeRequest requestID="9">
    <SubscriptionID>1234</SubscriptionID>
</UnsubscribeRequest>
```

Unsubscribe Reply

- Consists of:
  - Return (required).
  - ReturnDetail (optional).
  - requestID (required).

```
<UnsubscribeReply requestID="9">
    <Return>Success</Return>
</UnsubscribeReply>
```

```
<UnsubscribeReply requestID="9">
    <Return>Failure</Return>
    <ReturnDetail>Unknown subscription ID.</ReturnDetail>
</UnsubscribeReply>
```

Event

- Consists of:
  - subscriptionID (required)
  - An event (optional)
  - Error (optional)
  - ErrorDetail (optional)

Event Examples

```
<Event subscriptionID="1234">
    <UptimeCPULoad>
        <Load1>1.5</Load1>
        <Load5>1.6</Load5>
        <Load15>1.3</Load15>
        <TimeStamp>2000-11-09T21:51:45Z</TimeStamp>
    </UptimeCPULoad>
</Event>
```

```
<Event subscriptionID="1234">
    <UptimeCPULoad>
        <Error>Authorization</Error>
        <ErrorDetail>You are no longer authorized to receive
                     this information.</ErrorDetail>
    </UptimeCPULoad>
</Event>
```

Query Request

- Consists of:
  - Event type (required)
  - Input parameters (optional)
  - requestID (required)

```
<QueryRequest requestID="20">
    <Ping>
        <TargetHostName>bar.lbl.gov</TargetHostName>
    </Ping>
</QueryRequest>
```

Query Reply

- Consists of:
  - Event (optional)
  - Return (required)
  - ReturnDetail (optional)
  - requestID (required)

```
<QueryReply requestID=”20”>
    <Return>Success</Return>
    <Ping>
        <SourceHostName>foo.nas.nasa.gov</SourceHostName>
        <TargetHostName>bar.lbl.gov</TargetHostName>
        <RoundTripTime>7</RoundTripTime>
        <TimeStamp>2000-11-09T21:53:45Z </TimeStamp>
    </Ping>
</QueryReply>
```

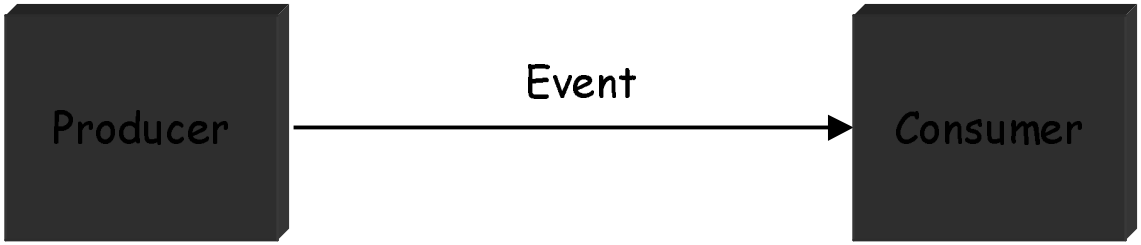
Subscribe Example

```
<SubscribeRequest requestID="11">
    <Ping>
        <Period>300</Period>
        <TargetHostName>bar.lbl.gov</TargetHostName>
    </Ping>
</SubscribeRequest>
```

Subscribe Example

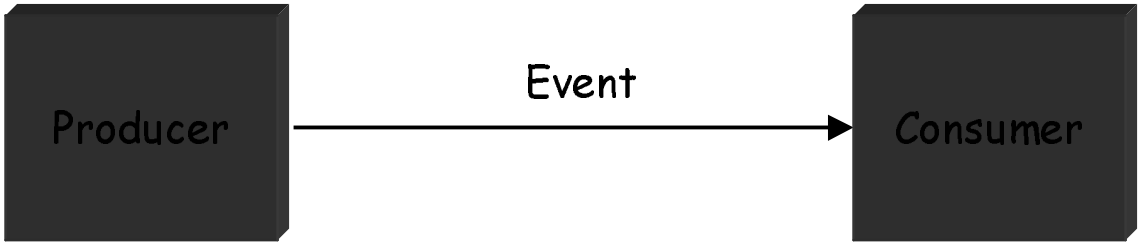
```
<SubscribeReply requestID="11">
    <Return>Success</Return>
    <SubscriptionID>4</SubscriptionID>
</SubscribeReply>
```

Subscribe Example




```
<Event subscriptionID="4">
    <Ping>
        <SourceHostName>foo.nas.nasa.gov</SourceHostName>
        <TargetHostName>bar.lbl.gov</TargetHostName>
        <RoundTripTime>7</RoundTripTime>
        <TimeStamp>2001-03-06T9:53:45Z</TimeStamp>
    </Ping>
</Event>
```

Subscribe Example




```
<Event subscriptionID="4">
    <Ping>
        <Error>Ping failed</Error>
        <ErrorDetail>Could not find target host in DNS.
          </ErrorDetail>
        <TimeStamp>2001-03-06T10:03:47Z</TimeStamp>
    </Ping>
</Event>
```

Subscribe Example

```
<UnsubscribeRequest requestID="25">
    <SubscriptionID>4</SubscriptionID>
</UnsubscribeRequest>
```

Subscribe Example

```
<UnsubscribeReply requestID="25">
    <Return>Success</Return>
</UnsubscribeReply>
```

Implementations

- Dan Gunter
  - Java
  - Xerces, Custom XML parser
  - www-didc.lbl.gov/prod-cons/index.html
- Warren Smith
  - C++
  - expat parser
  - Can't give it to you yet – lots of paperwork first
- Interoperability
  - Each ran an UptimeCPULoad producer
  - Other implementation could subscribe, receive events, unsubscribe, and query

Summary

- Defined a relatively simple XML-based producer-consumer protocol
- Two implementations of the protocol
- One implementation available to the community so far
- Gained and continue to gain experience

Ongoing Work

- Producer-initiated subscription
- Producer and consumer subscription IDs for each subscription
- Either side of subscription can initiate unsubscribe
- XML Schema for the protocol and events