Java is a trademark of Sun Microsystems, Inc.

JavaOne℠

Taking a SIP of Java

RJ Auburn
Voxeo Corporation
Chief Technology Officer

Telephony

Sucks

Expensive

Complex

Proprietary

This is not how it should be...

Simple

Ubiquitous

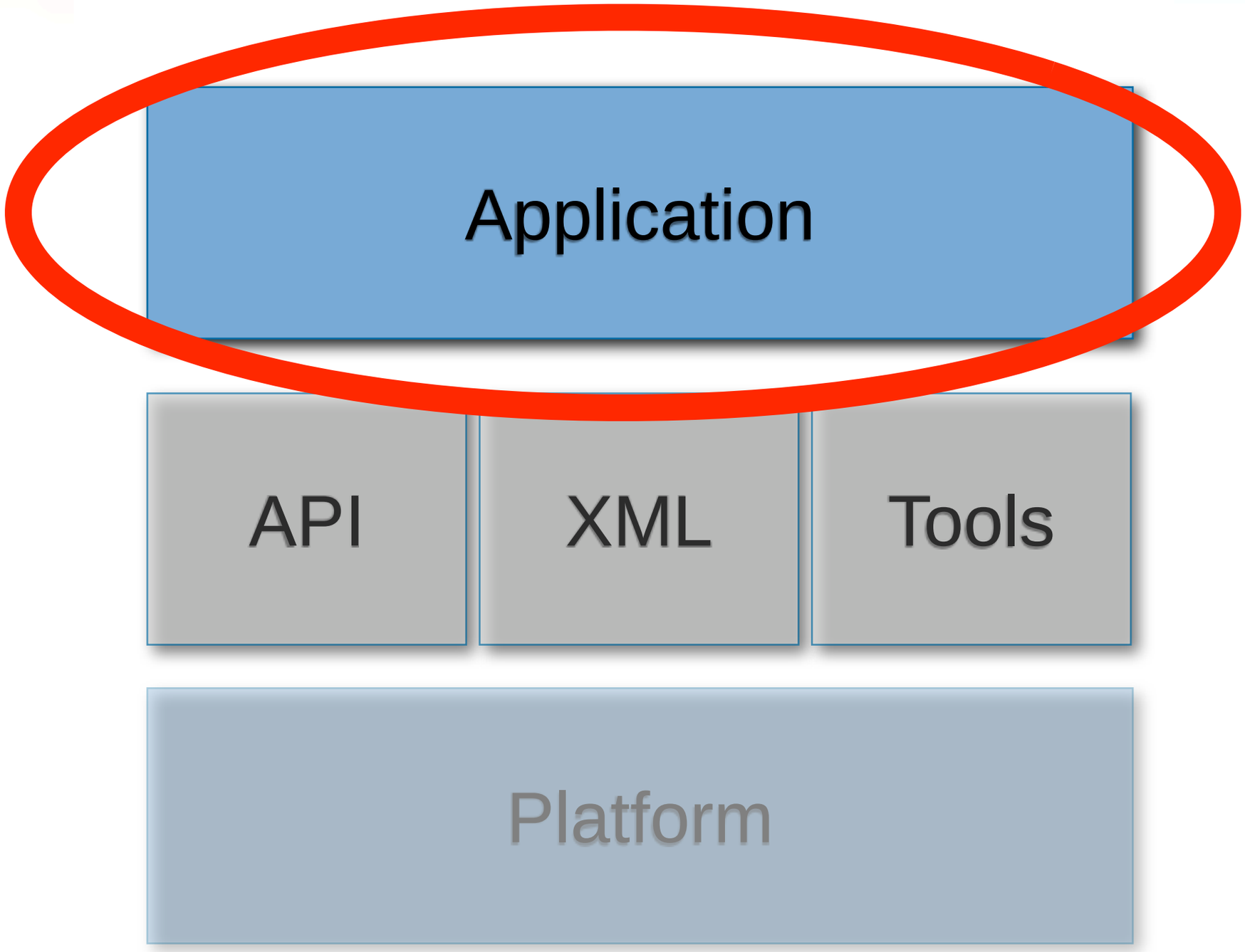
Open

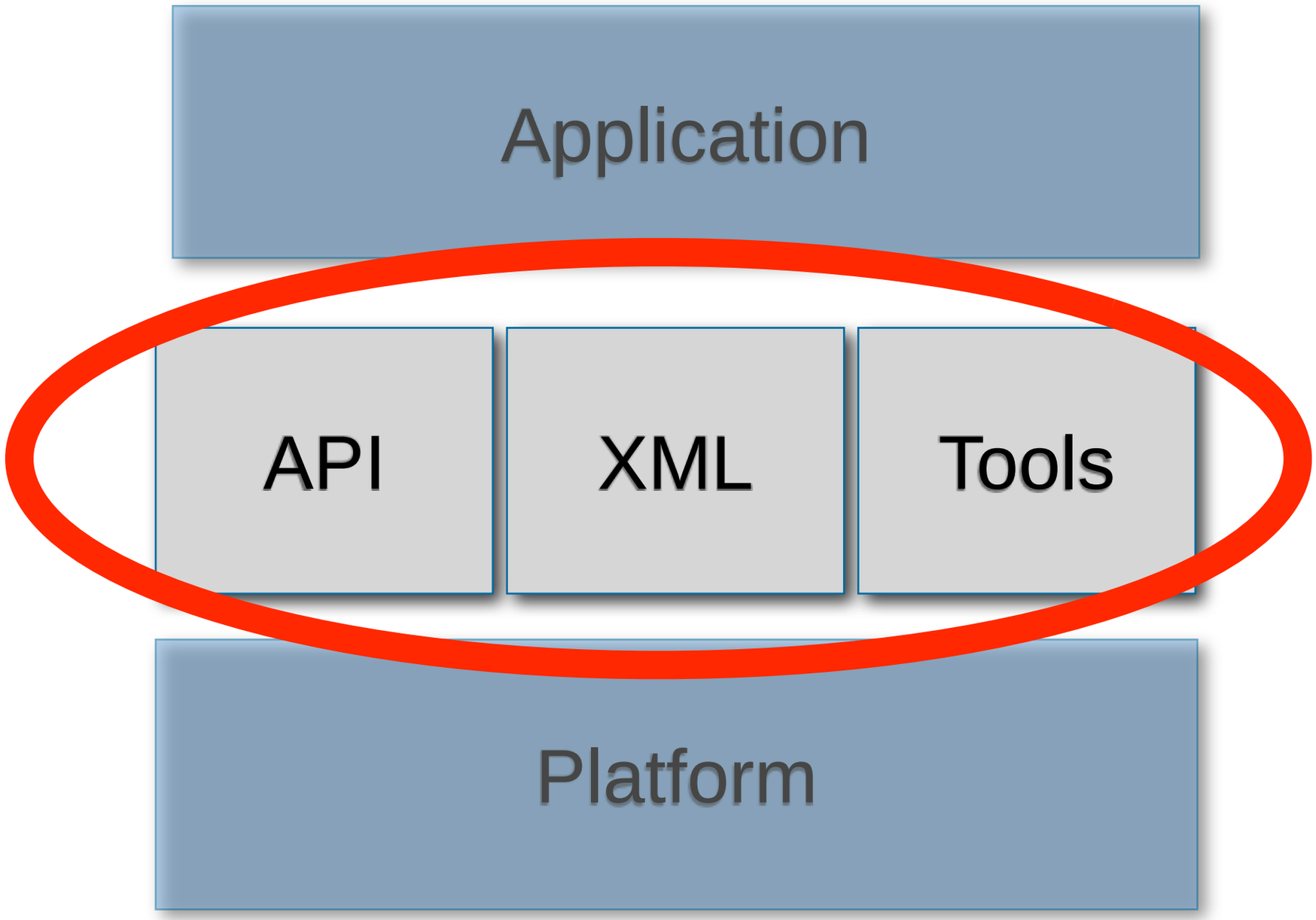
So what are the layers?

Application

API | XML | Tools

Platform

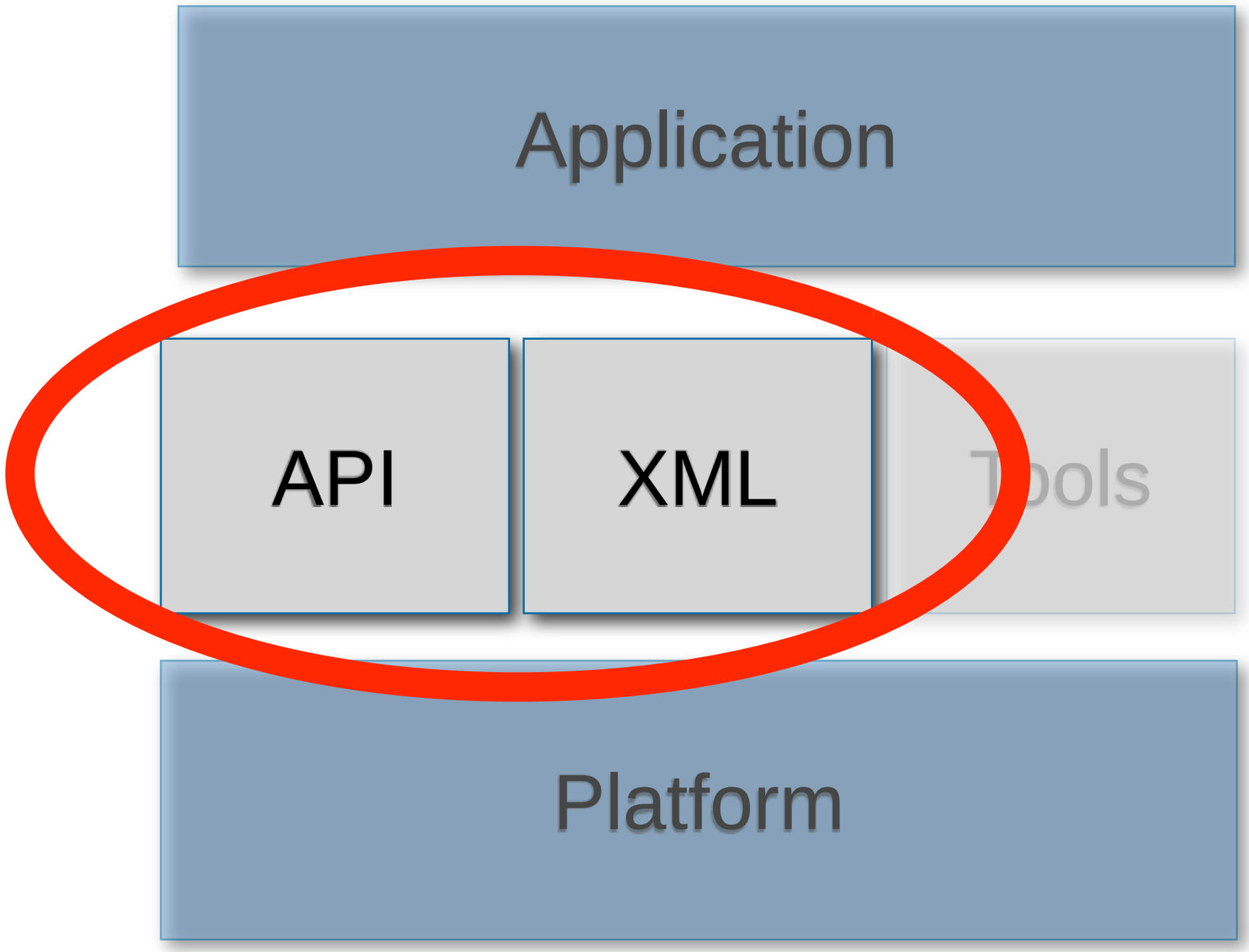

Application
API
XML
Tools
Platform


Application
API
XML
Tools
Platform

Application

API | XML | Tools

Platform


Application
API
XML
Tools
Platform

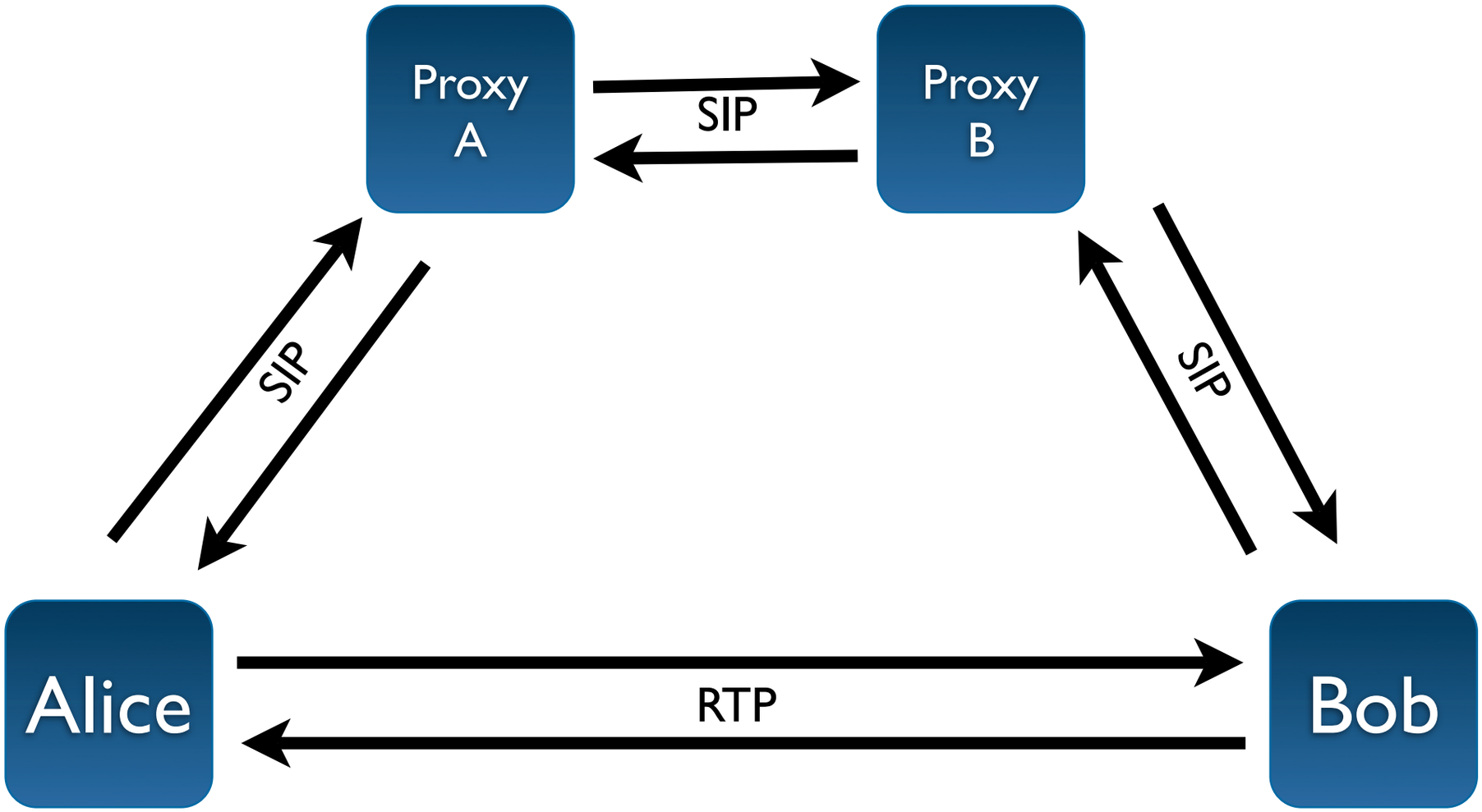
Some Basics:

Signaling vs Media

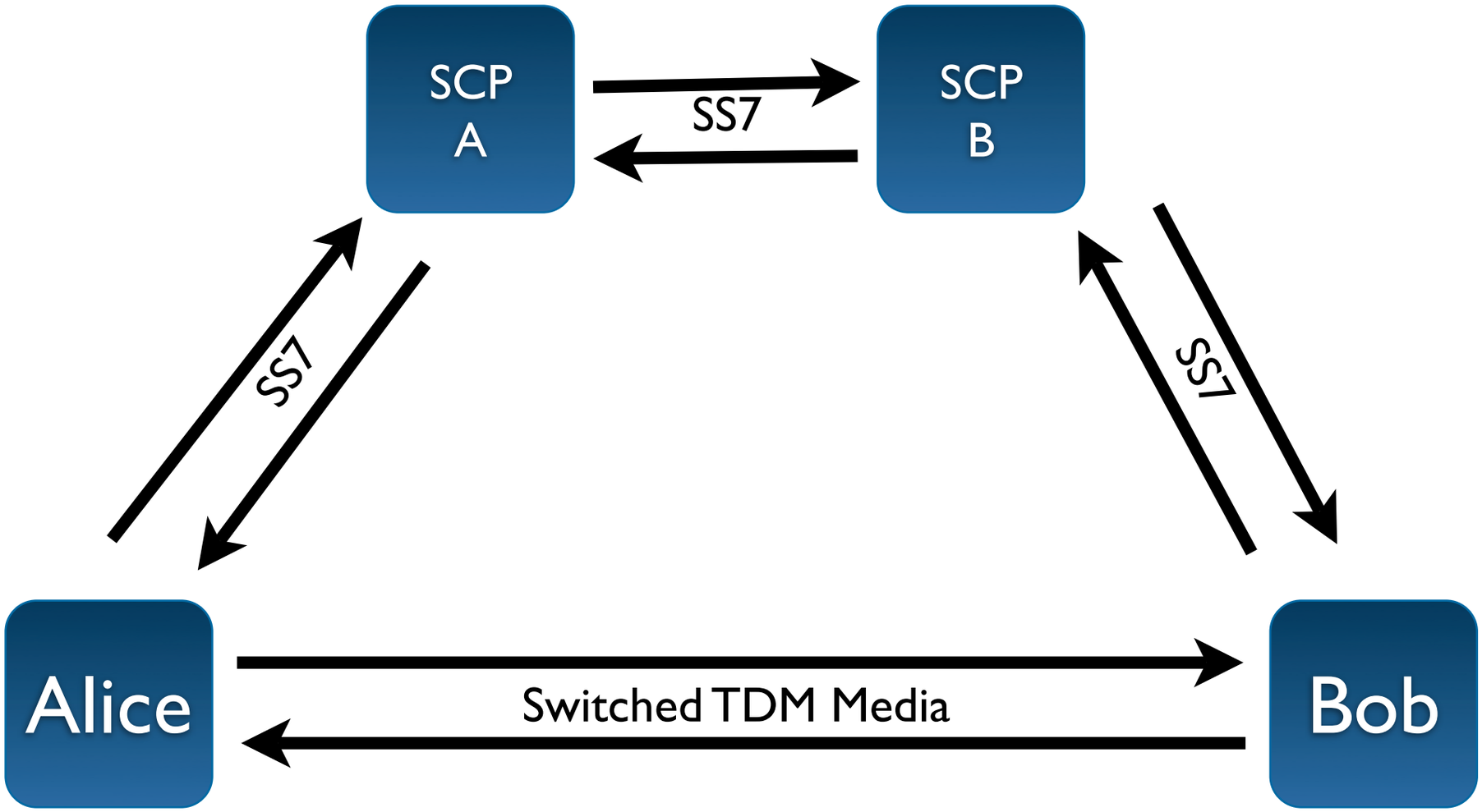
Call Control vs Media

- Phone systems often are split into two components: Signaling and Media
- Signaling handles the setup and tear-down of sessions and phone calls.
- Media is responsible for the transport voice path

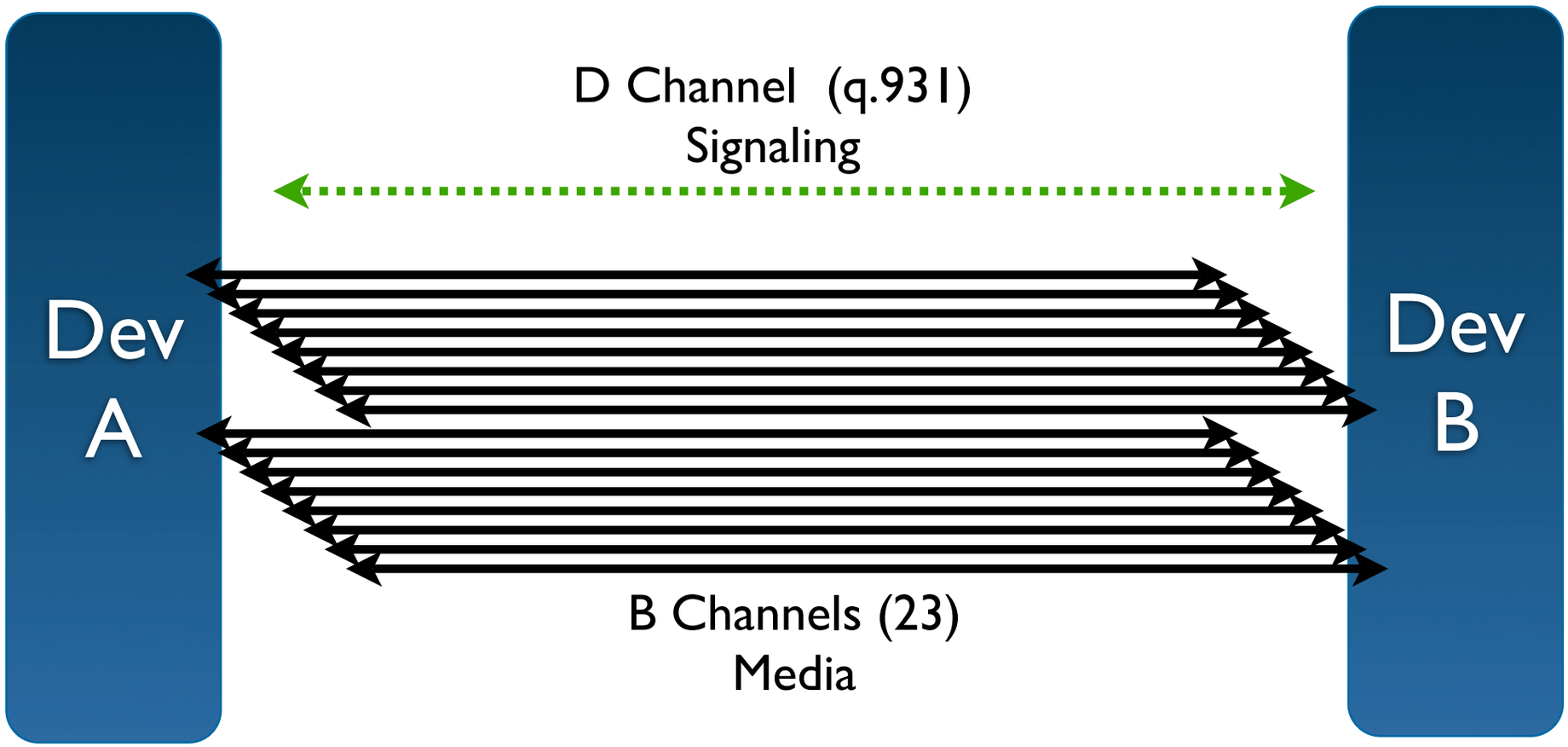
Media vs Call Control in the PSTN - SIP



Media vs Call Control in the PSTN - SS7



Media vs Call Control in the PSTN - ISDN



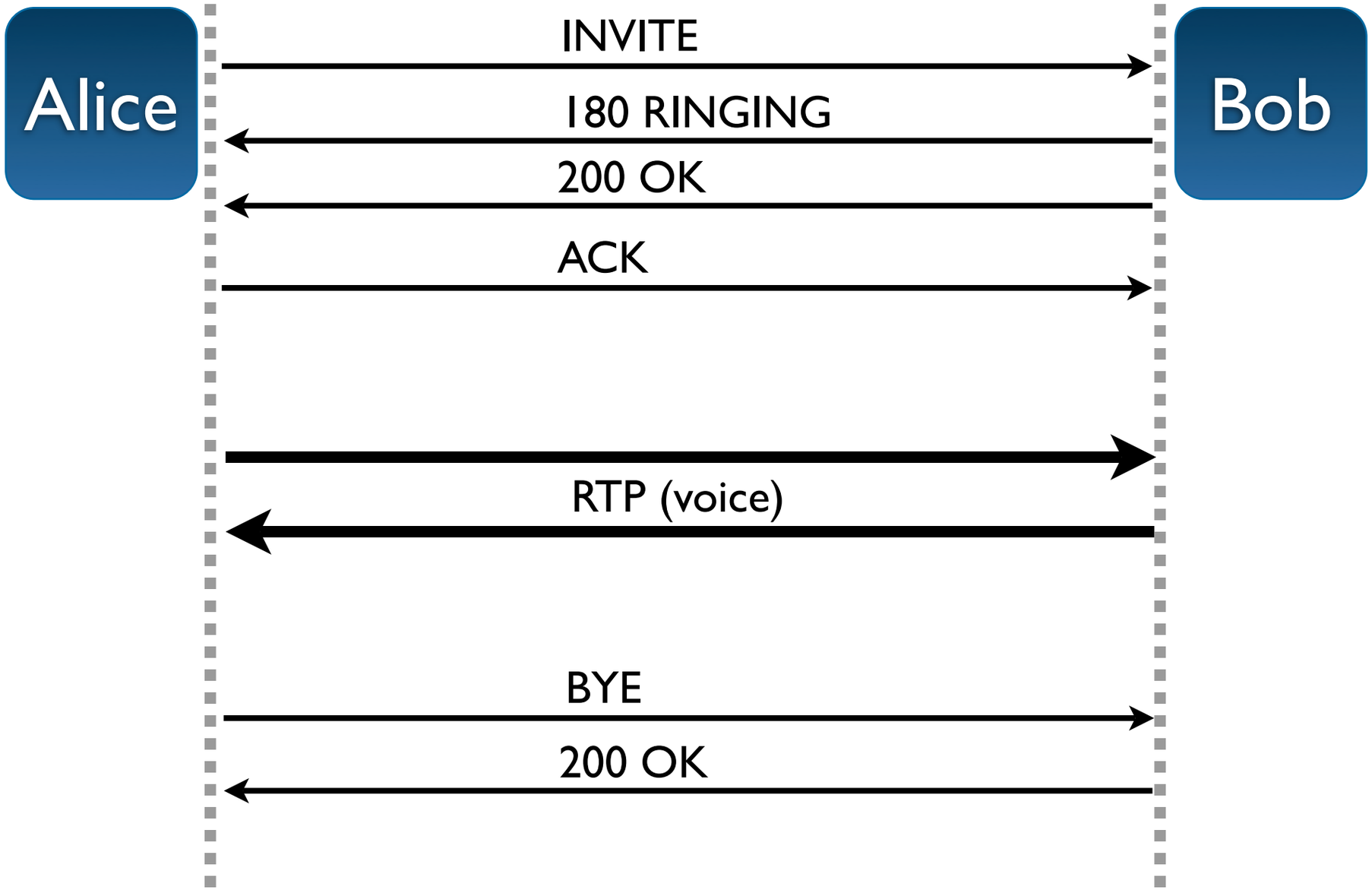
All this means is the API's and standards for media vs call control tend to be different.

So now for a bit about SIP (Call Control)

SIP

- Session Initiation Protocol (SIP) defines how to establish a communication session between two endpoints
- Primarily used for voice, but can for IM or virtually any other protocol
- Almost always used in client/server configuration with "SIP proxies" in control of "SIP endpoints"
  - Work going on in P2PSIP - see www.p2psip.org
- Text-based protocol, originally modeled on HTTP

SIP Communication



Major SIP Methods

- INVITE
- BYE
- REFER
- REGISTER
- SUBSCRIBE
- NOTIFY

  oh ya...
- INFO (wait, we are not supposed to talk about this one)

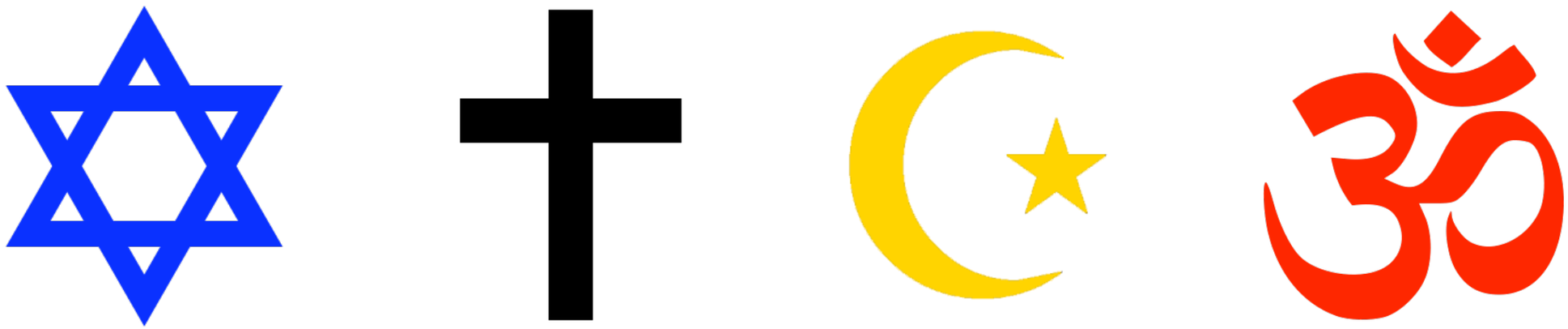
Response Codes

- 1xx - Provisional
- 2xx - All is good. Final
- 3xx - Redirects
- 4/5/6xx - Errors

SIP Resources

- Internet Engineering Task Force (IETF)
  - RFC 3261
  - Hitchhiker's Guide to SIP
- Open Source Info
  - VoIP Info Wiki: www.voip-info.org
- Industry Sites
  - SIP Forum: www.sipforum.org
  - SIP Foundry: www.sipfoundry.org

So lets talk about...



Religion

<?xml?>

Apache

solaris

Microsoft .net

python

Religion

<?xml?>

XML

VoiceXML and CCXML

VoiceXML

- Language created by the W3C to model computer human dialogs.
- Supports speech and touchtone.
- Built around a form filling model called the FIA.
- Voice equivalent to HTML.
- Focused on dialogs. Very limited call control.

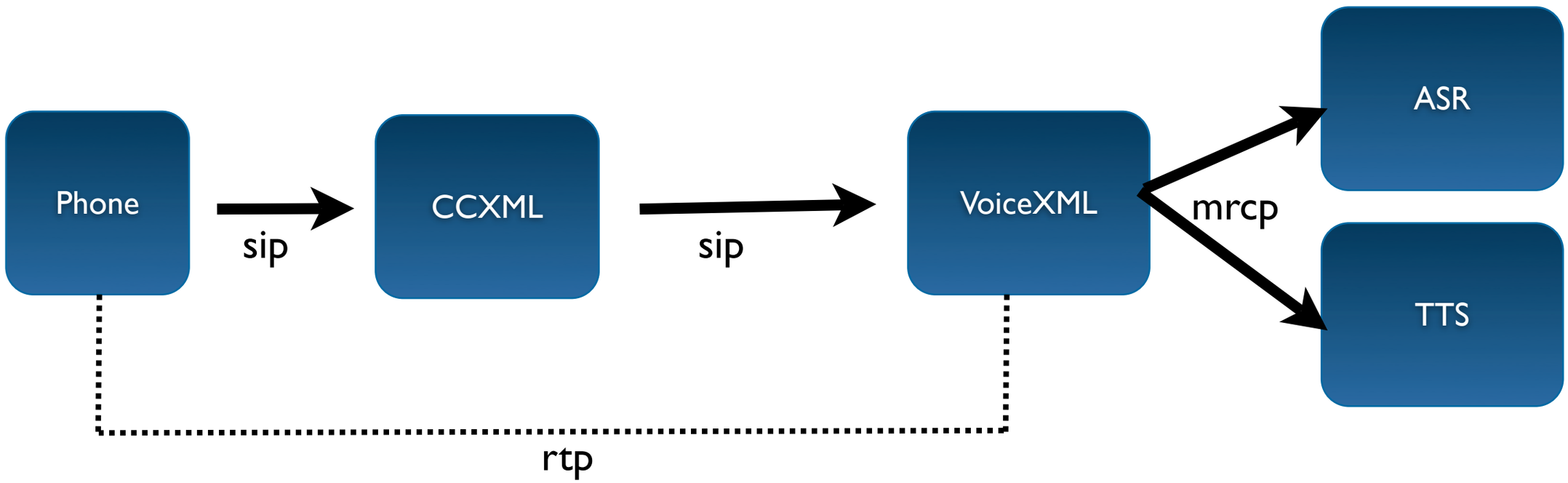
- http://www.w3.org/TR/voicexml/

CCXML

- Call Control XML (CCXML) is the W3C standard for call control using XML
- Sister standard to VoiceXML
- Integrates with VoiceXML for dialog control
- Provides a framework for issuing call control commands and handling call control events

- http://www.w3.org/TR/ccxml/

VoiceXML and CCXML Architecture



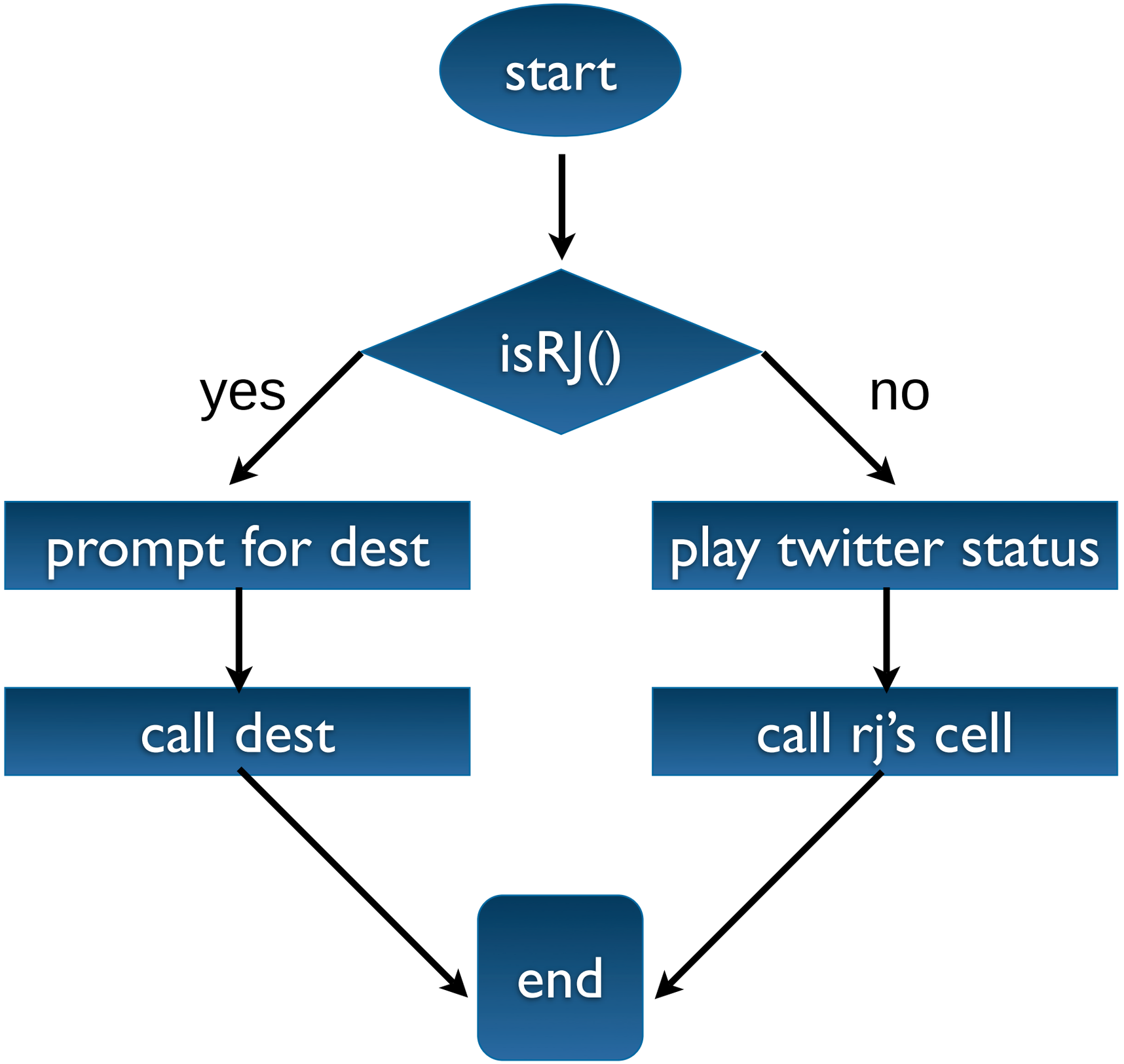
Like Tribbles...

It has taken over the Enterprise...

How about some code?

Sample CCXML/VoiceXML application.

- Caller dials in to the application
- Caller is bridged to the subscriber
- Results of the call attempt are posted to Twitter via their REST API


start
isRJ()
yes
no
prompt for dest
call dest
play twitter status
call rj's cell
end

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
        <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
        <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
        <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
        <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
       cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <createcall dest="'tel:18015551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
       <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
       <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml version="1.0"
      xmlns="http://www.w3.org/2002/09/ccxml">
</ccxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
       cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
       <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
       <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'connected'"/>
      <dialogstart src="'main.vxml'"
    </transition>
    <transition event="connection.connected" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
       <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
       <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

```
<var name="state" expr="'init'"/>
<var name="incomingcall"/>
<eventprocessor statevariable="state">
</eventprocessor>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
       cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
       <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
       <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
      <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
      <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

```
<transition event="connection.alerting" state="init">
  <assign name="incomingcall" expr="event$.connectionid"/>
  <accept/>
</transition>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
       cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
       <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
       <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
       cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="dialog.exited" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
       <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
       <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```


```
<transition event="connection.connected" state="init"
   cond="event$.connection.remote=='tel:+18315551234'">
  <assign name="state" expr="'rjconnected'"/>
  <dialogstart src="'rj.vxml'"
</transition>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <var name="tURL"
       expr="'http://zscgeek:password@twitter.com/statuses/update.xml'"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <accept/>
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <dialogprepare dialogid="connected"/>
    </transition>
    <transition event="connection.failed" state="calling">
      <assign name="state" expr="'done'"/>
      <var name="status" expr="'RJ is not answering his phone'"/>
      <send targettype="'basichttp'" name="'update'"
            target="tURL" namelist="status"/>
    </transition>
    <transition event="connection.disconnected" state="connected">
      <assign name="state" expr="'done'"/>
      <var name="status" expr="'RJ is off the phone'"/>
      <send targettype="'basichttp'" name="'update'"
            target="tURL" namelist="status"/>
    </transition>
    <transition event="send.successful" state="done">
      <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml xmlns="http://www.w3.org/2001/vxml" version="2.1">
    <form>
    <field name="dest" type="digits?length=10">
        <prompt>Welcome RJ. Please enter the phone
                number you wish to reach.</prompt>
    <filled>
        <exit namelist="dest"/>
    </filled>
    </field>
    </form>
</vxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
        <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
        <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
      <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
      <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

```
<transition event="connection.connected" state="init">
  <assign name="state" expr="'callconnected'"/>
  <dialogstart src="'caller.vxml'"
</transition>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml xmlns="http://www.w3.org/2001/vxml" version="2.1">
  <form>
    <block>
      <data src="http://twitter.com/statuses/
user_timeline.xml?screen_name=zscgeek"
                    name="twitter" ecmaxmltype="e4x" />
      <prompt>Thank you for calling RJ.
        We will connect you now.
        His twitter status is
        <value expr="twitter.statuses.status[0].text"/>
      </prompt>
    </block>
  </form>
</vxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
       <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
       <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
      <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
      <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
      <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
      <exit/>
    </transition>

  </eventprocessor>
</ccxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
       cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
       <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
       <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

```
<transition event="dialog.exit" state="rjconnected">
  <assign name="state" expr="'calling'"/>
  <createcall dest="event$.values.dest"/>
</transition>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
        <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
        <exit/>
    </transition>

  </eventprocessor>
</ccxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'calling'"/>
      <if cond="event$.connection.remote=='tel:+18315551234'"/>
      </if>
      <assign name="state" expr="'rjconnected'"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
      <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
      <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```


```
<transition event="connection.connected"
state="calling">
  <assign name="state" expr="'connected'"/>
  <join id1="event$.connectionid" id2="incomingcall"/>
</transition>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
        <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
        <exit/>
    </transition>

  </eventprocessor>
</ccxml>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="caller.vxml"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
      <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
      <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

```
<transition event="connection.failed" state="calling">
    <exit/>
</transition>
<transition event="connection.disconnected" state="connected">
    <exit/>
</transition>
```

Call router + Twitter

```
<?xml version="1.0" encoding="UTF-8"?>
<ccxml xmlns="http://www.w3.org/2002/09/ccxml" version="1.0">
  <var name="state" expr="'init'"/>
  <var name="incomingcall"/>
  <eventprocessor statevariable="state">
    <transition event="connection.alerting" state="init">
      <assign name="incomingcall" expr="event$.connectionid"/>
      <accept/>
    </transition>
    <transition event="connection.connected" state="init"
        cond="event$.connection.remote=='tel:+18315551234'">
      <assign name="state" expr="'rjconnected'"/>
      <dialogstart src="'rj.vxml'"
    </transition>
    <transition event="connection.connected" state="init">
      <assign name="state" expr="'callconnected'"/>
      <dialogstart src="'caller.vxml'"
    </transition>
    <transition event="dialog.exit" state="callconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="'tel:+18315551234'"/>
    </transition>
    <transition event="dialog.exit" state="rjconnected">
      <assign name="state" expr="'calling'"/>
      <createcall dest="event$.values.dest"/>
    </transition>
    <transition event="connection.connected" state="calling">
      <assign name="state" expr="'connected'"/>
      <join id1="event$.connectionid" id2="incomingcall"/>
    </transition>
    <transition event="connection.failed" state="calling">
        <exit/>
    </transition>
    <transition event="connection.disconnected" state="connected">
        <exit/>
    </transition>
  </eventprocessor>
</ccxml>
```

So... How about API's...

A Favorite of Carriers

Java

SIP Servlets

- Standard Java based API for writing SIP applications.
- 1.0 standardized as JSR-116.
- 1.1 just released as JSR-289
- Extends the HTTP Servlet model to support SIP and telephony applications
- http://www.sipservlet.com/
- Supported by a large number of application servers including Oracle (BEA), IBM, Sun, Voxeo.






Request Methods

- doInvite(SipServletRequest req);
- doAck(SipServletRequest req);
- doOptions(SipServletRequest req);
- doBye(SipServletRequest req);
- doCancel(SipServletRequest req);
- doSubscribe(SipServletRequest req);
- doNotify(SipServletRequest req);
- doMessage(SipServletRequest req);
- doInfo(SipServletRequest req);
- doPrack(SipServletRequest req);

Response Methods

- doProvisionalResponse(SipServletResponse res);
- doSuccessResponse(SipServletResponse res);
- doRedirectResponse(SipServletResponse res);
- doErrorResponse(SipServletResponse res);

Basic Request

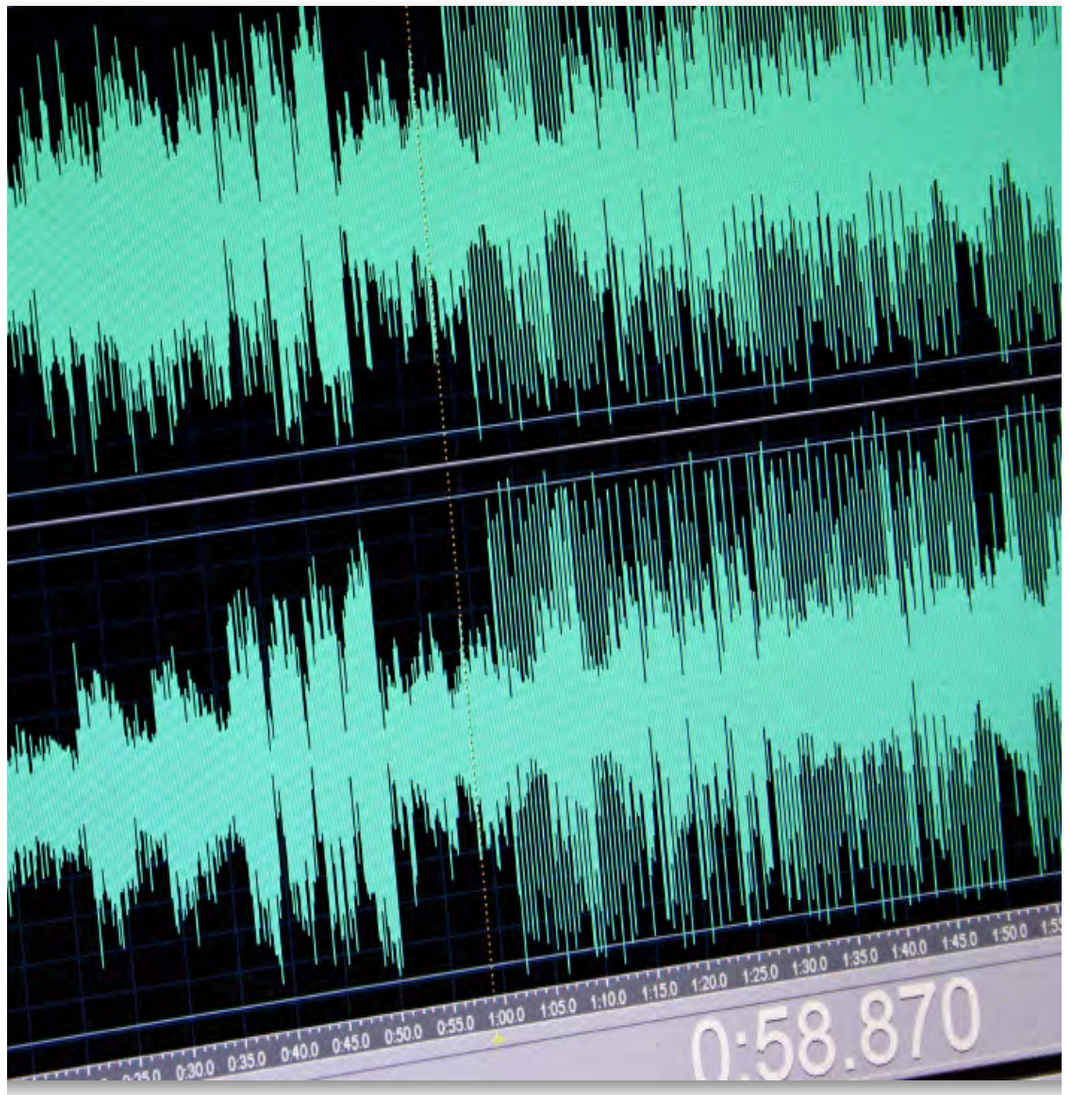
```
public class BasicSIPServlet extends SipServlet {
    protected void doInfo(SipServletRequest req)
       throws ServletException, IOException
    {
       req.createResponse(SipServletResponse.SC_TRYING).send();
       // do stuff
       req.createResponse(SipServletResponse.SC_OK).send();
    }
}
```

Accessing SIP Headers

```
protected void doInvite(SipServletRequest req) throws
ServletException, IOException {

 String cpc ;
 if (  req.getHeader("User-Agent").equals("Sonus")) {
    cpc =   req.getHeader("CallParty");
 } else if   (req.getHeader("User-Agent").equals("ZTE")) {
    cpc =   req.getHeader("CPS");
   }
}
```

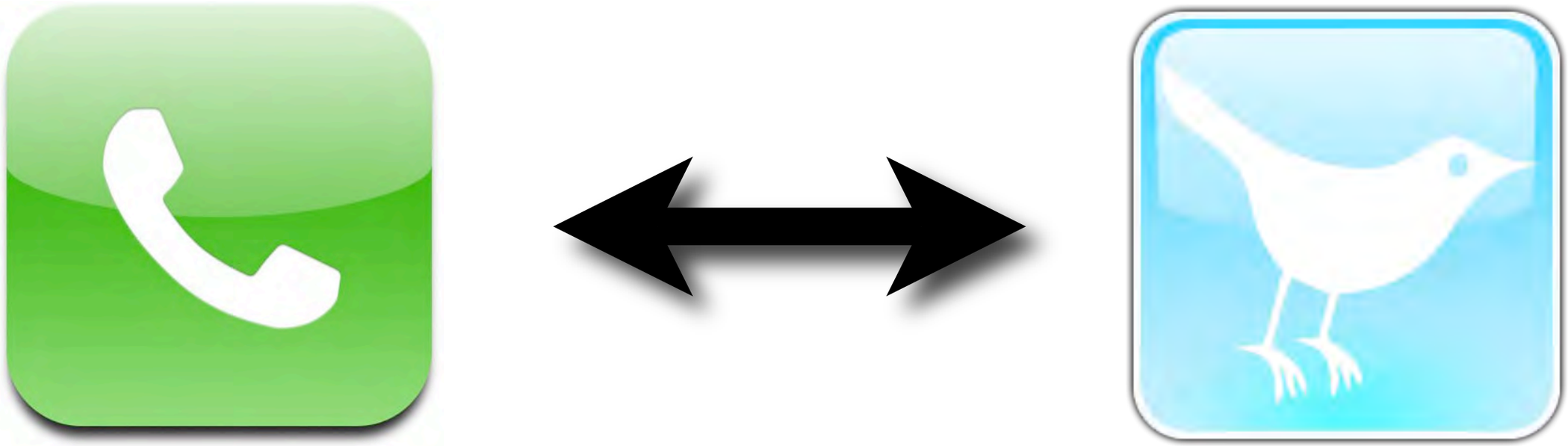
JSR-309

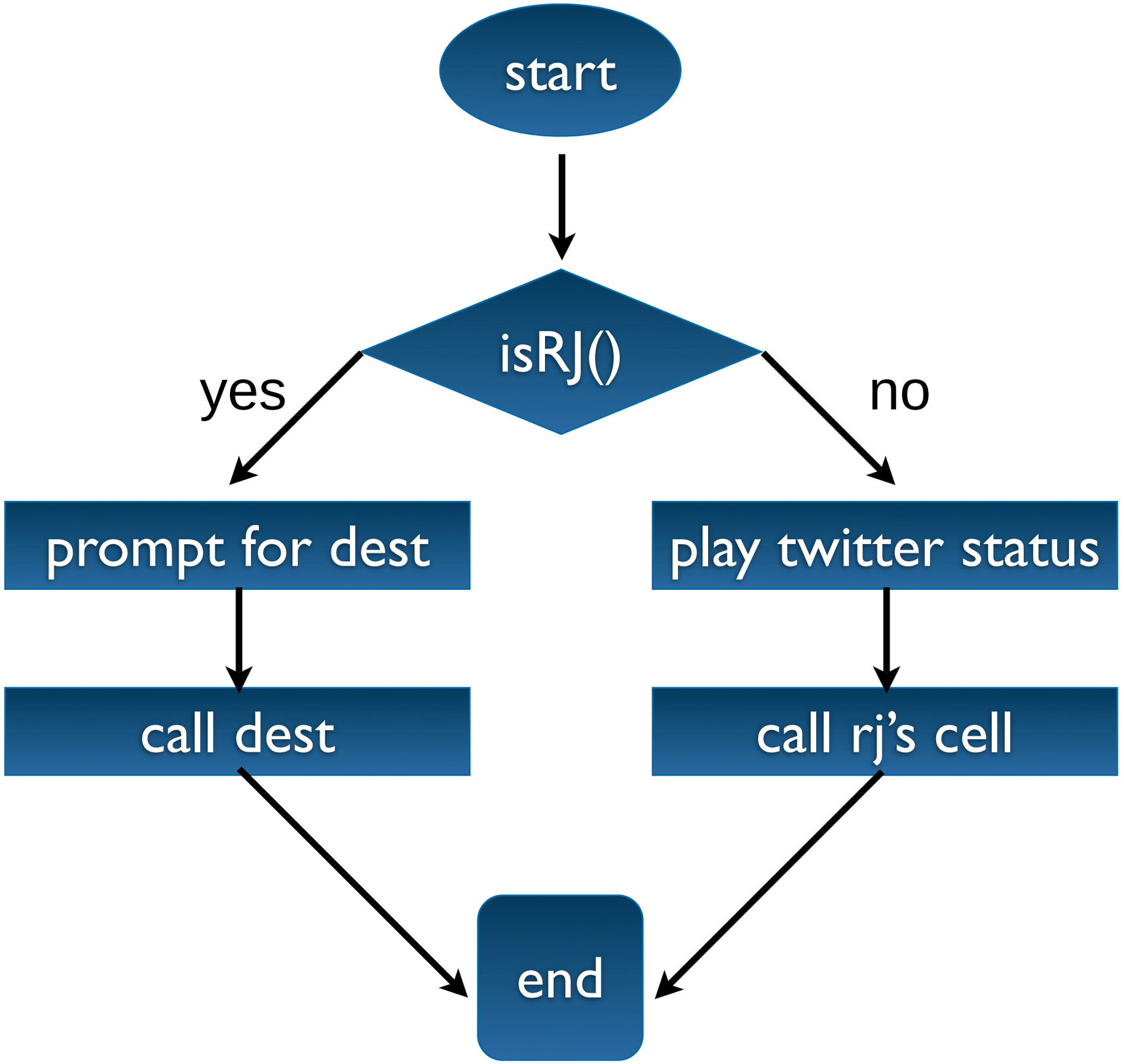
- Java Media Server API
- Based on the CCXML media model
- Still in draft stage
- Provides dialog resources, conferencing, media routing to Java applications



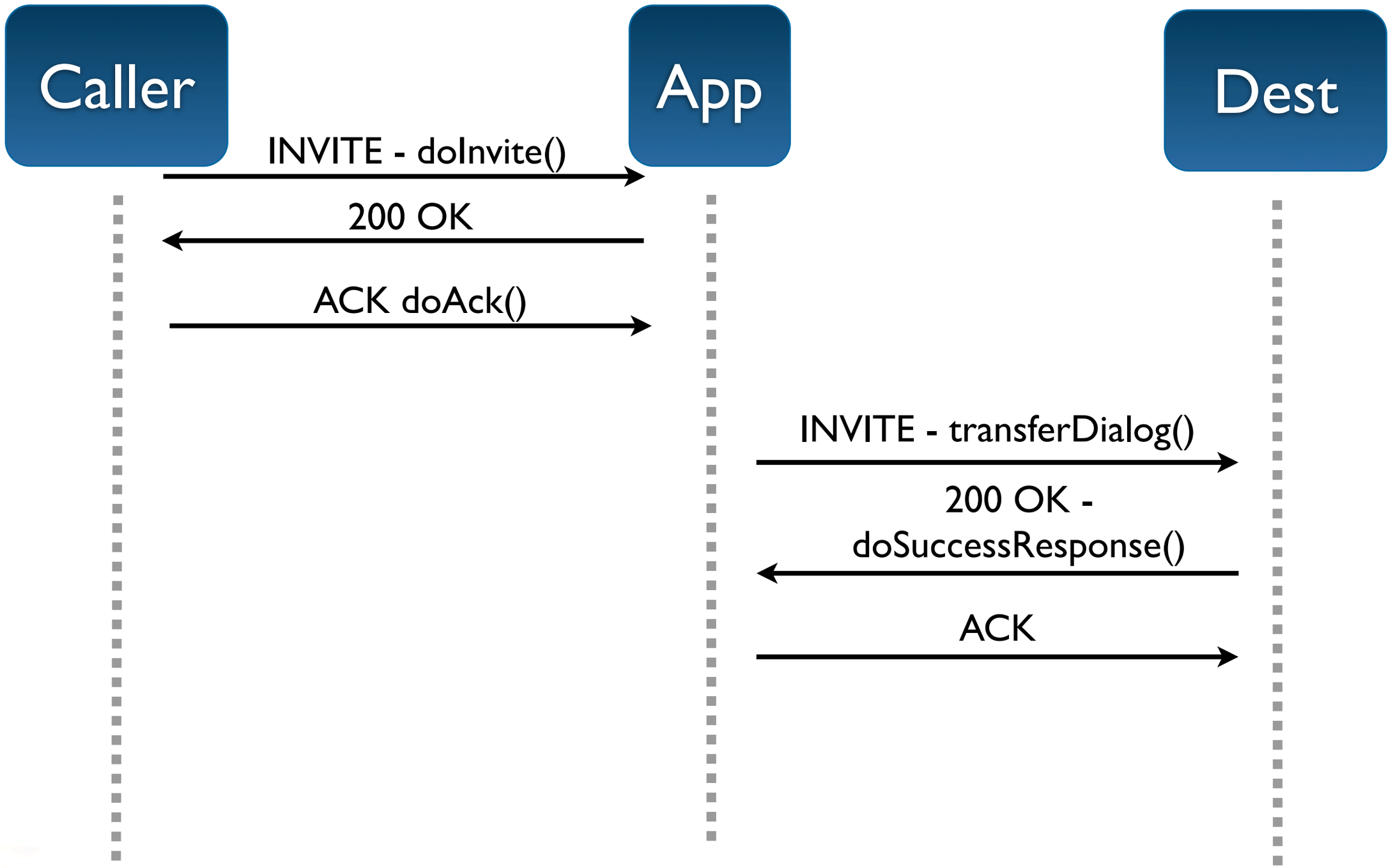
Sample Application Overview

- Lets try that application again in java...




start
isRJ()
yes
no
prompt for dest
call dest
play twitter status
call rj's cell
end

SIP Flow


Caller
App
Dest
INVITE - doInvite()
200 OK
ACK doAck()
INVITE - transferDialog()
200 OK - doSuccessResponse()
ACK

Call router + Twitter

```
public class TwitterSIPServlet extends SipServlet {
  SipFactory factory;

  public void init() throws ServletException {
    super.init();
    factory =(SipFactory)getServletContext().getAttribute("javax.servlet.sip.SipFactory");
  }

  protected void doInvite(SipServletRequest req) {
    if(req.isInitial()){
      SipApplicationSession sipApp = factory.createApplicationSession();
      TwitterSIPServletSession service = new TwitterSIPServletSession(req.getSession(),
factory);
      sipApp.setAttribute("router-app", service);
      service.init(req);
    }
  }

  protected void doSuccessResponse(SipServletResponse resp) {
    resp.createAck().send();
  }

  protected void doAck(SipServletRequest req) {
    final TwitterSIPServletSession service
          = (TwitterSIPServletSession) req.getApplicationSession().getAttribute("router-
app");
    service.startDialog();
  }

  protected void doBye(SipServletRequest req) {
    // Need to send BYE request to the other legs
  }
}
```

Call router + Twitter

```
public class TwitterSIPServlet extends SipServlet {
  SipFactory factory;

  public void init() throws ServletException {
    super.init();
    factory =(SipFactory)getServletContext().getAttribute("javax.servlet.sip.SipFactory");
  }

  protected void doInvite(SipServletRequest req) {
    if(req.isInitial()){
      SipApplicationSession sipApp = factory.createApplicationSession();
      TwitterSIPServletSession service = new TwitterSIPServletSession(req.getSession(),
factory);
      sipApp.setAttribute("router-app", service);
      service.init(req);
    }
  }

  protected void doSuccessResponse(SipServletResponse resp) {
    resp.createAck().send();
  }

  protected void doAck(SipServletRequest req) {
    final TwitterSIPServletSession service
          = (TwitterSIPServletSession) req.getApplicationSession().getAttribute("router-
app");
    service.startDialog();
  }

  protected void doBye(SipServletRequest req) {
    // Need to send BYE request to the other legs
  }
}
```

Call router + Twitter

```
public class TwitterSIPServlet extends SipServlet {
  SipFactory factory;

  public void init() throws ServletException {
    super.init();
    factory = (SipFactory)getServletContext()
                .getAttribute("javax.servlet.sip.SipFactory");
  }
}
```

Call router + Twitter

```
public class TwitterSIPServlet extends SipServlet {
  SipFactory factory;

  public void init() throws ServletException {
    super.init();
    factory =(SipFactory)getServletContext().getAttribute("javax.servlet.sip.SipFactory");
  }

  protected void doInvite(SipServletRequest req) {
    if(req.isInitial()){
      SipApplicationSession sipApp = factory.createApplicationSession();
      TwitterSIPServletSession service = new TwitterSIPServletSession(req.getSession(),
factory);
      sipApp.setAttribute("router-app", service);
      service.init(req);
    }
  }

  protected void doSuccessResponse(SipServletResponse resp) {
    resp.createAck().send();
  }

  protected void doAck(SipServletRequest req) {
    final TwitterSIPServletSession service
          = (TwitterSIPServletSession) req.getApplicationSession().getAttribute("router-
app");
    service.startDialog();
  }

  protected void doBye(SipServletRequest req) {
    // Need to send BYE request to the other legs
  }
}
```

Call router + Twitter

```
protected void doInvite(SipServletRequest req) {
  if(req.isInitial()){
    SipApplicationSession sipApp
                          = factory.createApplicationSession();
    TwitterSIPServletSession service
      = new TwitterSIPServletSession(req.getSession(), factory);
    sipApp.setAttribute("router-app", service);
    service.init(req);
  }
}
```

Call router + Twitter

```
public class TwitterSIPServlet extends SipServlet {
  SipFactory factory;

  public void init() throws ServletException {
    super.init();
    factory =(SipFactory)getServletContext().getAttribute("javax.servlet.sip.SipFactory");
  }

  protected void doInvite(SipServletRequest req) {
    if(req.isInitial()){
      SipApplicationSession sipApp = factory.createApplicationSession();
      TwitterSIPServletSession service = new TwitterSIPServletSession(req.getSession(),
factory);
      sipApp.setAttribute("router-app", service);
      service.init(req);
    }
  }

  protected void doSuccessResponse(SipServletResponse resp) {
    resp.createAck().send();
  }

  protected void doAck(SipServletRequest req) {
    final TwitterSIPServletSession service
          = (TwitterSIPServletSession) req.getApplicationSession().getAttribute("router-
app");
    service.startDialog();
  }

  protected void doBye(SipServletRequest req) {
    // Need to send BYE request to the other legs
  }
}
```

Call router + Twitter

```
public class TwitterSIPServlet extends SipServlet {
  SipFactory factory;

  public void init() throws ServletException {
    super.init();
    factory =(SipFactory)getServletContext().getAttribute("javax.servlet.sip.SipFactory");
  }

  protected void doInvite(SipServletRequest req) {
    if(req.isInitial()){
      SipApplicationSession sipApp = factory.createApplicationSession();
      TwitterSIPServletSession service = new TwitterSIPServletSession(req.getSession(),
```

```
  protected void doSuccessResponse(SipServletResponse resp)  {
    resp.createAck().send();
  }
```

```
  protected void doAck(SipServletRequest req) {
    final TwitterSIPServletSession service
            = (TwitterSIPServletSession) req.getApplicationSession().getAttribute("router-
app");
    service.startDialog();
  }

  protected void doBye(SipServletRequest req) {
    // Need to send BYE request to the other legs
  }
}
```

Call router + Twitter

```
public class TwitterSIPServlet extends SipServlet {
  SipFactory factory;

  public void init() throws ServletException {
    super.init();
    factory =(SipFactory)getServletContext().getAttribute("javax.servlet.sip.SipFactory");
  }

  protected void doInvite(SipServletRequest req) {
    if(req.isInitial()){
      SipApplicationSession sipApp = factory.createApplicationSession();
      TwitterSIPServletSession service = new TwitterSIPServletSession(req.getSession(),
factory);
      sipApp.setAttribute("router-app", service);
      service.init(req);
    }
  }

  protected void doSuccessResponse(SipServletResponse resp) {
    resp.createAck().send();
  }

  protected void doAck(SipServletRequest req) {
    final TwitterSIPServletSession service
          = (TwitterSIPServletSession) req.getApplicationSession().getAttribute("router-
app");
    service.startDialog();
  }

  protected void doBye(SipServletRequest req) {
    // Need to send BYE request to the other legs
  }
}
```

Call router + Twitter

```
protected void doAck(SipServletRequest req)  {
  final TwitterSIPServletSession service
      = (TwitterSIPServletSession) req.getApplicationSession()
                    .getAttribute("router-app");
  service.startDialog();
}
```

Call router + Twitter

```
public class TwitterSIPServlet extends SipServlet {
  SipFactory factory;

  public void init() throws ServletException {
    super.init();
    factory =(SipFactory)getServletContext().getAttribute("javax.servlet.sip.SipFactory");
  }

  protected void doInvite(SipServletRequest req) {
    if(req.isInitial()){
      SipApplicationSession sipApp = factory.createApplicationSession();
      TwitterSIPServletSession service = new TwitterSIPServletSession(req.getSession(),
factory);
      sipApp.setAttribute("router-app", service);
      service.init(req);
    }
  }

  protected void doSuccessResponse(SipServletResponse resp) {
    resp.createAck().send();
  }

  protected void doAck(SipServletRequest req) {
    final TwitterSIPServletSession service
          = (TwitterSIPServletSession) req.getApplicationSession().getAttribute("router-
app");
    service.startDialog();
  }

  protected void doBye(SipServletRequest req) {
    // Need to send BYE request to the other legs
  }
}
```

Call router + Twitter

```
protected void doBye(SipServletRequest req)  {
  // Who the frack has cleanup code in
  // slide demo's anyway?
}
```

Call router + Twitter

```
public class TwitterSIPServlet extends SipServlet {
  SipFactory factory;

  public void init() throws ServletException {
    super.init();
    factory =(SipFactory)getServletContext().getAttribute("javax.servlet.sip.SipFactory");
  }

  protected void doInvite(SipServletRequest req) {
    if(req.isInitial()){
      SipApplicationSession sipApp = factory.createApplicationSession();
      TwitterSIPServletSession service = new TwitterSIPServletSession(req.getSession(),
factory);
      sipApp.setAttribute("router-app", service);
      service.init(req);
    }
  }

  protected void doSuccessResponse(SipServletResponse resp) {
    resp.createAck().send();
  }

  protected void doAck(SipServletRequest req) {
    final TwitterSIPServletSession service
          = (TwitterSIPServletSession) req.getApplicationSession().getAttribute("router-
app");
    service.startDialog();
  }

  protected void doBye(SipServletRequest req) {
    // Need to send BYE request to the other legs
  }
}
```

Call router + Twitter

```
public class TwitterSIPServletSession  {
   final SipSession mySipSession;
   final SipFactory myFactory;
   MediaGroup myMediaGroup;
   boolean isRJ = false;

   public TwitterSIPServletSession(SipSession aSipSession, SipFactory aFactory) {
      ...
   }

   public void init(final SipServletRequest req) {
      ...
   }

   public void startDialog() {
      ...
   }


   class CallerPlayerListener implements MediaEventListener<PlayerEvent> {
      ...
   }

   class RJDialerSignalDetectorListener
                              implements MediaEventListener<SignalDetectorEvent> {
      ...
   }

   void transferDialog(String uri) throws ServletParseException, IOException {
      ...
   }
}
```

Call router + Twitter

```
public class TwitterSIPServletSession  {
    final SipSession mySipSession;
    final SipFactory myFactory;
    MediaGroup myMediaGroup;
    boolean isRJ = false;

    public TwitterSIPServletSession(SipSession aSipSession, SipFactory aFactory) {
        ...
    }

    public void init(final SipServletRequest req) {
        ...
    }

    public void startDialog() {
        ...
    }


    class CallerPlayerListener implements MediaEventListener<PlayerEvent> {
        ...
    }

    class RJDialerSignalDetectorListener
                            implements MediaEventListener<SignalDetectorEvent> {
        ...
    }

    void transferDialog(String uri) throws ServletParseException, IOException {
        ...
    }
}
```

Call router + Twitter

```
public class TwitterSIPServletSession  {
    final SipSession mySipSession;
    final SipFactory myFactory;
    MediaGroup myMediaGroup;
    boolean isRJ = false;

    public TwitterSIPServletSession(SipSession aSipSession,
                                    SipFactory aFactory) {
       mySipSession = aSipSession;
       myFactory = aFactory;
    }
}
```

Call router + Twitter

```
public class TwitterSIPServletSession  {
   final SipSession mySipSession;
   final SipFactory myFactory;
   MediaGroup myMediaGroup;
   boolean isRJ = false;

   public TwitterSIPServletSession(SipSession aSipSession, SipFactory aFactory) {
      ...
   }

   public void init(final SipServletRequest req) {
      ...
   }

   public void startDialog() {
      ...
   }


   class CallerPlayerListener implements MediaEventListener<PlayerEvent> {
      ...
   }

   class RJDialerSignalDetectorListener
                           implements MediaEventListener<SignalDetectorEvent> {
      ...
   }

   void transferDialog(String uri) throws ServletParseException, IOException {
      ...
   }
}
```

Call router + Twitter

```
public void init(final SipServletRequest req) throws ServletException {
   MediaSession myMediaSession = MediaSessionFactory.createMediaSession();

   NetworkConnection myNetworkConnection =
myMediaSession.createContainer(NetworkConnectionConfig.c_Basic);

    myMediaGroup =
myMediaSession.createContainer(MediaGroupConfig.c_PlayerSignalDetector);

   MediaEventListener<NetworkConnectionEvent> myNetworkConnectionListener
            = new MediaEventListener<NetworkConnectionEvent>() {
        ....
   };
   myNetworkConnection.addListener(myNetworkConnectionListener);

   myNetworkConnection.modify("", new String(req.getRawContent()));
}
```

Call router + Twitter

```
MediaEventListener<NetworkConnectionEvent> myNetworkConnectionListener = new
MediaEventListener<NetworkConnectionEvent>() {
   public void onEvent(NetworkConnectionEvent anEvent) {
      if (NetworkConnectionConstants.ev_Modify.equals(anEvent.getEventID())) {
         if (req.getFrom().getURI().getUser().equals("8312392883")) {
           isRJ = true;
           myMediaGroup.getSignalDetector().addListener(
                       new RJDialerSignalDetectorListener());
         } else {
           myMediaGroup.getSignalDetector().addListener(
                       new CallerPlayerListener());
         }
         myMediaGroup.join(Direction.DUPLEX, myNetworkConnection);
         String sdpAnswer = myNetworkConnection.getRawLocalSessionDescription();
         SipServletMessage msg = req.createResponse(200, "OK");
         msg.setContent(sdpAnswer.getBytes(), "application/sdp");
         msg.send();
      }
   }
};
```

Call router + Twitter

```
public class TwitterSIPServletSession  {
   final SipSession mySipSession;
   final SipFactory myFactory;
   MediaGroup myMediaGroup;
   boolean isRJ = false;

   public TwitterSIPServletSession(SipSession aSipSession, SipFactory aFactory) {
      ...
   }

   public void init(final SipServletRequest req) {
      ...
   }

   public void startDialog() {
      ...
   }

   class CallerPlayerListener implements MediaEventListener<PlayerEvent> {
      ...
   }

   class RJDialerSignalDetectorListener
                           implements MediaEventListener<SignalDetectorEvent> {
      ...
   }

   void transferDialog(String uri) throws ServletParseException, IOException {
      ...
   }
}
```

Call router + Twitter

```
public void startDialog() {
    if (isRJ) {
      Parameters collectOptions = mediaSessionFactory.createParameters();
        URI prompt = URI.create("data:application/ssml+xml,"+
           "<?xml version=\"1.0\"?>"+
            "<speak>"+
                  "Welcome RJ. Please enter the phone number you wish to reach."+
            "</speak>");
      collectOptions.put(SignalDetectorConstants.p_Prompt, prompt);
      mg.getSignalDetector().receiveSignals(10, null, RTC.bargeIn, collectOptions);
    } else {
      net.unto.twitter.Api twitter_api
                  = new net.unto.twitter.Api("zscgeek", "password");
      URI prompt = URI.create("data:application/ssml+xml,"+
          "<?xml version=\"1.0\"?>"+
          "<speak>"+
              "Thank you for calling RJ. We will connect you now."+
              "His twitter status is:"+
              twitter_api.getStatus()+
          "</speak>");
      myMediaGroup.getPlayer().play(prompt, null, null);
    }
}
```

Call router + Twitter

```
public class TwitterSIPServletSession  {
   final SipSession mySipSession;
   final SipFactory myFactory;
   MediaGroup myMediaGroup;
   boolean isRJ = false;

   public TwitterSIPServletSession(SipSession aSipSession, SipFactory aFactory) {
      ...
   }

   public void init(final SipServletRequest req) {
      ...
   }

   public void startDialog() {
      ...
   }

   class CallerPlayerListener implements MediaEventListener<PlayerEvent> {
      ...
   }

   class RJDialerSignalDetectorListener
                           implements MediaEventListener<SignalDetectorEvent> {
      ...
   }

   void transferDialog(String uri) throws ServletParseException, IOException {
      ...
   }
}
```

Call router + Twitter

```
class CallerPlayerListener implements MediaEventListener<PlayerEvent> {
   public void onEvent(PlayerEvent anEvent) {
      log("Collected: "+anEvent.getSignalString());
      MediaSession mediaSession =
                        anEvent.getSource().getMediaSession().release();
      transferDialog("sip:+18315551234@gateway:5060");
   }
}
```

Call router + Twitter

```
public class TwitterSIPServletSession  {
   final SipSession mySipSession;
   final SipFactory myFactory;
   MediaGroup myMediaGroup;
   boolean isRJ = false;

   public TwitterSIPServletSession(SipSession aSipSession, SipFactory aFactory) {
      ...
   }

   public void init(final SipServletRequest req) {
      ...
   }

   public void startDialog() {
      ...
   }


   class CallerPlayerListener implements MediaEventListener<PlayerEvent> {
      ...
   }

   class RJDialerSignalDetectorListener
                           implements MediaEventListener<SignalDetectorEvent> {
      ...
   }

   void transferDialog(String uri) throws ServletParseException, IOException {
      ...
   }
}
```

Call router + Twitter

```
class RJDialerSignalDetectorListener implements
MediaEventListener<SignalDetectorEvent> {
   public void onEvent(SignalDetectorEvent anEvent) {
      log("Collected: "+anEvent.getSignalString());
      MediaSession mediaSession =
                  anEvent.getSource().getMediaSession().release();
      transferDialog("sip:" + anEvent.getSignalString()
                  + "@gateway:5060");
   }
}
```

Call router + Twitter

```
public class TwitterSIPServletSession  {
   final SipSession mySipSession;
   final SipFactory myFactory;
   MediaGroup myMediaGroup;
   boolean isRJ = false;

   public TwitterSIPServletSession(SipSession aSipSession, SipFactory aFactory) {
      ...
   }

   public void init(final SipServletRequest req) {
      ...
   }

   public void startDialog() {
      ...
   }


   class CallerPlayerListener implements MediaEventListener<PlayerEvent> {
      ...
   }

   class RJDialerSignalDetectorListener
                              implements MediaEventListener<SignalDetectorEvent> {
      ...
   }

   void transferDialog(String uri) throws ServletParseException, IOException {
      ...
   }
}
```

Call router + Twitter

```
void transferDialog(String uri)  {
    SipServletRequest request =
          myFactory.createRequest(mySipSession.getApplicationSession(),
                                  "INVITE",
                                  "sip:addressbook@sip-as-uri:5060",
                                  uri);
    request.send();
}
```

Call router + Twitter

```
public class TwitterSIPServletSession  {
   final SipSession mySipSession;
   final SipFactory myFactory;
   MediaGroup myMediaGroup;
   boolean isRJ = false;

   public TwitterSIPServletSession(SipSession aSipSession, SipFactory aFactory) {
      ...
   }

   public void init(final SipServletRequest req) {
      ...
   }

   public void startDialog() {
      ...
   }


   class CallerPlayerListener implements MediaEventListener<PlayerEvent> {
      ...
   }

   class RJDialerSignalDetectorListener
                              implements MediaEventListener<SignalDetectorEvent> {
      ...
   }

   void transferDialog(String uri) throws ServletParseException, IOException {
      ...
   }
}
```

Made it to the finish line!

So. We Have Java...

But is it Simple?

Is it cool?

Is It Web 2.0?

Well Not Exactly...

So...

Tropo.com

```
answer();
say("Hello, world!");
hangup();
```

Tropo is Simple

ecmascript
the language of the web

(JavaScript)

Ruby

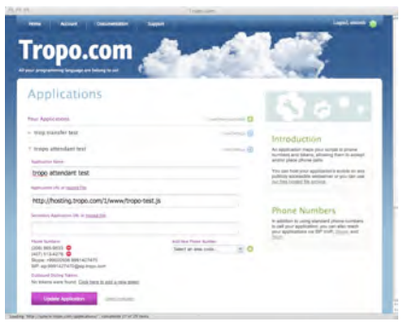
python

Telephony in **YOUR** Language (thanks to the magic of JSR223)

- answer
- redirect
- reject
- ask
- say
- record
- log
- wait
- default
- call
- transfer
- hangup

Simple to Learn

- Hosted service
- Accessible via Phone, SIP, Skype etc
- Inbound and Outbound calling
- Free for developers
- No setup costs
- Five minutes from sign-up to live deployment



+

= GO

Simple to Deploy

What are the Ingredients?

SIP Servlets (JSR289)

SIPMethod

SIP Servlets (JSR289)

Media Control (JSR309)

SIPMethod

Prophecy

SIP Servlets (JSR289)

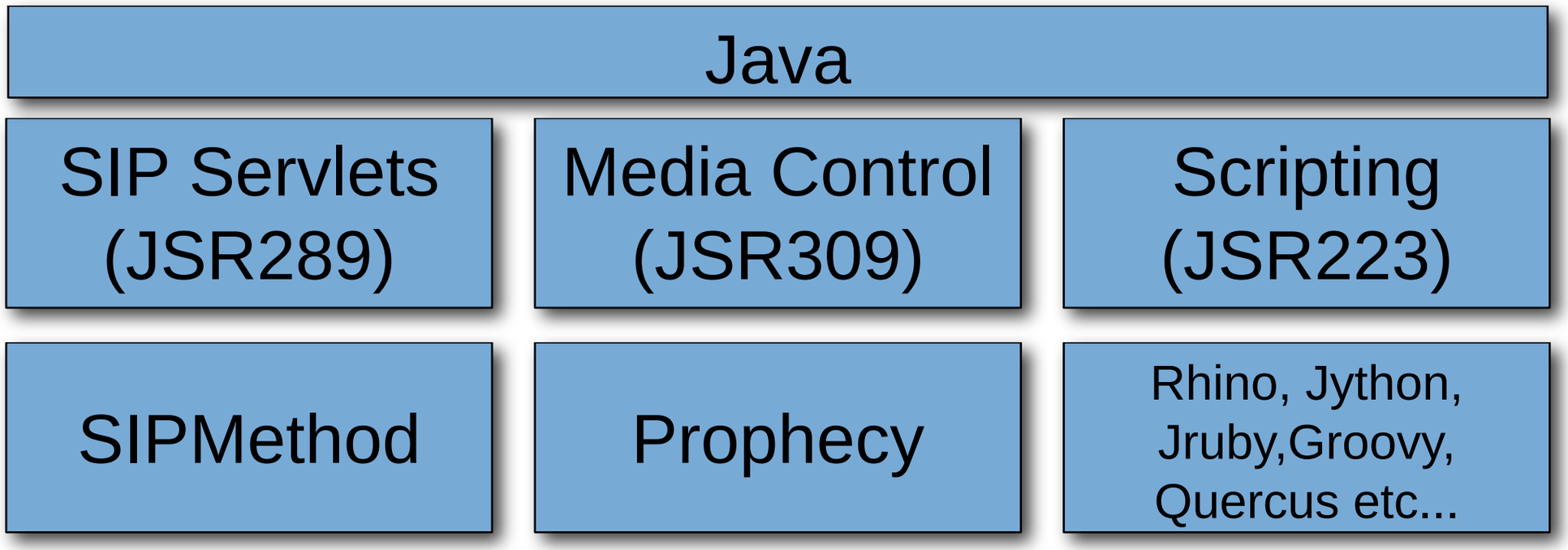
Media Control (JSR309)

Scripting (JSR223)

SIPMethod

Prophecy

Rhino, Jython, Jruby,Groovy, Quercus etc...


Java
SIP Servlets (JSR289)
Media Control (JSR309)
Scripting (JSR223)
SIPMethod
Prophecy
Rhino, Jython, Jruby,Groovy, Quercus etc...

Tropo.com

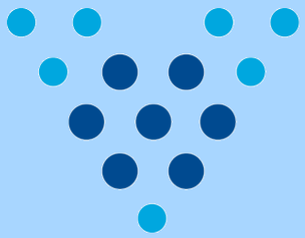
Java

SIP Servlets (JSR289)	Media Control (JSR309)	Scripting (JSR223)
SIPMethod	Prophecy	Rhino, Jython, Jruby,Groovy, Quercus etc...

Tropo.com

Java

SIP Servlets (JSR289)	Media Control (JSR309)	Scripting (JSR223)
SIPMethod	Prophecy	Rhino, Jython, Jruby,Groovy, Quercus etc...



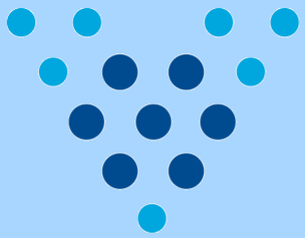
voxeo

Applications

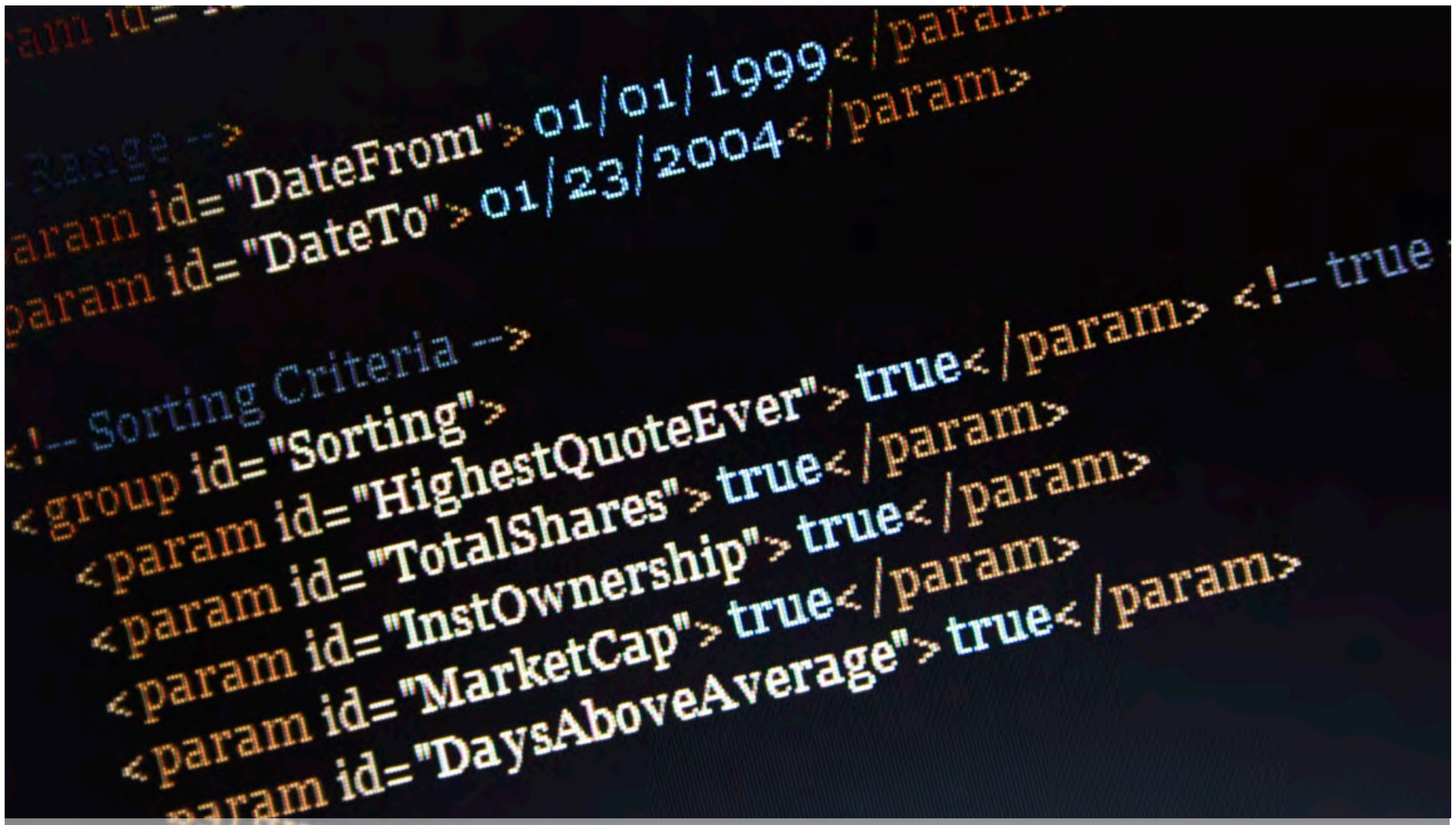
Tropo.com

Java

SIP Servlets (JSR289)	Media Control (JSR309)	Scripting (JSR223)
SIPMethod	Prophecy	Rhino, Jython, Jruby,Groovy, Quercus etc...



voxeo



How about some code?

T.1: Hello World

JavaScript and PHP

```
answer();
say("Hello, world!");
hangup();
```

Ruby

```
answer
say "Hello, world!”
hangup
```

Python

```
answer()
say("Hello, world !")
hangup()
```

Groovy

```
answer()
say 'Hello, world!'
hangup()
```

Asking for Input - JavaScript

```
// -----------
// asking for input
// -----------


answer();

result=ask( "Hi. For sales, press 1. For support, press 2.", {choices:"1, 2"} );

if (result.name=='choice')
{
 if (result.value=="1") { say( "sales is not available right now.") }
 if (result.value=="2") { say( "support is currently on the other line." ) }
}


hangup();
```

Using ASR - Python

```
# Using speech input instead of touch-tone

answer()

result = ask("Hi. For sales, say sales. For support, say support",
{'choices':"sales, support", 'repeat':3})

if (result.name == 'choice'):
    if (result.value == "sales"):
        say("Sales is not available right now")
    if (result.value == "support"):
        say("Support is currently on the other line.")

hangup()
```

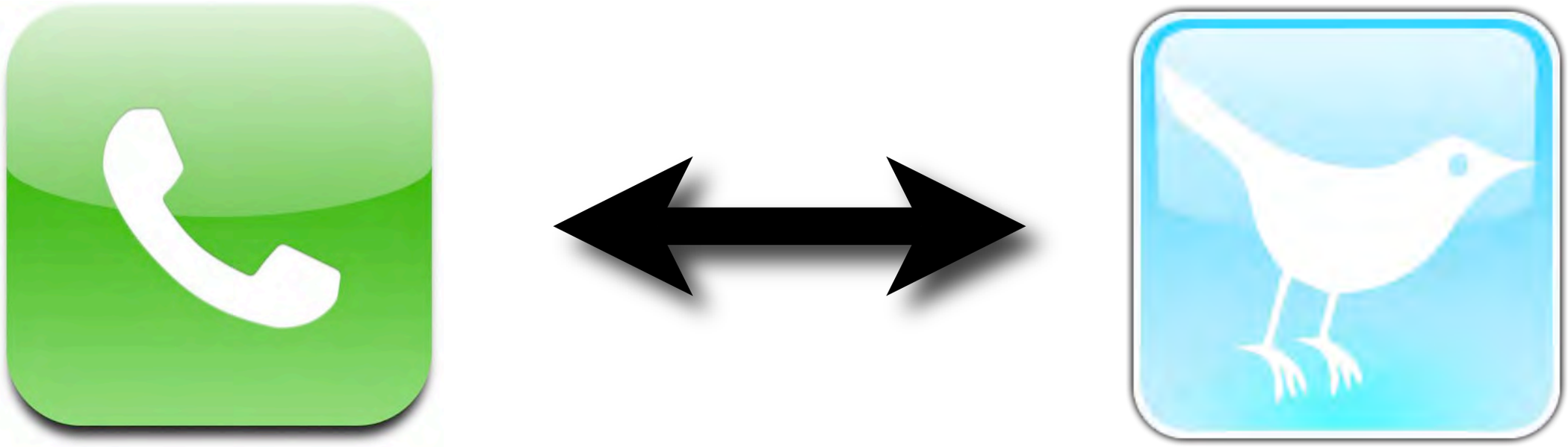
Using ASR and DTMF - JavaScript

```
answer();

result=ask( "For sales, just say sales or press 1. For support, say support or press 2.",
 { choices:"sales( 1, sales), support( 2, support)", repeat:3,
    onBadChoice: function() { say("I'm sorry, I didn't understand what you said.") }
 } );

if (result.name=='choice')
{
 if (result.value=="sales")
 {
     say( "Ok, let me transfer you to sales."       );
     transfer( "14075551111");
    }
 if (result.value=="support")
 {
     say( "Sure, let me get support.  Please hold." );
     transfer( "14085552222");
    }
}
```

How about a mashup?

Sample Application Overview

- Lets try the same application again in tropo...



Twitter Example - Groovy

```
answer()
if (currentCall.callerID == "8315551234") {
    def event = ask("Welcome RJ. Please enter the phone number you wish",
                    [choices:"[10 DIGITS]",timeout:10])
    if (event.name=="choice") {
        transfer(event.value)
    } else {
        say ("too slow bro.")
        hangup();
    }
} else {
  say("Thank you for calling RJ. We will connect you now.")
  say("His twitter status is")
  String xml =
 "http://twitter.com/statuses/user_timeline.xml?screen_name=zscgeek".toURL().text
  def strXML = new XmlParser().parseText(xml)
  say(strXML.statuses.status[0].text.text())
  transfer("tel:+18315551234")
}
```


CHERRY PIE
INGREDIENTS:
2 1/2 tablespoons quick cooking tapioca
1cup sugar
1/8 teaspoon salt
2 cans (1 pound each) water packed pitted red sour cherries, drained
1/2 cup cherry juice
6 drops red food coloring
1/4 teaspoon almond extract
Pastry for 2 cruet 9 inch pie, unbaked
1 tablespoon butter
DIRECTIONS:

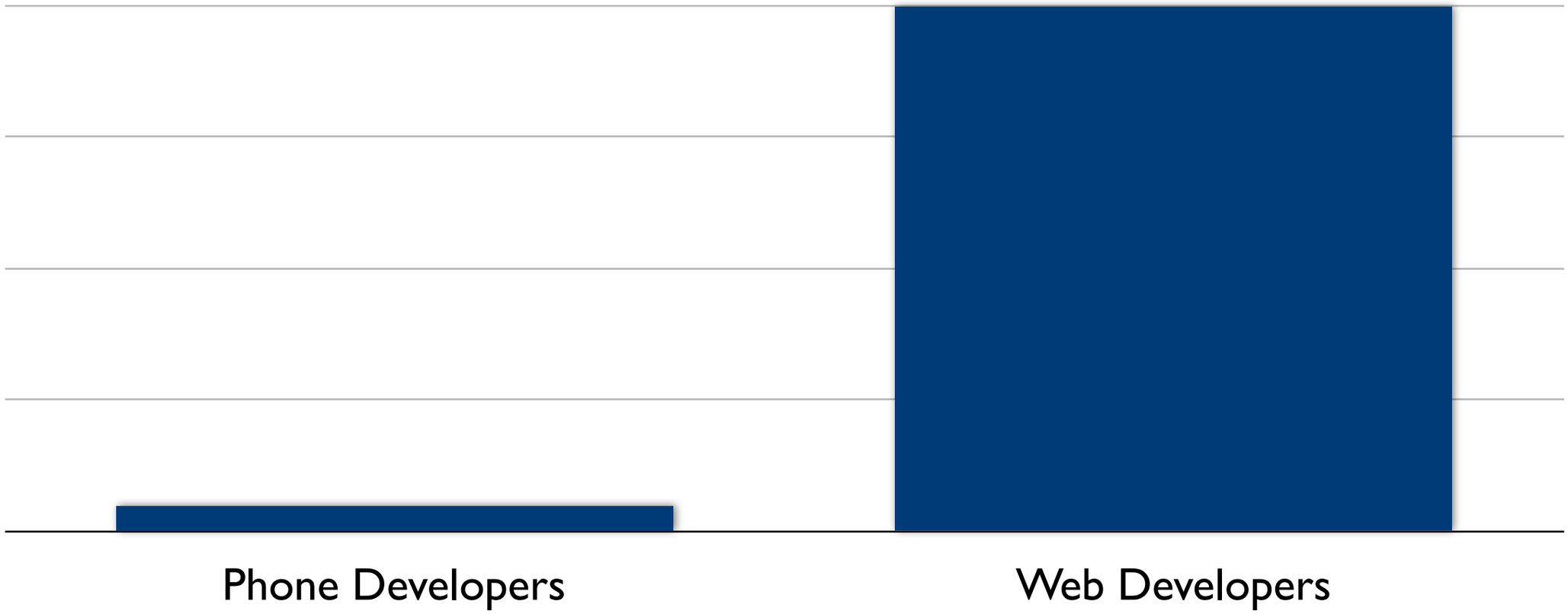
Easy as Pie!

So...

Wrapping Up

So why is this important?

Quick Poll: Are you a phone developer?



Fact of Life

Web

Innovation

Don’t create phone applications...

Instead create cool applications that use the phone...

Vooices.us – Control Games, App... | Tropo.com | Voxeo Corporation: IVR Made Easy

Home | Developers Look Here | The Vooice Company | Contact Us | Vooices Blog

VOOICES

Home | Thats Cool, What Else | Game Gallery | Score-Boards | Developer Account | Player Account

Try our Sample Control Demo **Wordz** by clicking the game picture above, dialing the given number and then entering the keycode in the window when asked to do so.

Phone Controlled, Realtime, Multiplayer Gaming

Vooices is a Phone Controlled, RealTime, MultiPlayer Gaming Platform for Screens in Public Spaces or for Games on the internet. What we are saying is that you can control a game or application with your voice or keypad by simply dialling the game.

You don't have to install anything on your phone to play a Vooices Game. It works on any phone, on any carrier or network, in any country of the world.

So what does this mean? Basically anyone who can code in Javascript, Flash or Silverlight can hook up a Vooice input system to their game providing cutting edge input features for Free. Try our sample game Words, Its all Javascript powered by your voice.

Vooices updates the in-game session state on screens or on web browsers using its own custom technology, and can send audio, sms messages, pictures, wallpapers, ringtones, etc back to each user playing the game. Vooices uses this to deliver out of call media advertising.

Vooices can be used to control anything from games to real time applications such as maps.


Thank you

Love me? Hate me?
Then say what you want about me...

RJ Auburn

rj@voxeo.com

http://www.voxeo.com/free

Tropo.com Script Based Telephony

MICROMETHOD Java SIP Application Server

voxeo XML Telephony