SIV for VoiceXML 3.0: Language and Application Design Considerations

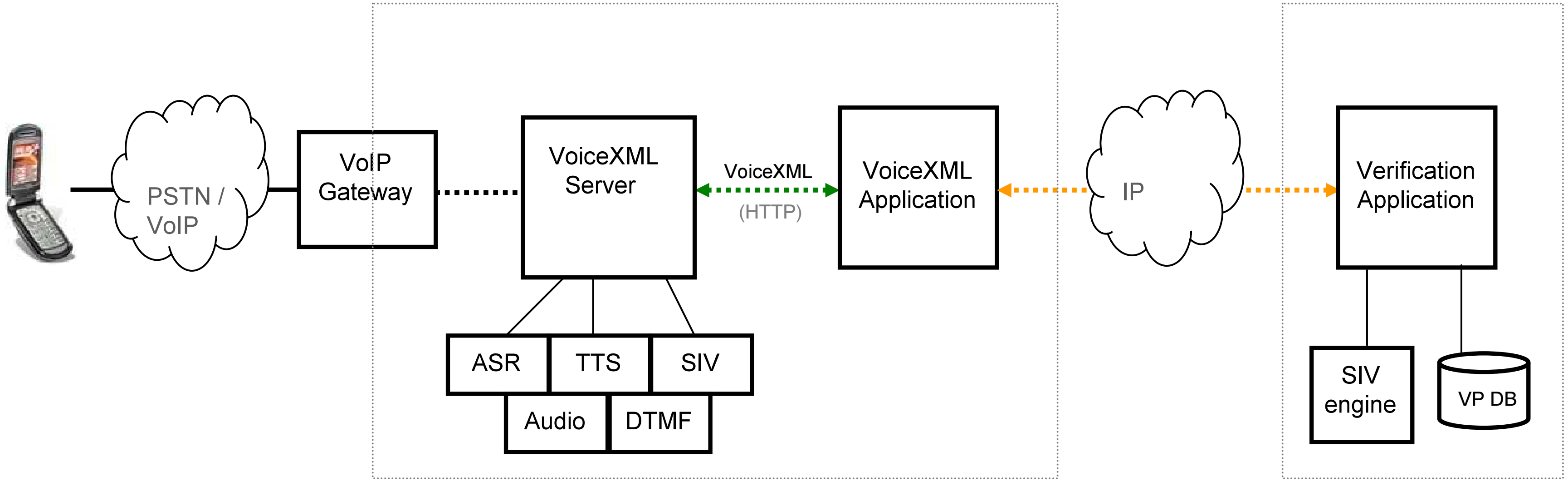
Ken Rehor

Cisco Systems, Inc.

krehor@cisco.com

March 05, 2009

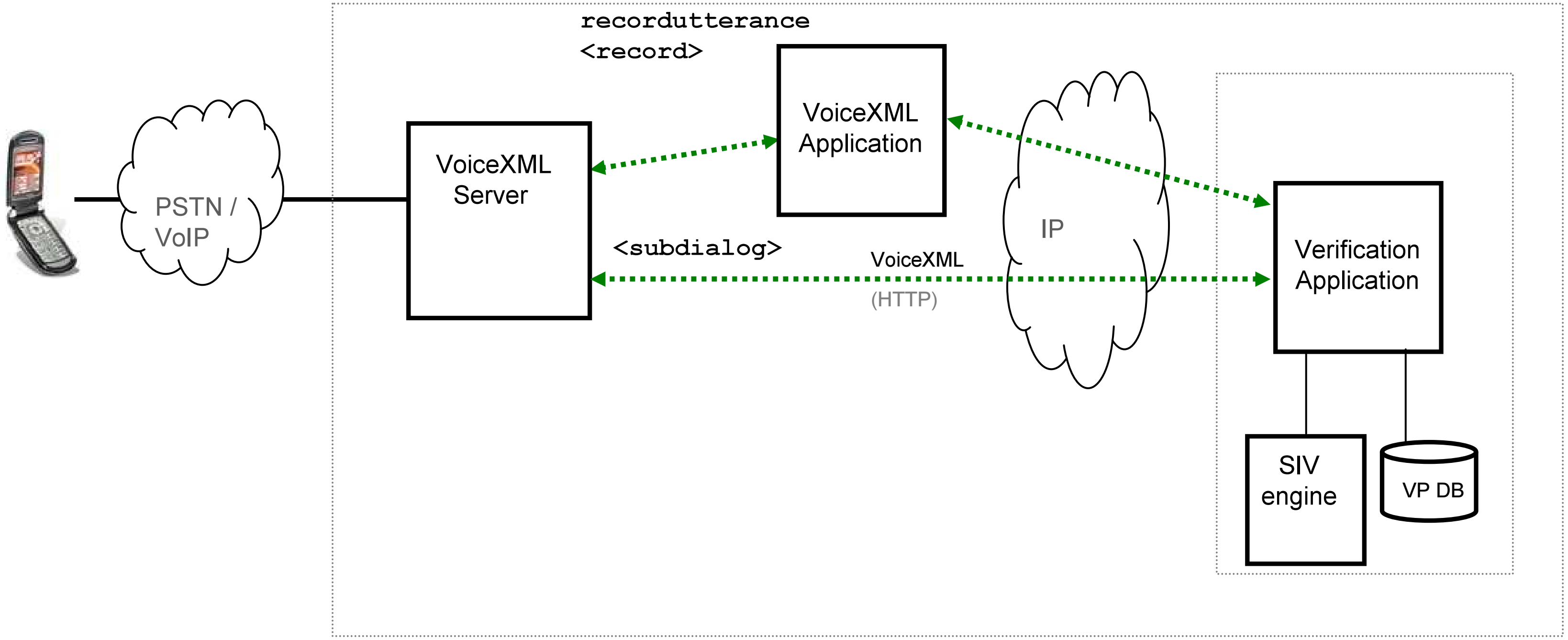
VoiceXML Application Architecture



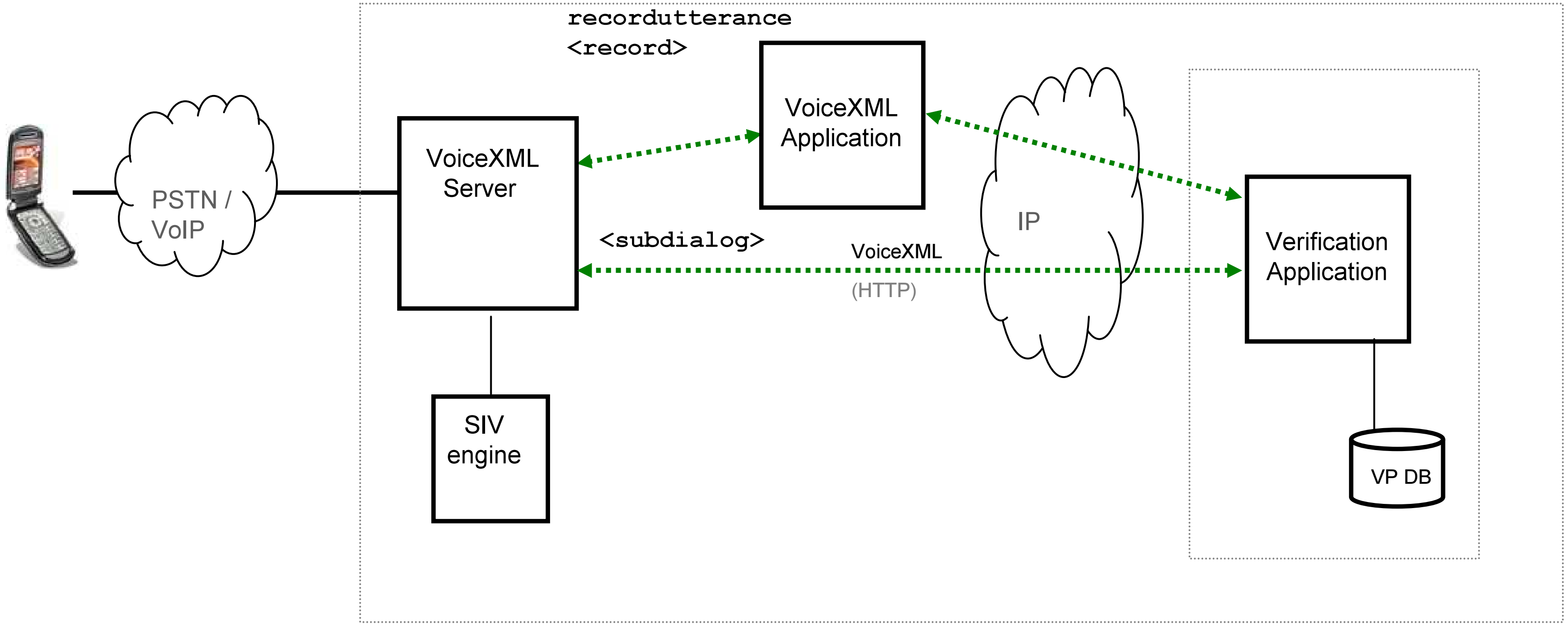
SIV in VoiceXML 2.x

- Server-side SIV processing
  - <record>
  - <field> with recordutterance
- Language extensions
  - Nuance "voiceprint forms"
  - BeVocal

VoiceXML 2.x SIV Integration



VoiceXML 2.x SIV Integration



Standard VoiceXML prompt/field model

- Text-independent
  - <prompt> / <record>
  - Submit recording to application server

- Text-dependent, Text-prompted
  - <prompt> / <field> (with recordutterance)
  - Submit utterance recording to application server

VoiceXML 2.x <record>

```
<form name="verify">

<!-- could use external grammar -->
   <record name="utterance" maxtime="5s
     <prompt> Say this digit sequence: one two three four five.</prompt>
     <noinput> I didn't hear anything, please try again. </noinput>
   </record>

   <block>
      <submit next="check_utterance.pl" enctype="multipart/form-data"
      method="post" namelist="utterance"/>
   </block>

</form>
```

VoiceXML 2.1 `<field>`

```
<form name="verify">



   <prompt>Say this digit sequence: one two three four five.</prompt>

   <field type="digits">
            <filled>
                <!-- if spoken digits match expected response,
                  then process voice model -->
            </filled>
     </field>

</form>
```

VoiceXML 2.1 `<field>` with `recordutterance`

```
<form name="verify">

<property name="recordutterance" value="true"/>

   <prompt>Say this digit sequence: one two three four five.</prompt>

   <field type="digits">
            <filled>
                <!-- if spoken digits match expected response,
                  then process voice model -->
            </filled>
     </field>

</form>
```

Security Concerns

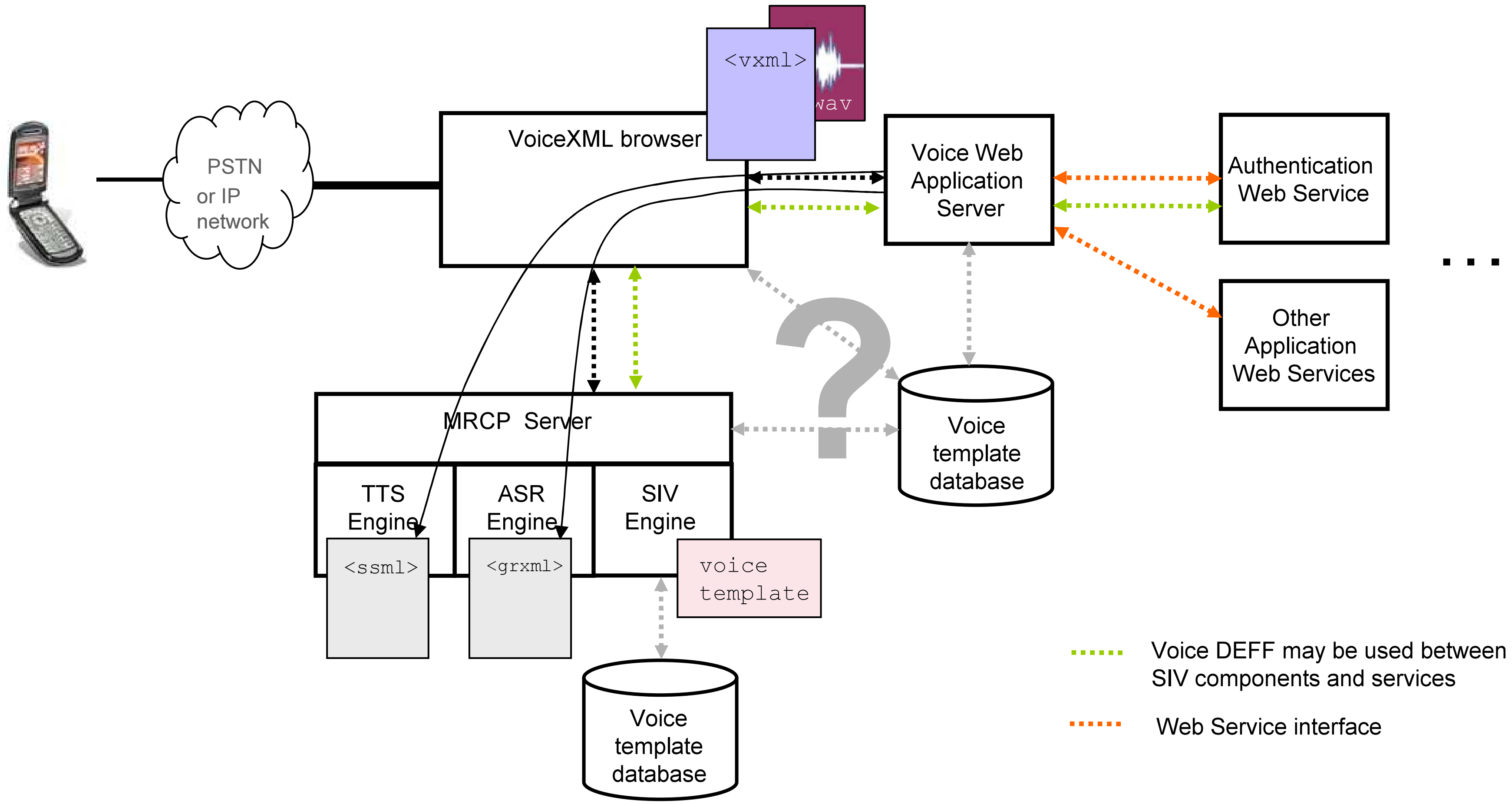
Architecture / Security / Trust

- One architecture may not be suitable for every use case

➔ Some architectures may not support the level of (dis)trust required for a particular deployment

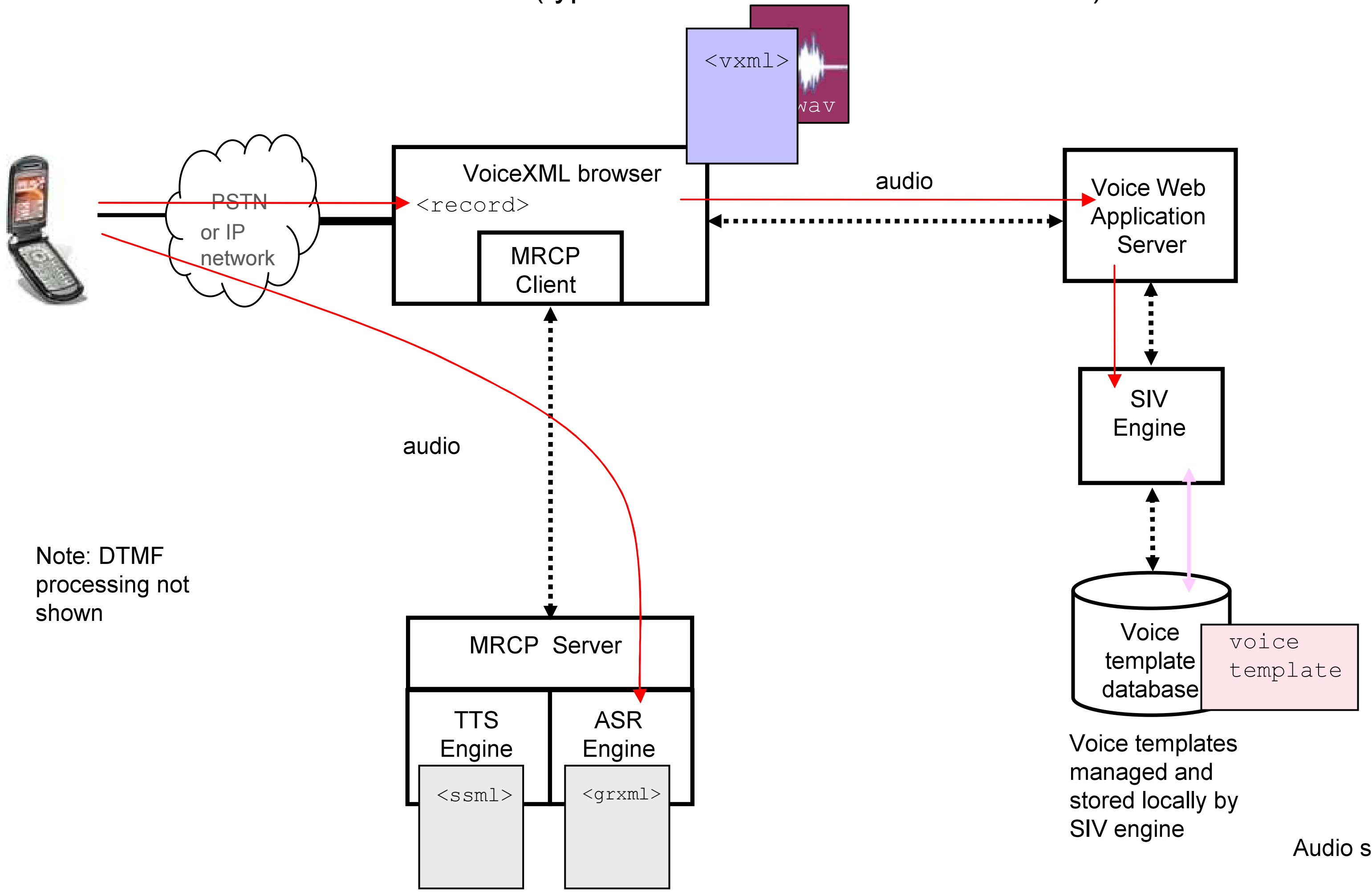
Security, Trust and Protocol Considerations in Distributed Voice Web Applications

Architecture options carry security implications



SIV engine and database managed by App server

VoiceXML browser records the utterance and forwards to app server
(typical scenario for VoiceXML 2.0/2.1)



<vxml>

wav

PSTN
or IP
network

VoiceXML browser

<record>

MRCP
Client

audio

Voice Web
Application
Server

SIV
Engine

audio

MRCP Server

TTS
Engine

<ssml>

ASR
Engine

<grxml>

Voice
template
database

voice
template

Note: DTMF
processing not
shown

Voice templates
managed and
stored locally by
SIV engine

Audio stream vs. buffers

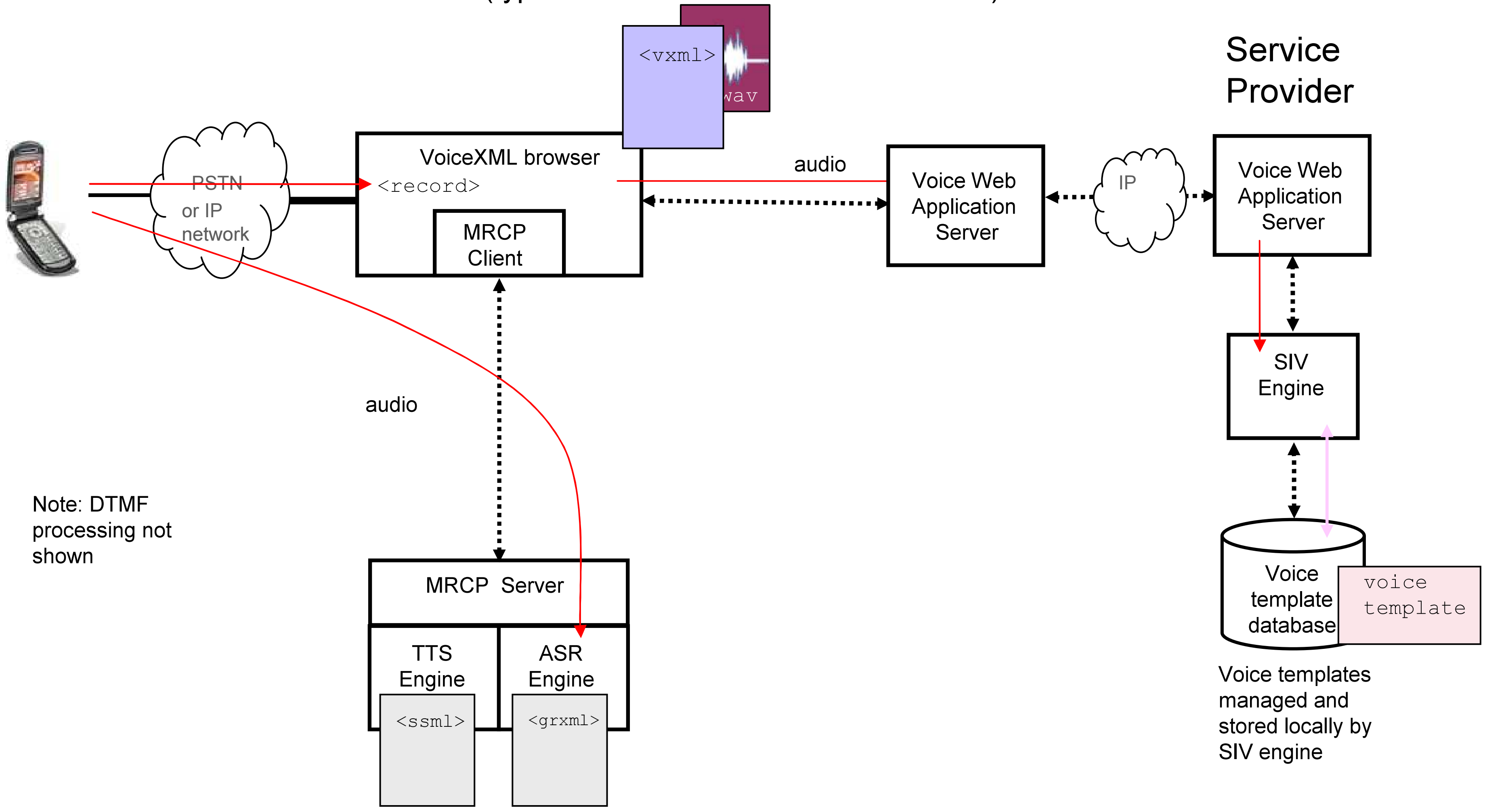
Streaming handled by RTP?

Buffers may be handled by
audio recorder function. Part
of browser or MRCP engine?

SIV engine and database managed by App server

VoiceXML browser records the utterance and forwards to app server

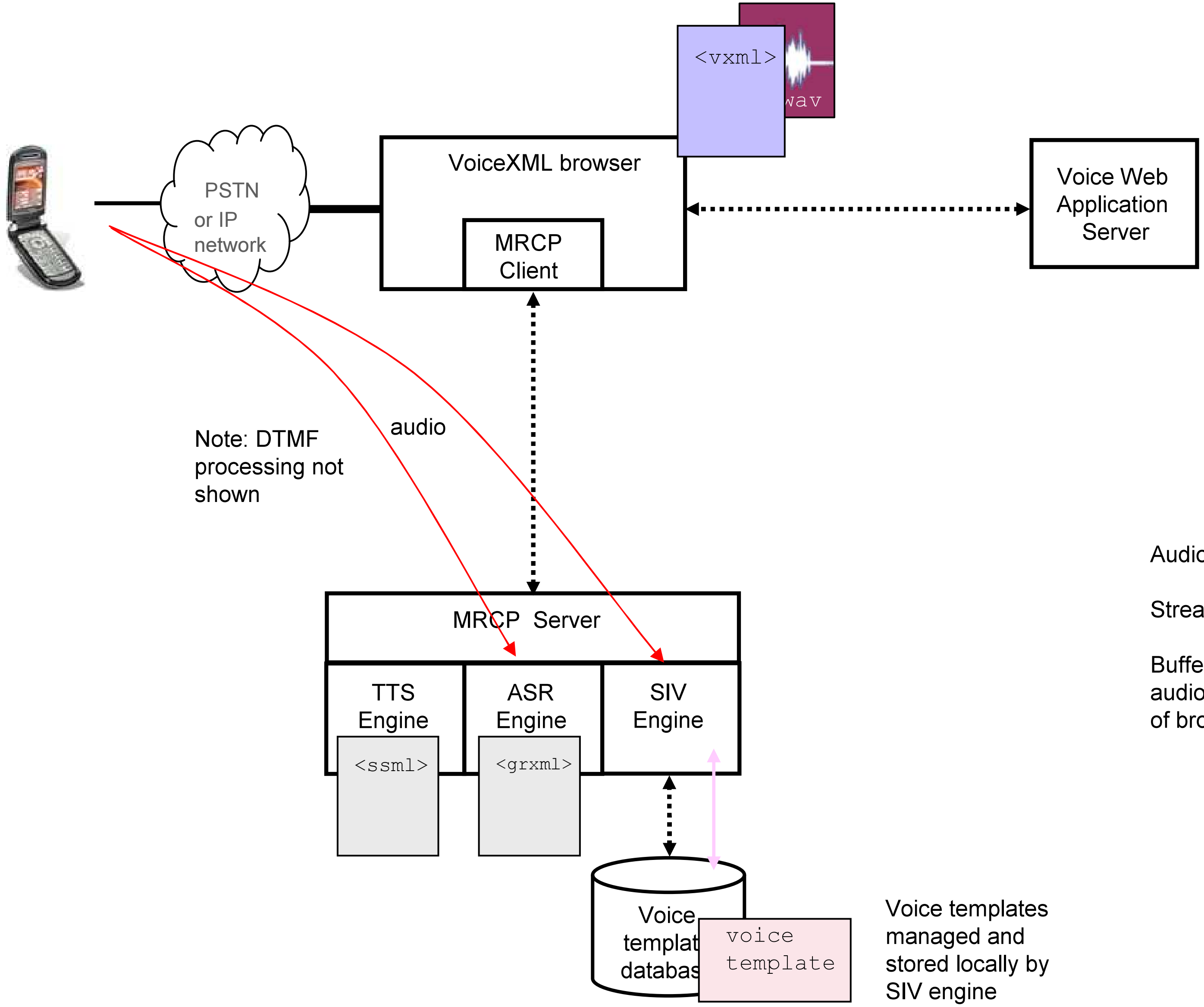
(typical scenario for VoiceXML 2.0/2.1)



Note: DTMF processing not shown

Voice templates managed and stored locally by SIV engine

SIV engine and database managed by MRCP server



Note: DTMF processing not shown

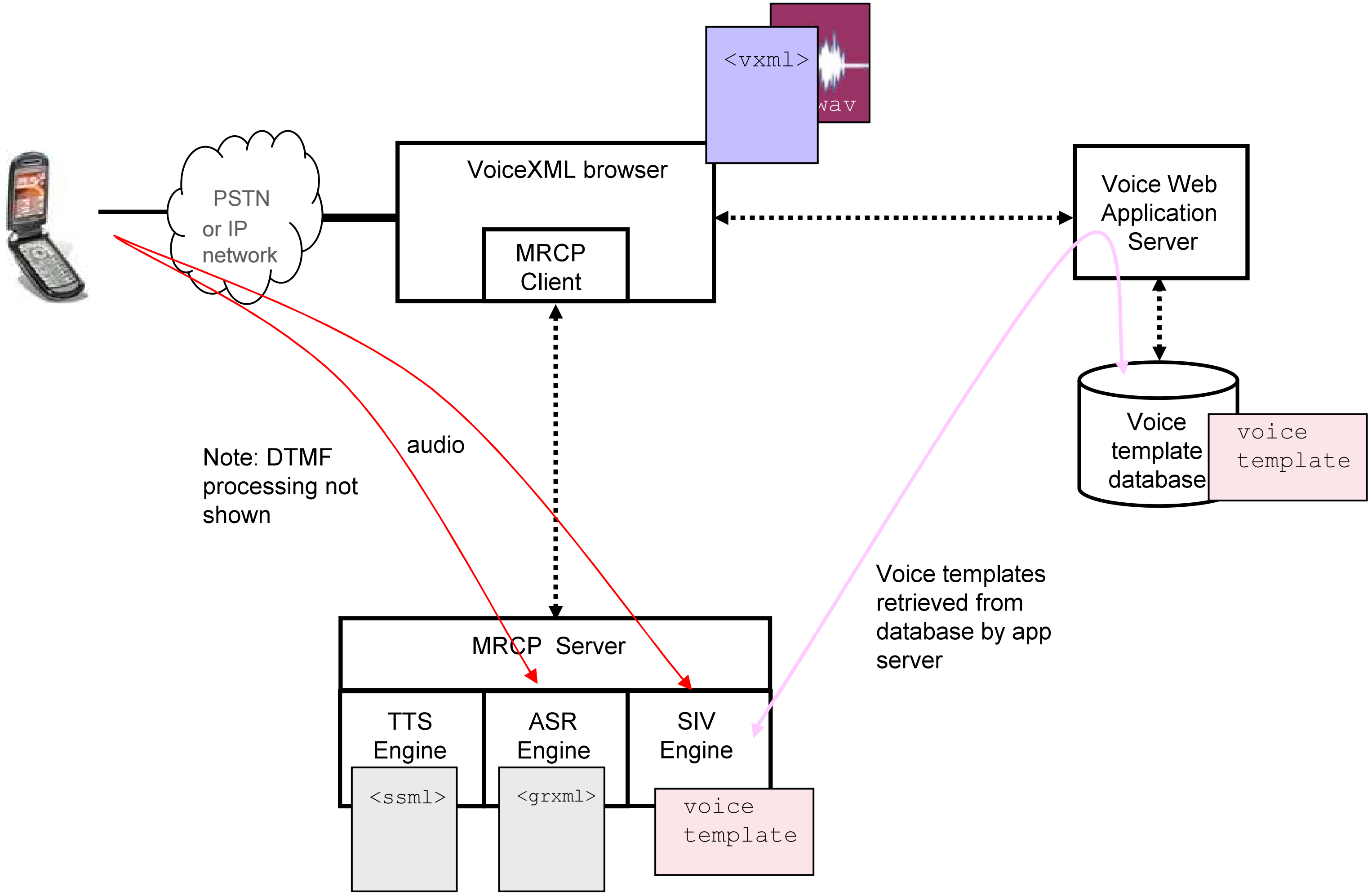
Voice templates managed and stored locally by SIV engine

Audio stream vs. buffers

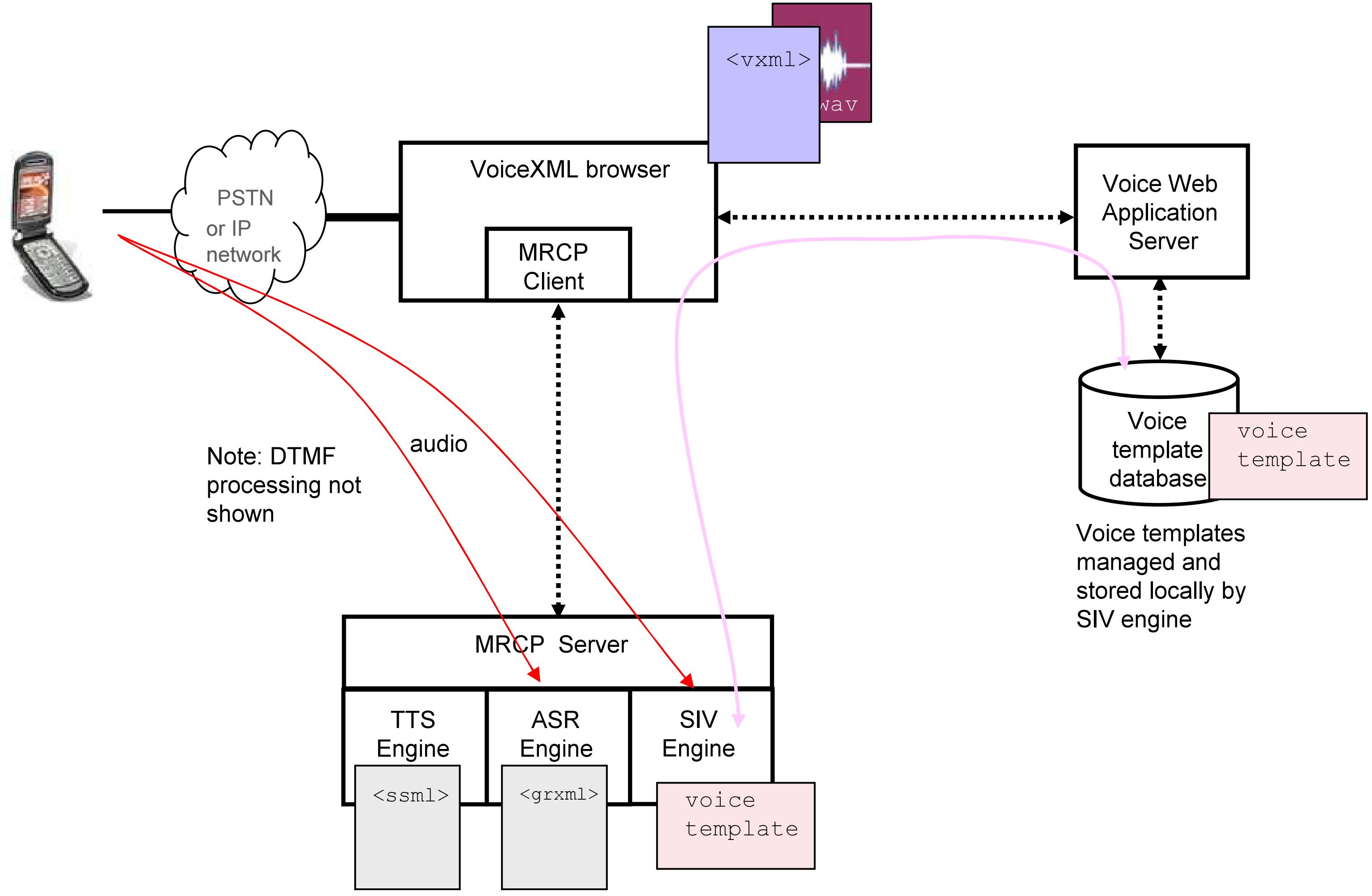
Streaming handled by RTP?

Buffers may be handled by audio recorder function. Part of browser or MRCP engine?

SIV engine managed by MRCP server
SIV database managed by app server
Voice model transmission managed by engine or MRCP Server



SIV engine managed by MRCP server
SIV database managed by app server
Voice model transmission managed by VoiceXML browser



PSTN or IP network

VoiceXML browser

MRCP Client

<vxml>

wav

Voice Web Application Server

Voice template database

voice template

audio

Note: DTMF processing not shown

MRCP Server

TTS Engine

ASR Engine

SIV Engine

<ssml>

<grxml>

voice template

Voice templates managed and stored locally by SIV engine

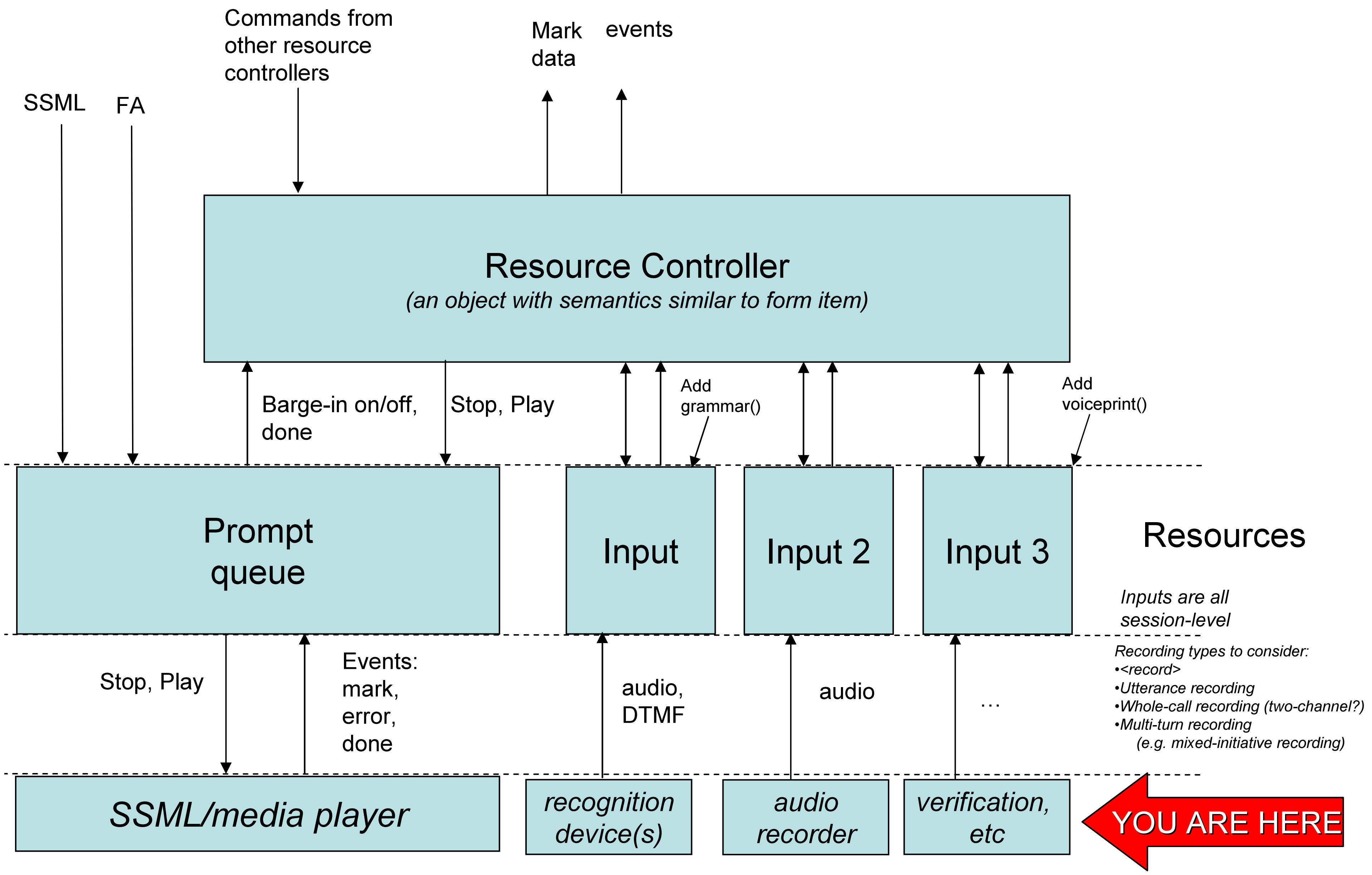
Voice templates retrieved from database by ap server

SIV in VoiceXML 3.0

V3 Integration Requirements

- Control multiple Input Resources
  - ASR and biometric engines
  - Simultaneously
  - Switch on a per <field> or verification basis
- Consistent with V3 overall design goals
- Simplify integration, yet provide sufficient control

V3 Data, Event relationship between components



Commands from other resource controllers

Mark data

events

SSML

FA

Resource Controller

(an object with semantics similar to form item)

Barge-in on/off, done

Stop, Play

Add grammar()

Add voiceprint()

Prompt queue

Input

Input 2

Input 3

Resources

Inputs are all session-level

Recording types to consider:

- *<record>*
- *Utterance recording*
- *Whole-call recording (two-channel?)*
- *Multi-turn recording (e.g. mixed-initiative recording)*

Stop, Play

Events: mark, error, done

audio, DTMF

audio

…

SSML/media player

recognition device(s)

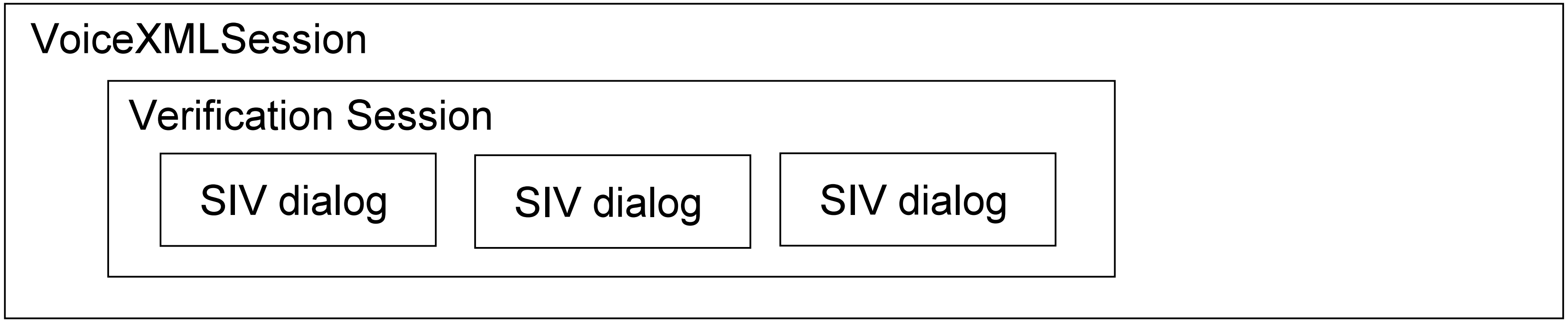
audio recorder

verification, etc

YOU ARE HERE

SIV "Session"

- Enrollment Session or Verification Session
- Verification process: *Uninterrupted process over several dialog states (having a Session-ID) where the results of each utterance are cumulated*



Define Data Model

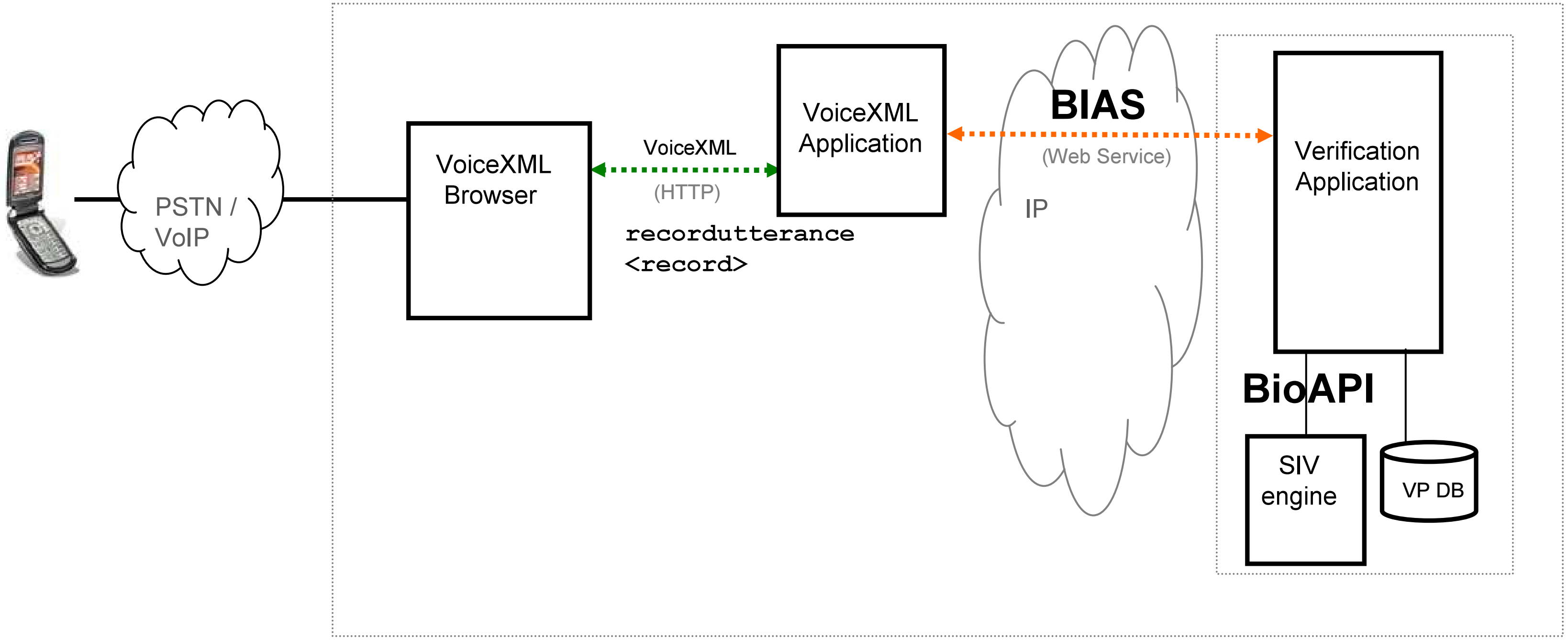
- Data passed to SIV engine
  - Environment
  - Properties
  - Attributes
  - Voice models
- Data returned from SIV engine
  - Results specified as an EMMA result
  - Errors/info
- Data used within SIV session
- Associate SIV result with ASR result

Define event model

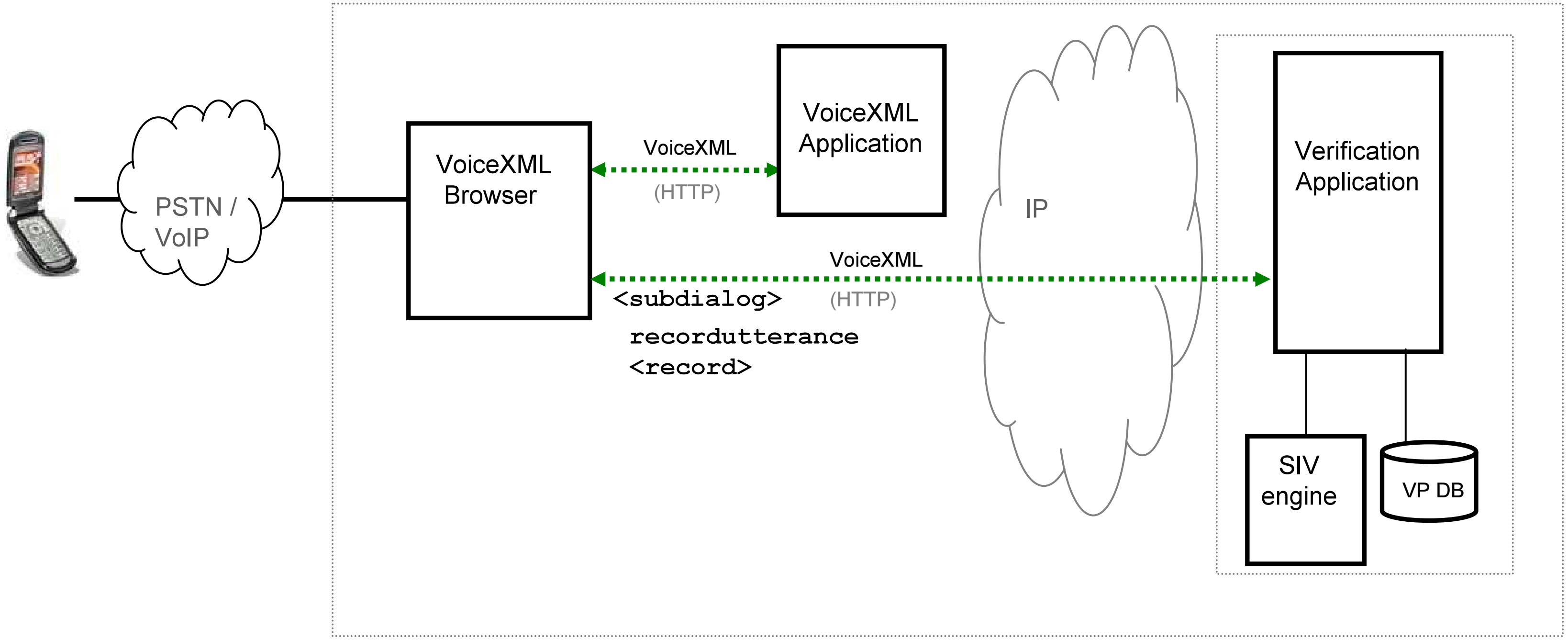
- Combine references from:
  - VoiceXML Forum
  - MRCP v2
  - Engine vendors

VoiceXML and SIV Web Services

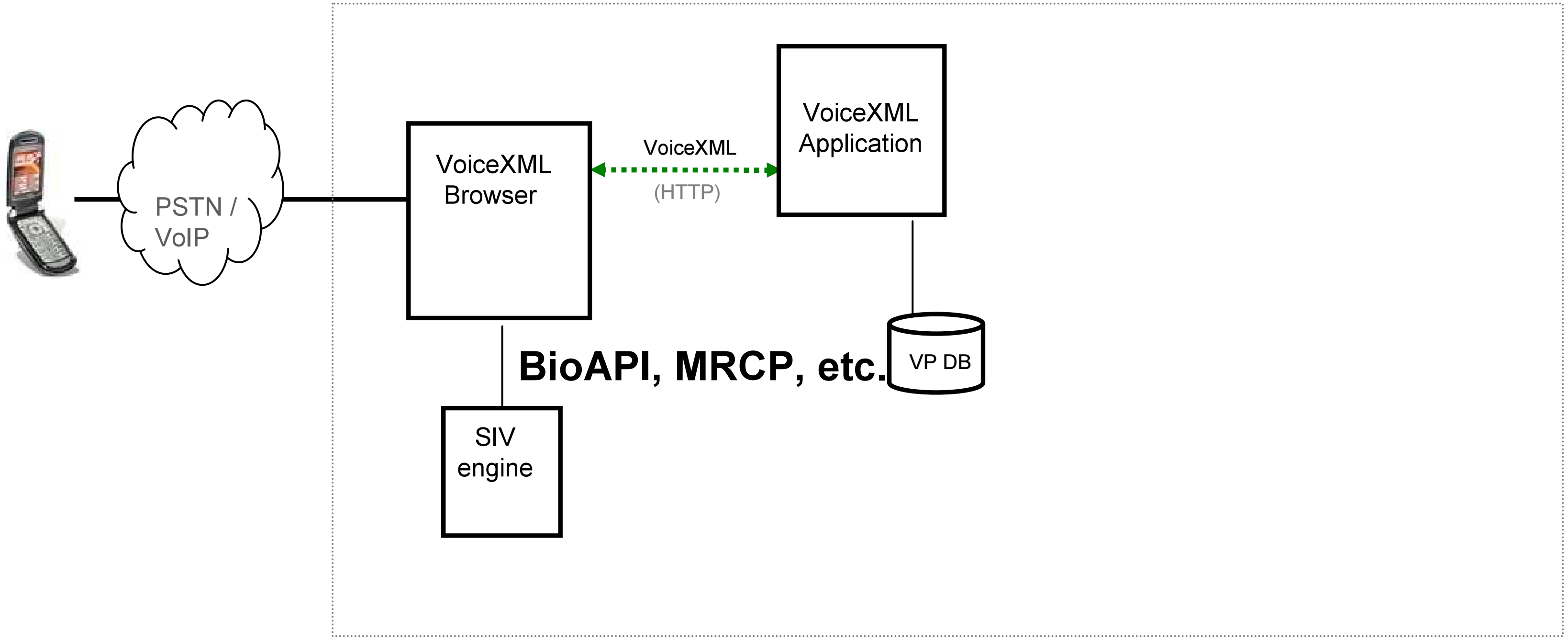
VoiceXML 2.x/3.x SIV Integration via BIAS web service



VoiceXML 2.x/3.x SIV Integration via <subdialog>

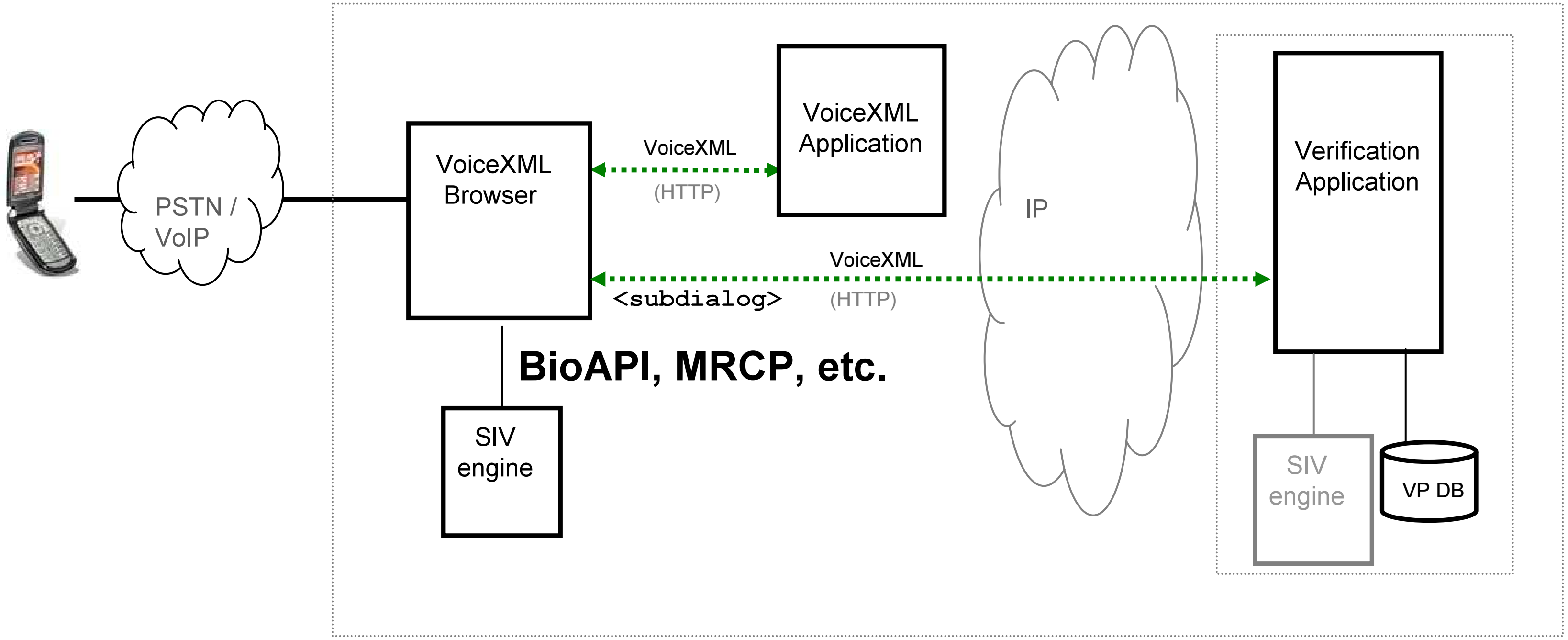


VoiceXML 3.0 SIV Integration



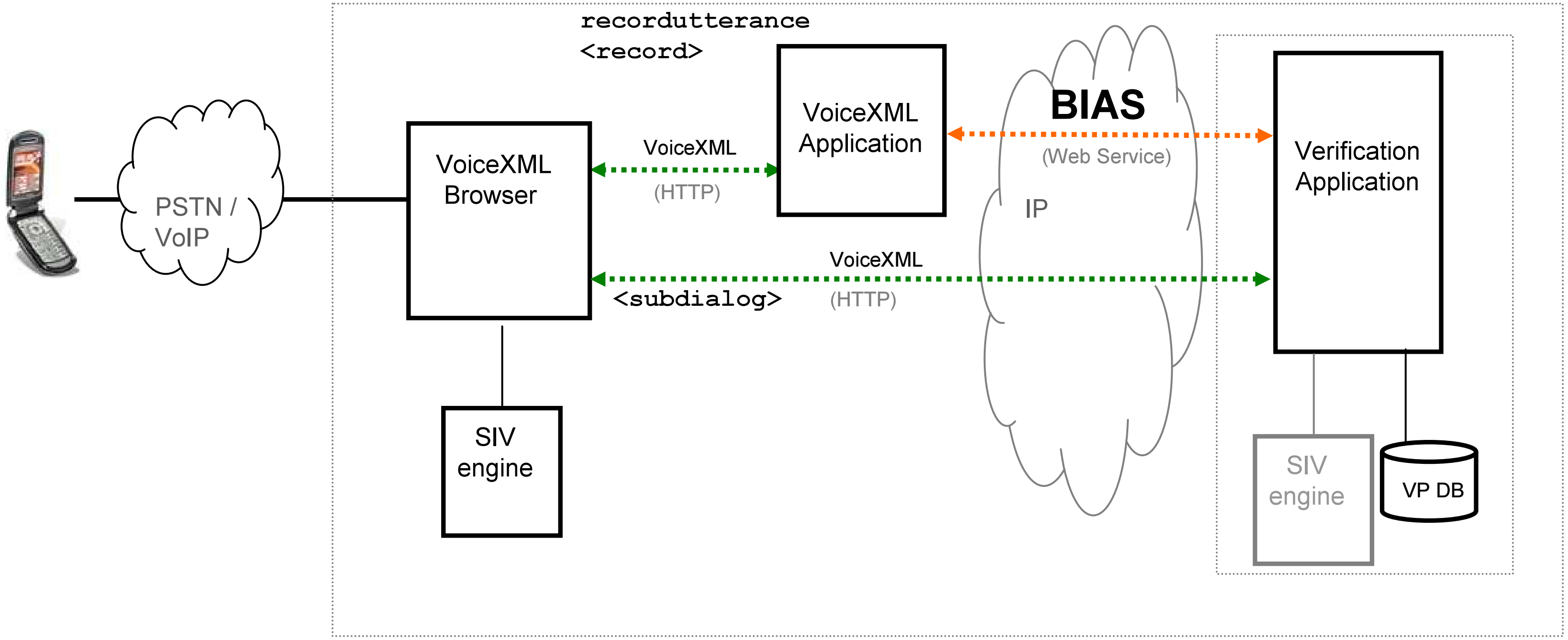
- V3 SIV native language features
- Browser/Engine integration via BioAPI, MRCP, proprietary API, etc.

VoiceXML 3.0 SIV Integration

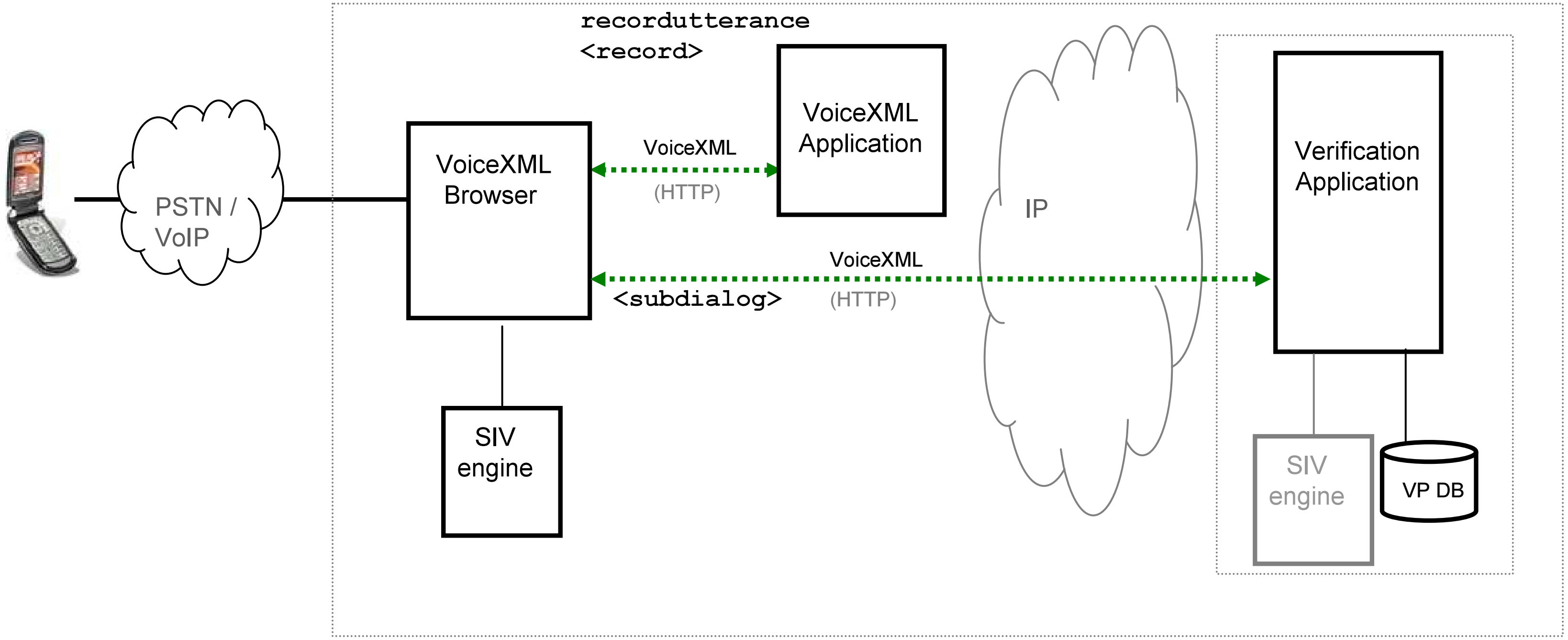


- V3 SIV native language features
- Browser/Engine integration via BioAPI, MRCP, proprietary API, etc.

VoiceXML SIV Integration via BIAS web service or `<subdialog>`



VoiceXML Application Switching



Pros and Cons of Native V3 SIV functions

V3 SIV Native Functions: Pros and Cons

SIV engines controlled directly by VoiceXML

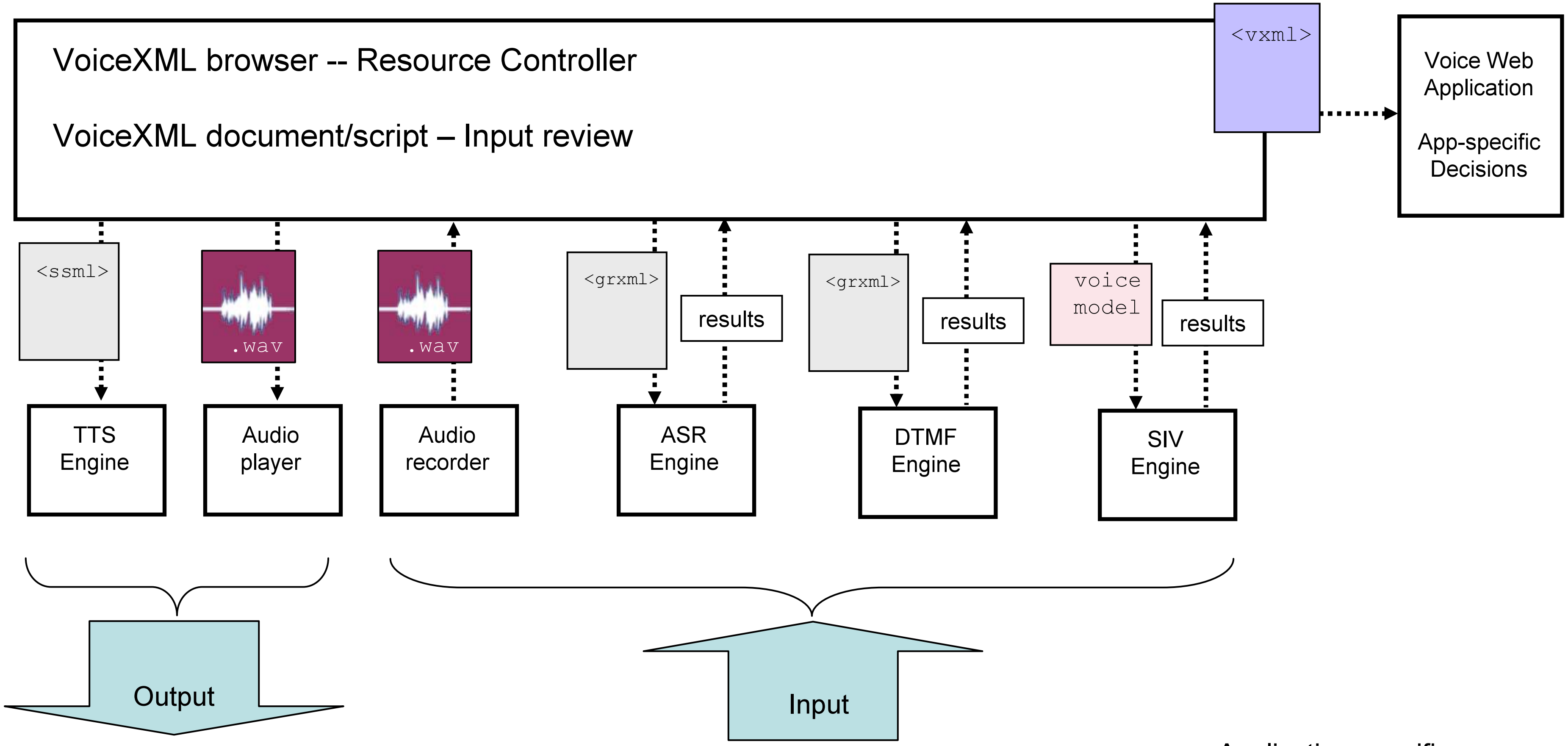
Pros

- V3 Requirement: simultaneous processing
  - More than one input engine, e.g. ASR+Verify
- Performance
  - Speed/responsiveness
  - Accuracy
- Consistency of resource control
  - Aligned with other input resources
- Benefit to app developers
  - Don't have to buy, install, maintain SIV engine
  - Shared resource on VXML platform
- Benefit to Platform Vendors / Service Providers
  - Shared resource
  - Ease of deployment
  - Enhanced service offering

Cons

- Not available today; need an interim solution
- App concerns
  - Enables developers to do 'bad' things
- Security concerns
  - Enables developers to do 'bad' things
- Full portability still a long way off
  - Voice models, engine capabilities, results/errors, etc. are all proprietary
- Platform integration not standard yet
  - MRCPv2 not sufficient; need more features (MRCPv3?)

Resource Control and Distributed Decision Making



- ASR
  - audio quality
  - confidence/threshold or nomatch
  - result: word or phrase
- DTMF
  - result: digit string or no match

Application-specific Decisions
- user selection
- authentication