

O₂S Planning Technology

Day 1, Session 1, Part 4
July 2006
Quanta JustPlay Seminar

Justin Mazzola Paluska
jmp@mit.edu

Overview

- System Model
 - JustPlay Music Demo
- Model Details
 - Extended Examples

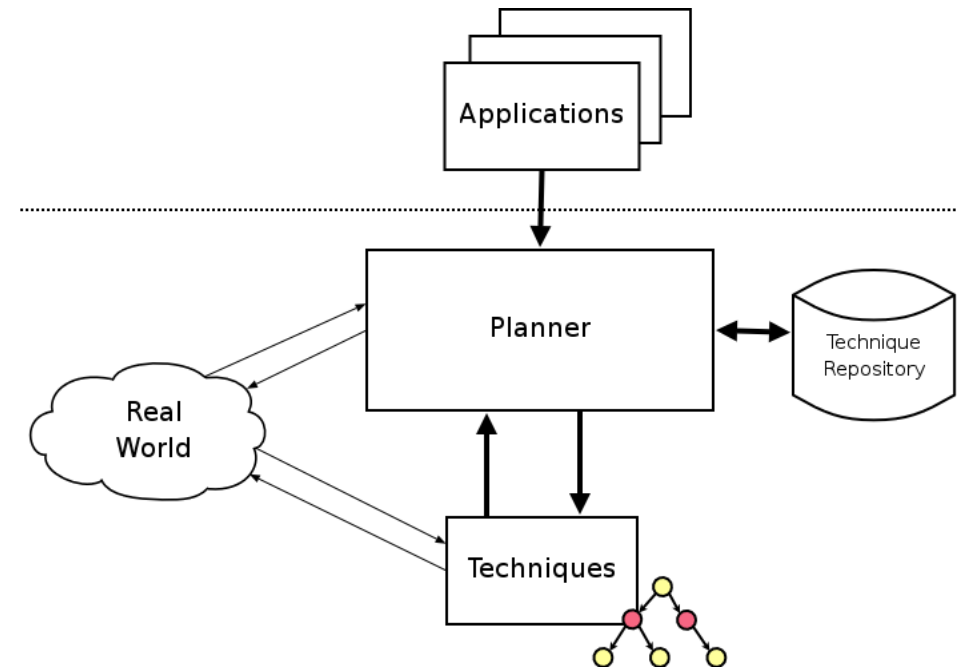
Approach

- Make application adaptation points explicit
 - Provide mechanism to choose, startup, teardown, and swap alternatives.
- Evaluation-based
 - Very little analysis
 - Exploit contexts or cached results to “learn” what should be done
- Reduce ad-hoc decision making
 - How well can you make do with what you have?

System Model

Planner Concepts

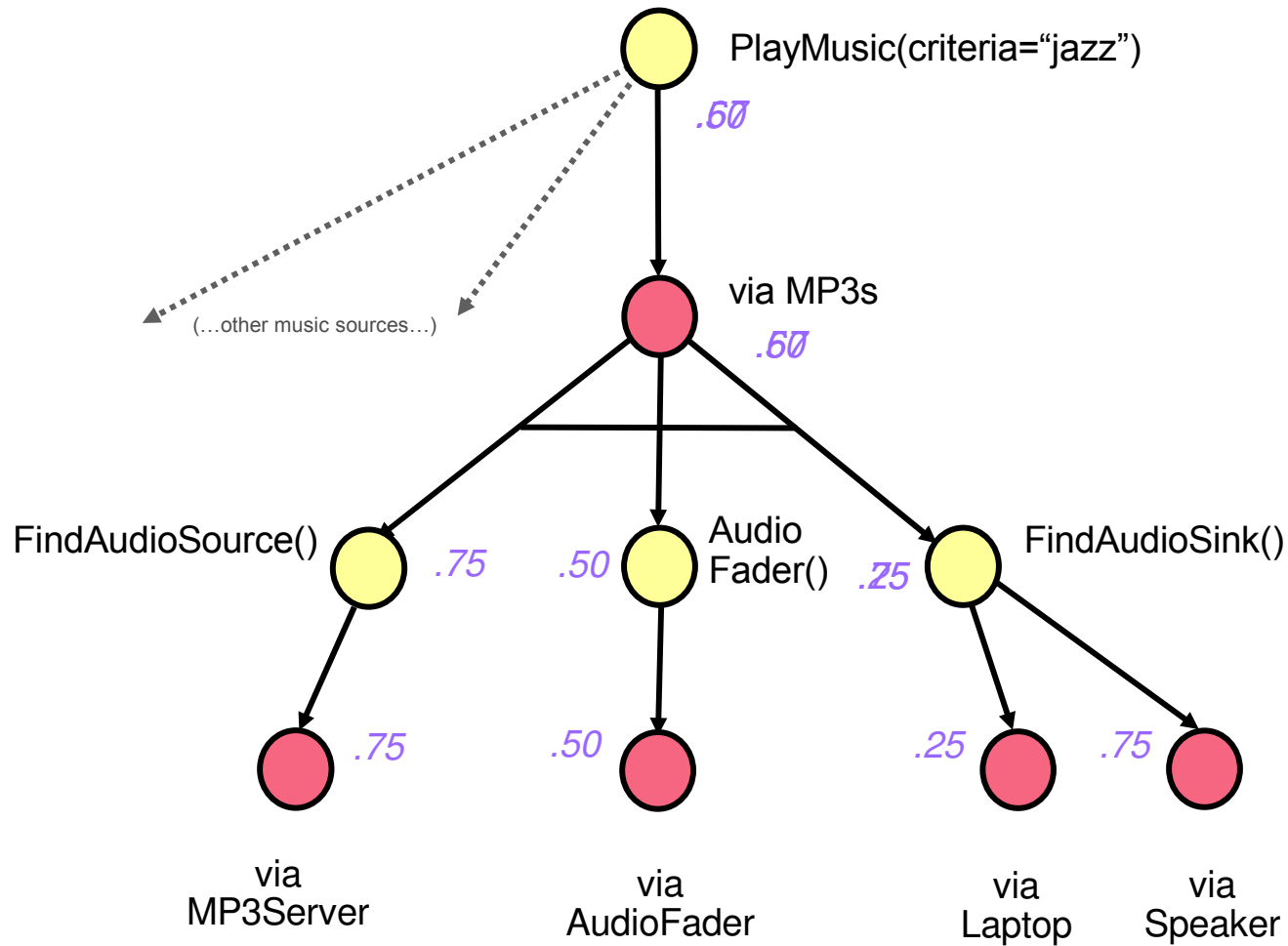
- Goal
 - parameterized intent or representation of a desired condition
- Technique
 - code that implements a Goal
- Planner
 - runtime system that matches Goals to Techniques



Planner

- Recursively bind Techniques to Goals
 - AND/“OR” tree
- At each Goal (OR) node, choose a node for execution.
 - based on the satisfaction property
- Execute the chosen nodes
- Maintain the tree over time
 - Re-execute relevant stages
 - Propagate property updates

Planner



Model Details

Goals

- parameterized specification of a class of intents or desires
 - input parameters
 - types and names of returned Properties
- Goal names the specification used.

```
<?xml version="1.0" standalone="yes" ?>
<goal>
  <documentation>
    <brief_description>
      A <this /> connects two users so that they may
      communicate.
    </brief_description>
    <long_description>
      A <this /> connects two users so that they may
      communicate. The initiator of the conversation may set an
      "importance" level for the teleconference, which may be
      used by the
      receiving end to determine if the conversation should be
      accepted and
      the like.
    </long_description>
  </documentation>
  <parameters>
    <parameter>
      <name>person1</name>
      <type>http://o2s.lcs.mit.edu/specs/rtypes/Person.xml</type>
      <documentation>
        <brief_description>
          The initiating end of the teleconference.
        </brief_description>
      </documentation>
    </parameter>
    <parameter>
      <name>person2</name>
      <type>http://o2s.lcs.mit.edu/specs/rtypes/Person.xml</type>
      <documentation>
        <brief_description>
          The other end of the teleconference.
        </brief_description>
      </documentation>
    </parameter>
  </parameters>
</goal>
```

Goal Instances

- Goal bound with Parameters
- Return value of “call” is a Solution
 - Properties mirror that of the chosen Technique
- A Goal Instance persists until it is told to shutdown

Techniques

- Parameterized template for satisfying a particular Goal
 - Both declarative and imperative code
 - May have subgoals
- Technique Instances satisfy Goal Instances
 - parameters bound to Goal Instance's
 - fill in Properties of Solutions

Technique Code

- Partitioned into Stages
 - write stage N+1 as though stage N has completed
- All state is stored in Solution objects
 - subgoal properties
 - one's own properties

```
to PlayMusic(criteria=any):  
  via MP3s:  
  
  subgoals:  
    source = FindMusicStream(goal.criteria)  
    sink = FindAudioSink()  
    fader = AudioFader()  
  
  eval:  
    solution.satisfaction =  
      (source.satisfaction +  
       sink.satisfaction) / 2  
  
  exec:  
    fader.resource.connect(source.out,  
                           sink.in)  
  
  update sink from old:  
    fader.replace(old, sink.in)
```

A Few Examples

PlayMusic via MP3 Technique

```
to PlayMusic(criteria):
```

```
    via MP3s:
```

```
    subgoals:
```

```
        source = FindMusicStream(criteria=goal.criteria,  
                                   stream_format="MP3")  
        sink = FindAudioSink(stream_format="MP3")  
        fader = AudioFader(in_stream_format="MP3",  
                             out_stream_format="MP3")
```

```
    ...
```

PlayMusic via GenericStream

```
to PlayMusic(criteria):
```

via GenericStream:

subgoals:

```
source = FindMusicStream(criteria=goal.criteria)
```

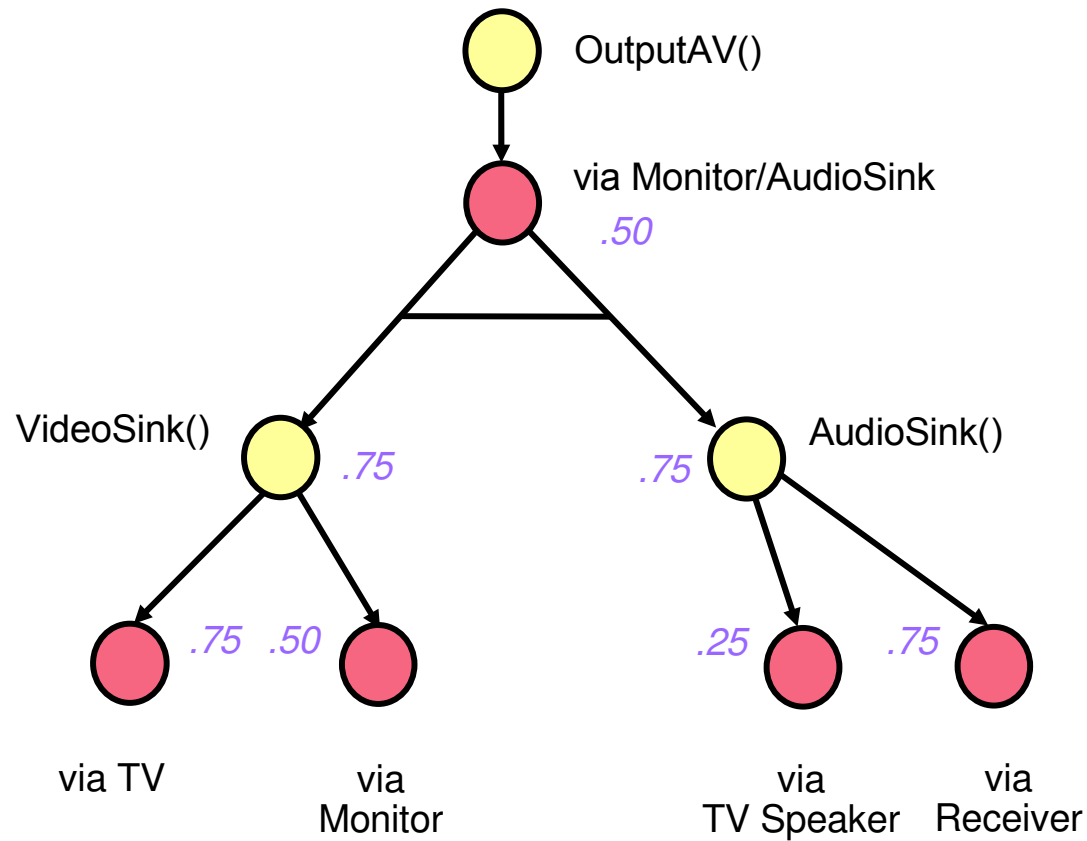
```
sink = FindAudioSink()
```

subgoals:

fader = \

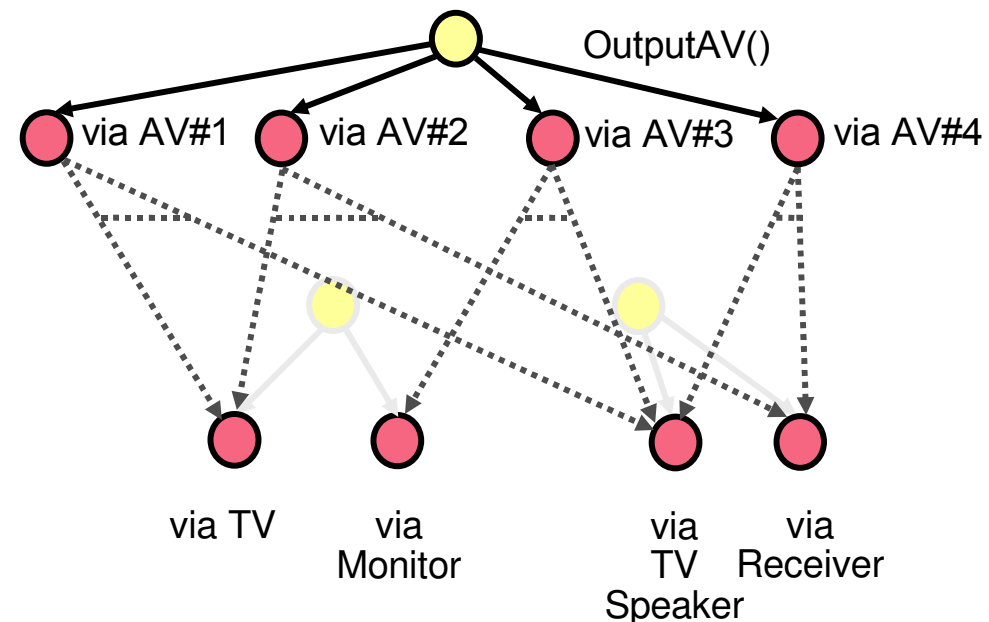
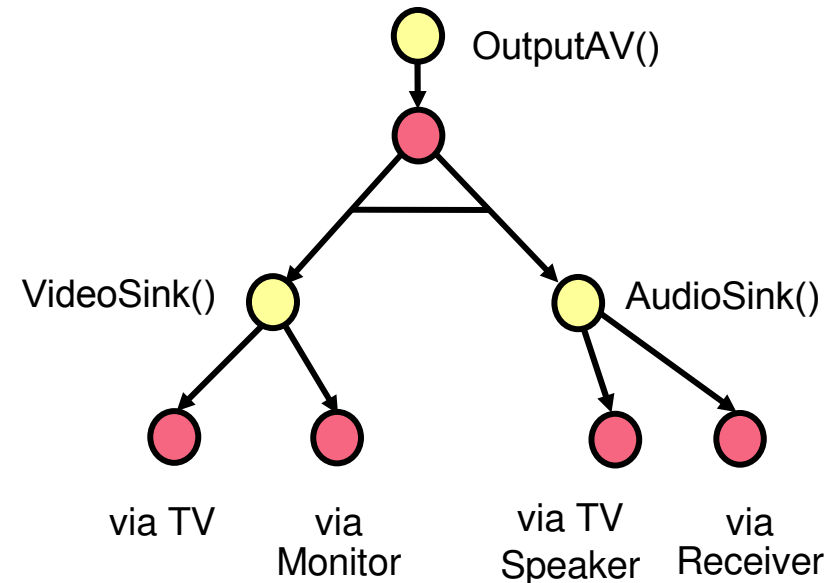
```
AudioFader(in_stream_type=source.stream_format,  
           out_stream_type=sink.stream_format)
```

Making the Best Choices



AND/"OR" Tree?

- Nodes may be "cloned"
 - GenerateSolution subgoal
 - "when necessary"
- Push decision making up the tree.



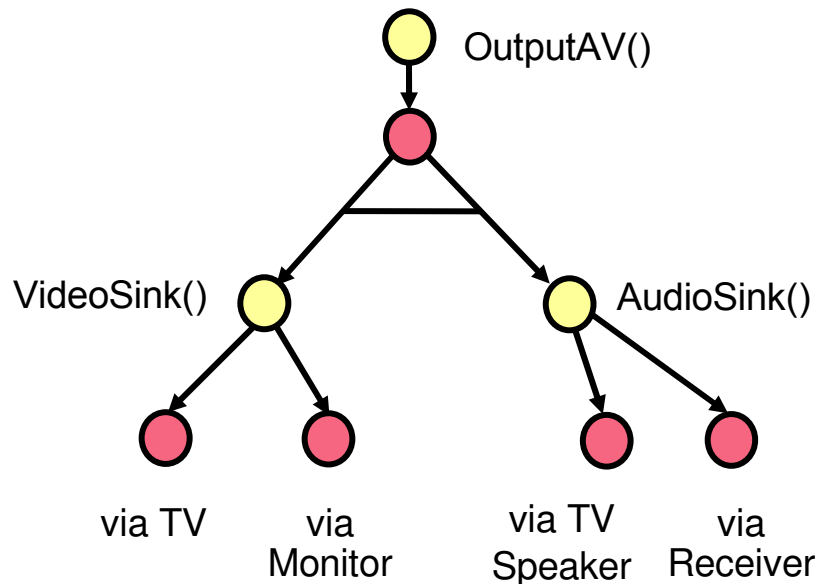
Free Communication

- Use parameters to influence the choices of the Planner
 - Pass down information
 - Fail if a choice is not good enough.

```
to FreeTeleconference(person_a,  
                      person_b):  
    via RequireFree:  
    subgoals:  
        tc = Teleconference(goal.person_a,  
                             goal.person_b,  
                             cost=0)  
  
    ...
```

```
to FreeTeleconference(person_a,  
                      person_b):  
    via CheckFree:  
    subgoals:  
        tc = Teleconference(goal.person_a,  
                             goal.person_b)  
  
    eval:  
        if tc.cost > 0:  
            fail("No free teleconference!")  
  
    ...
```

Questions?



Day 1, Session 1, Part 4
July 2006
Quanta JustPlay Seminar

Justin Mazzola Paluska
jmp@mit.edu