Improving Automatic Interface Generation with Smart Templates

Jeffrey Nichols • Carnegie Mellon University • jeffreyn@cs.cmu.edu • http://www.cs.cmu.edu/~pebbles/puc/

Problem

- Automatic interface generators have difficulty using domain-specific design conventions
  - Often added by interface designers after automatic generation
  - e.g. standard number pad on telephone
- Interfaces generated for end-users must apply conventions automatically
  - Users of the system are not likely to be trained interface designers
- Conventions must be applicable to multiple generation platforms

Solution

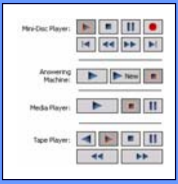
- Standardize on a set of "Smart Templates"
  - a template for each design convention (see list to right)
- Do not require every interface generator to implement every Smart Template
  - Smart Templates defined in terms of primitive spec. language features, so always renderable
  - Not implementing a template may be preferable for some generators (e.g. speech)
- Smart Templates have flexibility for use in multiple specifications
  - Templates are parameterized to cover the common and unique functions of each appliance

Templates

Some Smart Templates we have defined (we expect to find more):

date
datetime
dimmer
image
image-list
media-controls
mute-mic
mute-speaker
power
phone-dialpad
time-absolute
time-duration
volume
audio

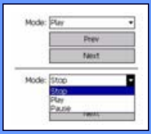
Examples of variation in one parameterized Smart Template (media-controls):



Smart Template Examples

media-controls

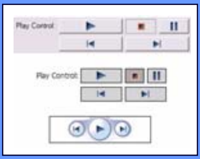
- Used for controlling the playback of a media stream – sound or video
- Possible devices include stereos, camcorders, answering machines, etc.



Generated Interface without Smart Template

Example #1

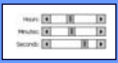
```
<group name="PlayGroup" is-a="media-controls">
  <labels><label>Play Controls</label></labels>
  <state name="Mode">
    <type>
      <enumerated><item-count>3</item-
count></enumerated>
      <valueLabels>
        <map index="1"><label>Play</label></map>
        <map index="2"><label>Stop</label></map>
        <map index="3"><label>Pause</label></map>
      </valueLabels>
    </type>
  </state>

  <command name="PrevTrack">
    <labels><label>Previous</label></labels>
  </command>

  <command name="NextTrack">
    <labels><label>Next</label></labels>
  </command>
</group>
```



Generated Interface with Smart Template on several platforms

time-duration

- Used for manipulating durations of time
- Possible devices include alarm clocks, media players, microwaves, etc.



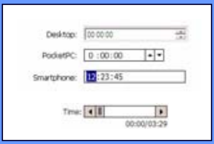
Generated Interface without Smart Template

Example #1

```
<group name="Counter" is-a="time-duration">
  <labels><label>Counter</label></labels>
  <state name="Hours">
    <type>
      <integer><min>0</min><max>8</max></integer>
    </type>
  </state>

  <state name="Minutes">
    <type>
      <integer><min>0</min><max>59</max></integer>
    </type>
  </state>
</group>
```

Example #2

```
<state name="SongLength" is-a="time-duration">
  <type>
    <integer><min>0</min><max>4440</max></integer>
  </type>
  <labels><label>Length</label></labels>
</state>
```



Generated Interface with Smart Template on several platforms

Background: Personal Universal Controller (PUC)

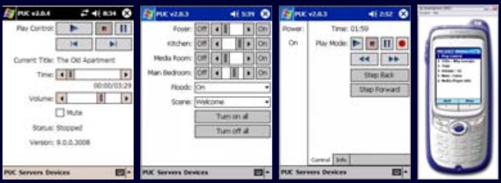
Problem

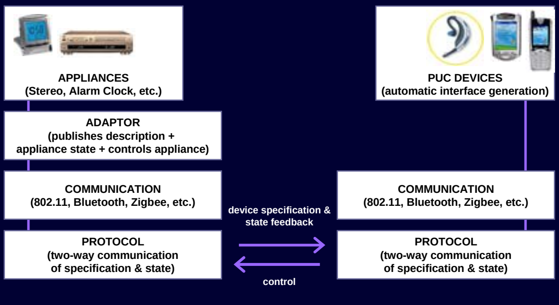
- Appliances, such as VCRs, kitchen appliances, etc., are becoming increasingly complex
- Increasing complexity often leads to poor user interfaces (Brouwer-Janse, et al.)

Solution

- Use hand-held devices that users already have (PDAs & mobile phones) to provide high-quality remote control user interface
- Automatically generate interfaces from an abstract specification of the appliance's functions

Example Interfaces





APPLIANCES
(Stereo, Alarm Clock, etc.)

ADAPTOR
(publishes description + appliance state + controls appliance)

COMMUNICATION
(802.11, Bluetooth, Zigbee, etc.)

PROTOCOL
(two-way communication of specification & state)

device specification & state feedback

control

PUC DEVICES
(automatic interface generation)

COMMUNICATION
(802.11, Bluetooth, Zigbee, etc.)

PROTOCOL
(two-way communication of specification & state)

Architecture diagram showing one connection (multiple are possible at both ends)

System Architecture

- **Appliance adaptors** allow system to connect to and control real appliances
- **Two-way communication protocol** allows UI to control appliance show current appliance state
- **Abstract specification language** allows functional description of appliances
- **Automatic interface generators** on multiple platforms (PocketPC, Microsoft's Smartphone, and TabletPC) and modalities (graphical and speech)