SUPPLE: Automatically Generating User Interfaces

Krzysztof Gajos
and Daniel S. Weld
Motivation
Motivation

• Current interfaces: complex & “One size fits all”
Motivation

• Current interfaces: complex & “One size fits all”
  ⇒ Adapt to users and tasks
Motivation

• Current interfaces: complex & “One size fits all”
  ⇒ Adapt to users and tasks

• Variety of display devices & interaction contexts makes hand-designed interfaces expensive
Motivation

• Current interfaces: complex & “One size fits all”
  ⇒ Adapt to users and tasks

• Variety of display devices & interaction contexts makes hand-designed interfaces expensive
  ⇒ Adapt to device characteristics
Motivation

• Current interfaces: complex & “One size fits all”
  ⇒ Adapt to users and tasks

• Variety of display devices & interaction contexts makes hand-designed interfaces expensive
  ⇒ Adapt to device characteristics

⇒ Automatic interface generation is a scalable solution
Approach

• Develop abstract representation for:
  • Interfaces
  • Display devices
  • Users

• Automatically generate interfaces from the abstractions
SUPPLE Architecture

1. Interface Model
2. Application or Appliance
3. Device Model
4. Target Device
5. SUPPLE Interface Model
6. Application or Appliance
7. Display
8. Device Model
9. User Model
10. User's Info Space

Target Device
SUPPLE Architecture

Interface Model
Application or Appliance

SUPPLE

Device Model
Target Device

User Model
User's Info Space
Examples of Applications

• A Classroom Controller
• An interactive FTP Client
• A distributed jukebox
• A stereo controller
Automatically Rendered Interfaces for Classroom Controller
Road Map

- ✔ Introduction
- □ Modeling
  - □ Interfaces
  - □ Devices
  - □ Users
- □ Inner workings of SUPPLE
- □ Results and evaluation
- □ Related Work
- □ Conclusions and future work
Road Map

- Introduction
- Modeling
- Interfaces
- Devices
- Users
- Inner workings of SUPPLE
- Results and evaluation
- Related Work
- Conclusions and future work
Modeling User Interfaces

- **simple types:** \( \text{int|float|string|bool} \)
- **derivative types:** \( \langle \tau, C_{\tau} \rangle \)
- **vectors:** \( \text{vector}(\tau) \)
- **containers:** \( \{ \tau_i \mid i \in 1...n \} \)
- **actions:** \( \tau \rightarrow \text{nil} \)
Modeling User Interfaces

- **simple types:** \( \text{int|float|string|bool} \)
- **derivative types:** \( \langle \tau, C_\tau \rangle \)
- **vectors:** \( \text{vector(}\tau) \)
- **containers:** \( \{\tau_i \mid i \in 1...n\} \)
- **actions:** \( \tau \rightarrow \text{nil} \)
Modeling User Interfaces

- **simple types:** $int | float | string | bool$

- **derivative types:**

- **vectors:**

- **containers:**

- **actions:**
Modeling User Interfaces

- simple types:
- derivative types: \( \langle \tau, C_\tau \rangle \)
- vectors:
- containers:
- actions:
Modeling User Interfaces

- simple types:
- derivative types:
- vectors: \( \{ \tau_i \}_{i \in 1 \ldots n} \)
- containers: \{bool, <string, {Computer 1, Computer 2, Video}>\}
- actions:
Modeling User Interfaces: Optional Attributes

• Label
• Set of likely values
• Exact value required
Road Map

- ✔ Introduction
- ✔ Modeling
  - ✔ Interfaces
- ✔ Devices
- ✔ Users
- ✔ Inner workings of SUPPLE
- ✔ Results and evaluation
- ✔ Related Work
- ✔ Conclusions and future work
Road Map

- Introduction
- Modeling
  - Interfaces
  - Devices
  - Users
- Inner workings of SUPPLE
- Results and evaluation
- Related Work
- Conclusions and future work
Modeling Device Capabilities
Modeling Device Capabilities

• Device constraints
Modeling Device Capabilities

- Device constraints
- Available widgets
  - Primitive widgets
  - Container widgets
Modeling Device Capabilities

• Device constraints
• Available widgets
  • Primitive widgets
  • Container widgets
• Match cost function for primitive widgets
Modeling Device Capabilities

• Device constraints

• Available widgets
  • Primitive widgets
  • Container widgets

• Match cost function for primitive widgets

• Navigation cost function for containers
Examples of Available Widgets

- Pointer and Keyboard
- Touch Screen
Examples of Available Widgets

Pointer and Keyboard

Touch Screen
Examples of Available Widgets

Pointer and Keyboard

Touch Screen
Examples of Available Widgets

Pointer and Keyboard

Touch Screen
Match Cost Function
For Primitive Widgets

$$\text{Match}(\langle \text{int}, [0,10]\rangle, \ 	ext{Level 7}, \text{change value from 7 to 8} \ ) = 3$$
Match Cost Function
For Primitive Widgets

\[ \text{Match}(<\text{int}, [0,10]>, \quad \text{Level} \ 7 \quad , \ \text{change value from 7 to 8} \ ) = 3 \]

\[ \text{Match}(<\text{int}, [0,10]>, \quad \text{Level} \ 7 \quad , \ \text{change value from 7 to 10} \ ) = 5 \]
Match Cost Function
For Primitive Widgets

\[ \text{Match}(\langle \text{int}, [0,10]\rangle, \text{Level 7}, \text{change value from 7 to 8} ) = 3 \]

\[ \text{Match}(\langle \text{int}, [0,10]\rangle, \text{Level 7}, \text{change value from 7 to 10} ) = 5 \]

\[ \text{Match}(\langle \text{int}, [0,10]\rangle, \text{Level 7}, \text{change value from 7 to 10} ) = 1 \]
Navigation Cost Function
For Container Widgets

• Inputs:
  • A transition type
  • A container widget

• Output: an estimate of user effort to navigate the interface
Example of Navigation Cost
Example of Navigation Cost
Road Map

☑ Introduction
☑ Interface
☑ Devices
☐ Users
☐ Inner workings of SUPPLE
☐ Results and evaluation
☐ Related Work
☐ Conclusions and future work
Road Map

- Introduction
- Modeling
- Interfaces
- Devices
- Users
- Inner workings of SUPPLE
- Results and evaluation
- Related Work
- Conclusions and future work
Modeling Users With Traces
Modeling Users With Traces

- *Trace* as a model of usage pattern composed of *Trails* [Wexelblat and Maes, CHI 1999]
Modeling Users With Traces

• *Trace* as a model of usage pattern composed of *Trails* [Wexelblat and Maes, CHI 1999]

• *Trail* as a “coherent” sequence of elements the user interacted with
Modeling Users With Traces

• *Trace* as a model of usage pattern composed of *Trails*  
  [Wexelblat and Maes, CHI 1999]

• *Trail* as a “coherent” sequence of elements the user interacted with

• Trail format independent of rendering
Modeling Users With Traces

• **Trace** as a model of usage pattern composed of **Trails** [Wexelblat and Maes, CHI 1999]

• **Trail** as a “coherent” sequence of elements the user interacted with

• Trail format independent of rendering

\[ T = \{ <\text{root}, -, -> \}
\langle \text{"Left light:Power"}, \text{off, on} \rangle
\langle \text{"Vent"}, 1, 3 \rangle
\langle \text{"Projector:Input"}, \text{video, computer 1} \rangle
\ldots \} \]
Road Map

- Introduction
- Modeling
  - Interfaces
  - Devices
  - Users
- Inner workings of SUPPLE
- Results and evaluation
- Related Work
- Conclusions and future work
Road Map

☑ Introduction
☑ Modeling
☑ Interfaces
☑ Devices
☑ Users
☐ Inner workings of SUPPLE
☐ Results and evaluation
☐ Related Work
☐ Conclusions and future work
Rendering As Optimization
• **Cost function** ($\$\$) -- estimated user effort to manipulate a rendering of the interface
Rendering As Optimization

• **Cost function** ($\,\$\,) -- estimated user effort to manipulate a rendering of the interface

• Inputs:
  • A rendering
  • A user trace
Rendering As Optimization

- **Cost function** ($\cdot$) -- estimated user effort to manipulate a rendering of the interface

- **Inputs:**
  - A rendering
  - A user trace

- Cost function derived from
  - Match cost function
$\mathcal{L}(\phi, \mathcal{T})$
Rendering As Optimization

\[ (\phi, \mathcal{T}) \equiv \sum_{T \in \mathcal{T}} \left( |T| - 1 \sum_{i=1}^{N} (\phi, e_{i-1}, e_{i}) + M(\phi(e_{i}), v_{old}, v_{new}) \right) \]

Iterate over all entries in all trails
Rendering As Optimization

\[ $(\phi, T) \equiv \sum_{T \in T} |T| - 1 \sum_{i=1}^{N(\phi, e_{i-1}, e_i)} + M(\phi(e_i), v_{old_i}, v_{new_i}) \]

Iterate over all entries in all trails

Cost of navigating between successive elements
$\$(\phi, T) \equiv \sum_{T \in T} \sum_{i=1}^{|T|-1} N(\phi, e_{i-1}, e_i) + M(\phi(e_i), v_{old_i}, v_{new_i})$

Iterate over all entries in all trails

Cost of navigating between successive elements

Cost of manipulating the value of the current element
A constrained branch-and-bound search algorithm (like A*) -- at each step assign a widget to an abstract interface element

Guaranteed to find a rendering with the lowest cost
Design Choices:

• **Constraint propagation methods:**
  - None, Forward Checking (FC), Full

• **Variable ordering:**
  - Bottom-up, Top-down, MRV

• **Admissible Heuristic:**
  - Estimate of the total cost of the entire interface
Performance

<table>
<thead>
<tr>
<th>Method</th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>none/bottom-up</td>
<td>89.1</td>
</tr>
<tr>
<td>FC/bottom-up</td>
<td>351.3</td>
</tr>
<tr>
<td>full/bottom-up</td>
<td>70.8</td>
</tr>
<tr>
<td>full/MRV</td>
<td></td>
</tr>
<tr>
<td>full/top-down</td>
<td></td>
</tr>
</tbody>
</table>

- Time in seconds:
  - none/bottom-up: 89.1
  - FC/bottom-up: 351.3
  - full/bottom-up: 70.8
  - full/MRV: 2
  - full/top-down: 2

University of Washington
Computer Science & Engineering
Road Map

- Introduction
- Modeling
- Interfaces
- Devices
- Users
- Inner workings of SUPPLE
- Results and evaluation
- Related Work
- Conclusions and future work
Results And Evaluation

• Adapting to device characteristics
• Building an interactive interface
• Conceptual study
  • Adapting to usage patterns
Adapting to Device Characteristics

Classroom
Light Bank
Left: Light
Center: Light
Right: Light
Level:

A/V Controls
- Projector
  Power
  Input: Computer 1
  Screen

Vent
  Off
  Low
  Med
  High

Classroom
Light Bank
Left: Light
Center: Light
Right: Light
Level:

A/V Controls
Projector
  Power
  Input: Computer 1

Vent
  Off

Computer 1
Video

Screen: Lowered

Light Bank
Left: On
Center: Level: 7
Right: Level: 7

Input
  << 7 >>

Light Bank
Left: On
Center: Level: 7
Right: Level: 7

Input
  << 7 >>
An Interactive Interface
An Interactive Interface
An Interactive Interface
An Interactive Interface
An Interactive Interface
An Interactive Interface
Preliminary Study

- Four “experts”
- Same widget library and conditions as for SUPPLE
Preliminary Study

Human Designer A

SUPPLE
Preliminary Study

Human Designer B
Preliminary Study

Human Designer C
Preliminary Study

Human Designer C

SUPPLE with a trace
Preliminary Study

Human Designer C SUPPLE with a trace
Road Map

- Introduction
- Modeling
  - Interfaces
  - Devices
  - Users
- Inner workings of SUPPLE
- Results and evaluation
- Conclusions and future work
Related Work

- Personal Universal Controller (PUC) [Nichols, et al, UIST’02]
- iCrafter [Ponnecanti, et al, UbiComp’01]
Related Work

- Personal Universal Controller (PUC)  
  [Nichols, et al, UIST’02]

- iCrafter  [Ponnecanti, et al, UbiComp’01]

- Xweb  [Olsen, et al, UIST’00]

- GADGET  [Fogarty, et al, UIST’03]
Related Work

- Personal Universal Controller (PUC)  [Nichols, et al, UIST’02]
- iCrafter  [Ponnecanti, et al, UbiComp’01]
- GADGET  [Fogarty, et al, UIST’03]
- XIML  [Puerta & Eisenstein, IUI’02]
Contributions

• Formal definition of the problem
• User-specific rendering
• An efficient algorithm
• Evaluation of speed and quality
Future Work
Future Work

- Cross-device consistency (with Anthony Wu)
Future Work

• Cross-device consistency (with Anthony Wu)
• Learning of match and navigation cost functions
Future Work

• Cross-device consistency (with Anthony Wu)
• Learning of match and navigation cost functions
• Transforming interfaces
Future Work

- Cross-device consistency (with Anthony Wu)
- Learning of match and navigation cost functions
- Transforming interfaces
- Explicit customization
Future Work

• Cross-device consistency (with Anthony Wu)
• Learning of match and navigation cost functions
• Transforming interfaces
• Explicit customization
• Incorporate design heuristics
Future Work

• Cross-device consistency (with Anthony Wu)
• Learning of match and navigation cost functions
• Transforming interfaces
• Explicit customization
• More complex applications
Future Work

• Cross-device consistency (with Anthony Wu)
• Learning of match and navigation cost functions
• Transforming interfaces
• Explicit customization
• More complex applications
• Wider range of devices
Acknowledgments

• The anonymous subjects

• Comments from: Mark Adler, Alan Borning, Gaetano Borriello, Tessa Lau, Jeffrey Nichols, Steven Wolfman, Alexander Yates
More Information

- **Demo tonight!**
- **SUPPLE:**
- **Krzysztof Gajos:**
  kgajos@cs.washington.edu
- **Daniel S. Weld:**
  weld@cs.washington.edu