

VeriWeb: Automatically Testing Dynamic Web Sites

Juliana Freire http://www-db.bell-labs.com/~juliana Bell Labs

Joint work with Michael Benedikt and Patrice Godefroid

Web pages became very complex

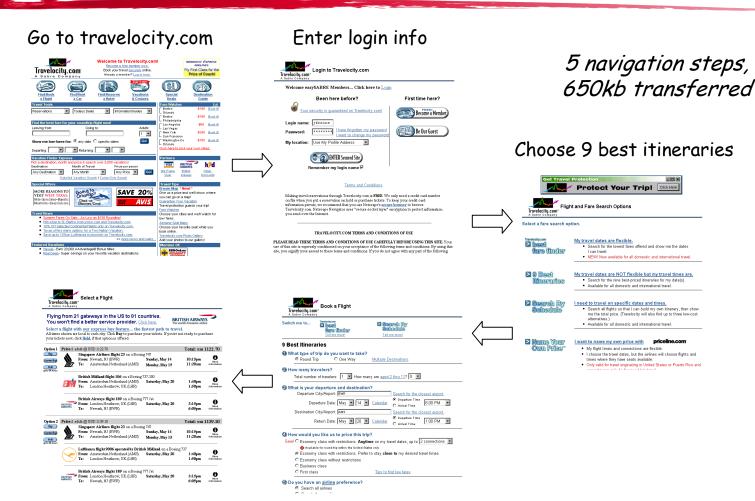




- Almost 90 different actions (85 links and 3 forms);
- 96 gif images;
- 113 lines of JavaScript code;
- ~570 lines of HTML

Web navigation became very complex





Flight list

Enter itinerary

Many things can go wrong



- Interactions between HTML pages
- Applications that run in Web pages (e.g., applets, javascript, plug-in applications)
- Applications that run on the server side (e.g., cgi scripts, database interfaces, logging applications, dynamic page generators, asp)
- Wide variety of servers and browsers
- Rapidly changing technologies, multiple standards and protocols

Many things can go wrong



Interactions between HTML pages

Applications that run in Web pages (e.g., applets,

javascript, plug-in applicat

Applications that run on the scripts, database interfactions dynamic page generators,

Wide variety of servers a

 Rapidly changing technology protocols



Need testing!

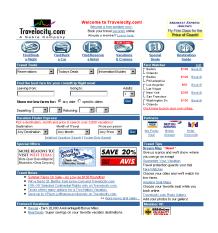
Web Site Testing: Challenges



- Sites are complex applications: hard to test
- ◆ Web sites are updated often (content and structure) → hard to keep testsuite up-to-date manually
- ♦ Web-based systems are highly intertwined with the environment (browsers, operating systems, database engines, applications, etc) → impossible to test stand-alone
- ◆ Users are inexperienced → unexpected behavior
 - Need to test system response to the actions of novice/inexperienced users: number of error conditions is huge

Changes to Travelocity













Before April, 2001

Before Feb. 2002





Since Feb, 2002

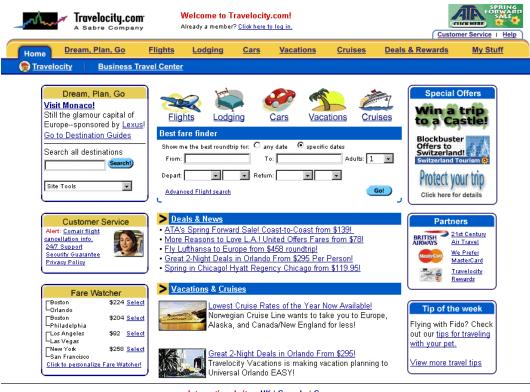
State of the Art



- Crawlers: check links and pages for common errors
 - -E.g., LinkCheck, SiteInspector, Weblint, Webtrends
 - -check for broken links, unreachable pages, HTML errors, etc
 - -limited to static Web pages: can't test Web applications
- ◆ Capture-replay tools: explore particular scenarios in a dynamic Web site
 - -E.g., Macrobot, Šilktest, etester, Rational TestStudio, etc.
 - -users record interactive Web site application scenarios that can be played back unattended on any predefined schedule
 - -user's must manually create testing scenarios

Too many paths





85 links, 3 forms

International sites: UK | Canada | Germany

Site Guide | Customer Service | About Travelocity.com | Privacy Policy | Security Guarantee | User Agreement Travel Protection | Advertise | Affiliates | Travelocity Magazine | Jobs | Awards | Contest Winners | Press Room | Investor Relations

Home | <u>Dream, Plan, Go</u> | <u>Flights</u> | <u>Lodging</u> | <u>Cars</u> | <u>Vacations</u> | <u>Cruises</u> | <u>Deals & Rewards</u> | <u>My Stuff</u>





Frommer's



@ 2001 Travelocity.com L.P. All rights reserved. Travelocity®, Travelocity.com and the Travelocity skyline logo are trademarks and/or service marks of Travelocity.com L.P. Use of this Website constitutes acceptance connected of the <u>Travelocity User Agreement</u> and <u>Privacy Policy</u>. CST# 2050374-40



Web sites change







British Midland flight 106 on a Boeing 737-300

British Airways flight 189 on a Boeing 777 Jet
From: London Heathrow, UK (LHR) Saturday, May 20
Te: Newark, NJ (EWR)

Lufthansa flight 9006 operated by British Midland on a Boeing 73

From: Amsterdam Netherland (AMS) Saturday, May 20 Te: London Heathrow, UK (LHR)

British Airways flight 189 on a Boeing 777 Jet
From: London Heuthrow, UK (LHR) Saturday, May 20
Te: Newsik, NJ (EWR)

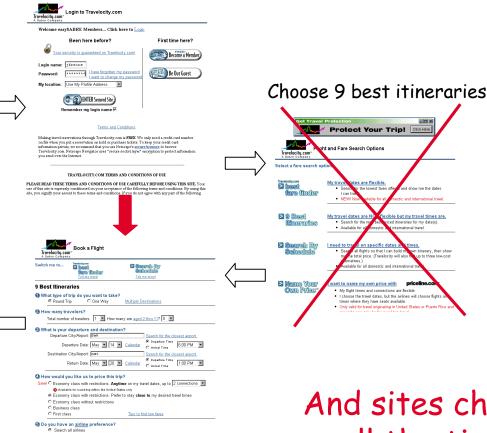
1:40рт 1:50рт

From: Amsterdam Netherland (AMS)
Te: London Heathrow, UK (LHR)

Option 2 Price: | adult @ USD 1139.10

Flight list

Enter login info



Enter itinerary

And sites change all the time!

10

VeriWeb



- Exercises multiple scenarios in a Web application that are automatically discovered by the system
 - Like a crawler it exhaustively searches the site for errors
 - Like a capture-replay tool, it exploits scenarios that include dynamic portions of sites
 - Unlike a crawler, it can go through forms
 - Unlike a capture-replay tool, no recording is needed

WWW2002

VeriWeb: Features



- Functional testing of both static and dynamic elements of Web site
- ◆ Regression testing
- ◆ Standard correctness checks (e.g., broken links, malformed URLs)
- ◆ Realistic testing
 - Tests are run through a Web browser and closely mimic users' actions

Outline

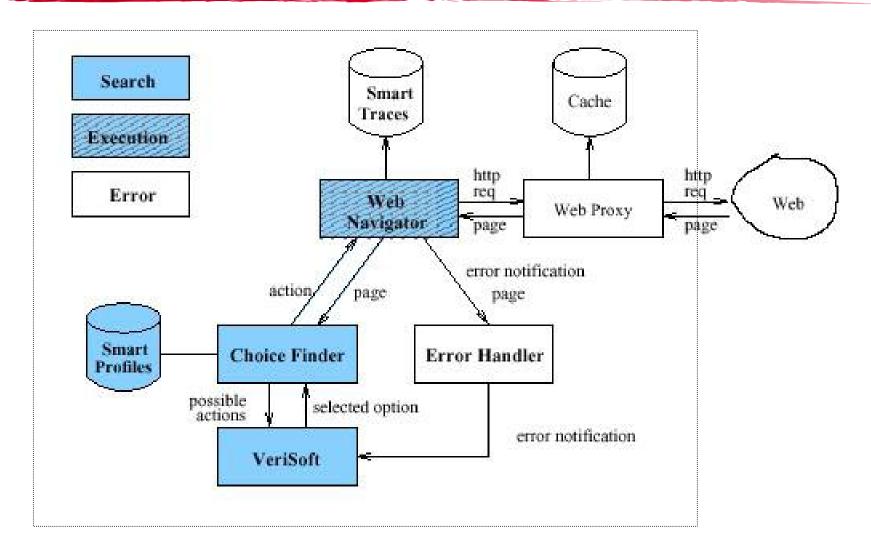


- VeriWeb architecture
- ♦ How does VeriWeb work?
- Automatically filling forms
- Controlling search
- Conclusion and Future Work

VeriWeb Architecture



14



Architecture Components



- ChoiceFinder
 - Find actions in page (JavaScript, links, forms)
- VeriSoft
 - Tool for systematic software testing ("model checking")
 - Controls the search
- ♦ WebNavigator
 - Executes browsing actions
- Error checker
 - Plugin your favorite checks (e.g., HTTP errors, HTML validation, ...)

How does VeriWeb work?



```
ExploreSite(startingURL, constraints)
currentPage = Navigator.load(startingURL);
while (true) {
 error = ErrorHandler(currentPage,constraints);
 if (error.status==true)
   VeriSoft.assert(currentPage,error);
 if (this page has been seen before)
   VeriSoft.abort(currentPage, ``cycle'');
 else {
   choices = ChoiceFinder(currentPage);
   selectedChoice = VeriSoft.toss(choices);
   currentPage =
     Navigator.execute(selectedChoice, choices);
   if (currentPage.error != null)
     VeriSoft.assert(currentPage,error);
```

Form: Which values to use?



Login:	
Password:	

Form: Many Choices



Search Auto Classifieds	
Keyword(s): BMW	(e.g. red mustang)
Price: From \$ Any	to \$ 40,000 💌
Sale Type: No Preference	Search More Options

- 23 different values for both the From and the to prices (i.e., Any, 1,000, 2,000, etc.);
- ◆ 4 values for the Sale Type (i.e., No Preference, By Owner, etc.),
- Virtually infinite number of possible choices for Keyword (s);

Each way to fill up this form corresponds to a different action

WWW2002

SmartProfiles



- Flexible mechanism for populating forms
 - Provide values for text fields
 - Constrain the values for enumerated fields (e.g., lists)
- ◆ Tester specifies a user view of the data
 - independent of the structure of the Web site
 - reusable as structure of Web site changes
 - not scenario-based: testing engine tries all 'reasonable' combinations of profiles

SmartProfile: Specification



```
<!DOCTYPE smartprofile [</pre>
<!ELEMENT smartprofile (signature* |
 profile*)>
<!ELEMENT signature (name, field+)>
<!ELEMENT field (name, synonym) >
<!ATTLIST field key CDATA>
<!ELEMENT profile (name, signature,
  fieldvalue+)>
<!ELEMENT fieldvalue (name, reqexp) >
<!ELEMENT regexp (#PCDATA) >
<!ELEMENT name (#PCDATA)>
<!ELEMENT synonym (#PCDATA)>
1 >
```

SmartProfiles: Signature



```
<signature>
   <name> CarType </name>
   <field key=true>
     <name> Make </name>
      <synonym> *carmake* </synonym>
   </field>
   <field key=false>
      <name> Model </name>
   </field>
   <field key=false>
      <name> Year </name>
   </field>
   <field key=false>
      <name> Interior </name>
   </field>
   <field key=false>
      <name> Type </name>
   </field>
</signature>
```

SmartProfiles: Profiles



```
cprofile>
   <name> HondaCivicSedan </name>
   <signature> CarType </signature>
                                            ofile>
   <fieldvalue>
                                               <name> Rolls </name>
      <name> Make </name>
                                               <signature> CarType </signature>
                                               <fieldvalue>
      <regexp> Honda </regexp>
                                                  <name> Make </name>
   </fieldvalue>
   <fieldvalue>
                                                  <regexp> Rolls-Royce </regexp>
                                               </fieldvalue>
      <name> Model </name>
      <regexp> *Civic*Sedan* </regexp>
                                               <fieldvalue>
   </fieldvalue>
                                                  <name> Model </name>
                                                  <regexp> *Corniche* </regexp>
   <fieldvalue>
                                               </fieldvalue>
      <name> Year </name>
                                               <fieldvalue>
      <regexp> 1999 2000 </regexp>
   </fieldvalue>
                                                  <name> Year </name>
   <fieldvalue>
                                                  <regexp> 2002 </regexp>
                                               </fieldvalue>
      <name> Interior </name>
      <regexp> Beige </regexp>
                                               <fieldvalue>
   </fieldvalue>
                                                  <name> Type </name>
                                                  <regexp> Convertible* </regexp>
   <fieldvalue>
                                               </fieldvalue>
      <name> Type </name>
      <regexp> Sedan </regexp>
                                            </profile>
   </fieldvalue>
</profile>
```

Filling out Forms



23

- Page analysis
 - find keywords related to fields in forms
- Profile-to-form matching
 - expression matching, default policies and thresholds

```
ExecuteForm(form,policy)
  formSchema = analyzeForm(form);
  candidateProfiles = findProfile(formSchema,policy);
  selectProfile = VeriSoft.toss(candidateProfiles);
  consistentCompletions = formCompletion(form, selectProfile);
  selectCompletion = VeriSoft.toss(consistentCompletions);
  currentPage = Navigator.fillOut(form, selectCompletion);
```

Controlling the Search



- Verisoft
- Filtering
 - Ignore "unimportant" actions, e.g., links to postscript, Word docs, external sites, mailto links,...

Cycle detection

- State vs. stateless: page that look identical may represent different states. E.g., the initial page of Amazon with and without cookies
- Configurable cycle-detection

Profile policies

- Number of profiles of interest to the user may be too large to exhaustively test every combination
- User can control the matching process, e.g., do not consider more than one match per signature in a path

Contributions: Summary



- Novel infrastructure to automate testing of dynamic Web sites
- Search algorithms for automatically exploring all the paths a user might follow in a Web application
- SmartProfiles as a high-level specification of test data to populate forms
- Strategies for automatically filling forms during site exploration.

Conclusions



26

- Creating profiles is an iterative process
 - System prompts tester for missing information
- Flexibility is key
- Currently experimenting with different Web sites
- ◆ Sites are often developed in an "ad hoc" manner: they are hard to test
 - Need web engineering guidelines
 - High-level specifications

Juliana Freire

Acknowledgements



- Avinash Vyas
- Randy Hackbarth
- ◆ Joanna McCaffrey