

# Leveraging User-Session Data to Support Web Application Testing

Authors: Sebastian Elbaum, Gregg Rotheermal,  
Srikanth Karre, and Marc Fisher II

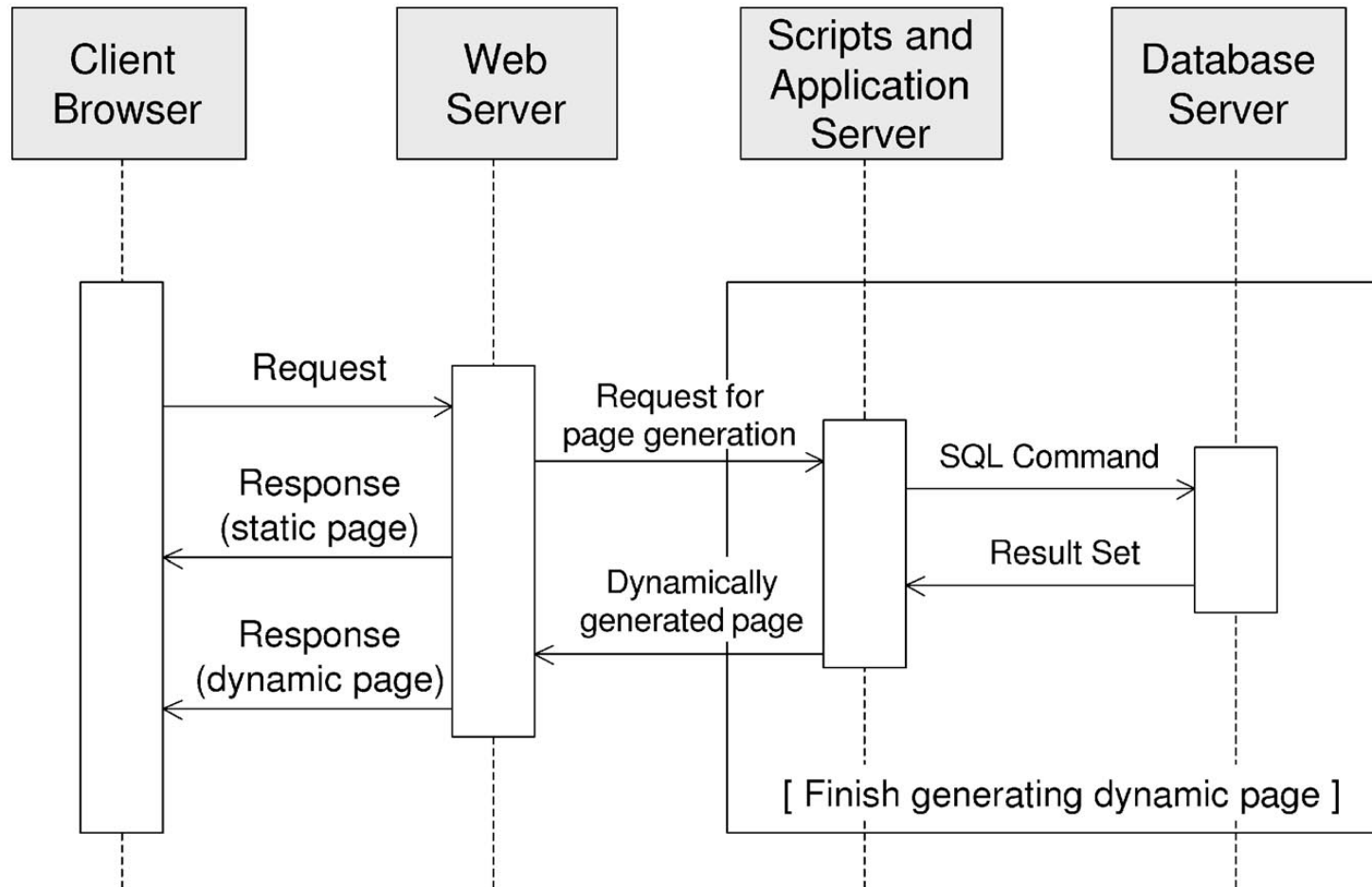
Presented By:

Rajiv Jain

# Outline

- Introduction
- Related Work
- Testing Techniques
- Experiment and Results
- Additional Considerations
- Conclusion

# What is a Web Application?



# Introduction

- Web applications
  - Can have large numbers of users
  - Change rapidly
  - Multi-tiered architectures
- Similar to GUIs
  - Event and user driven
  - Conventional testing techniques may not work

# Related Work

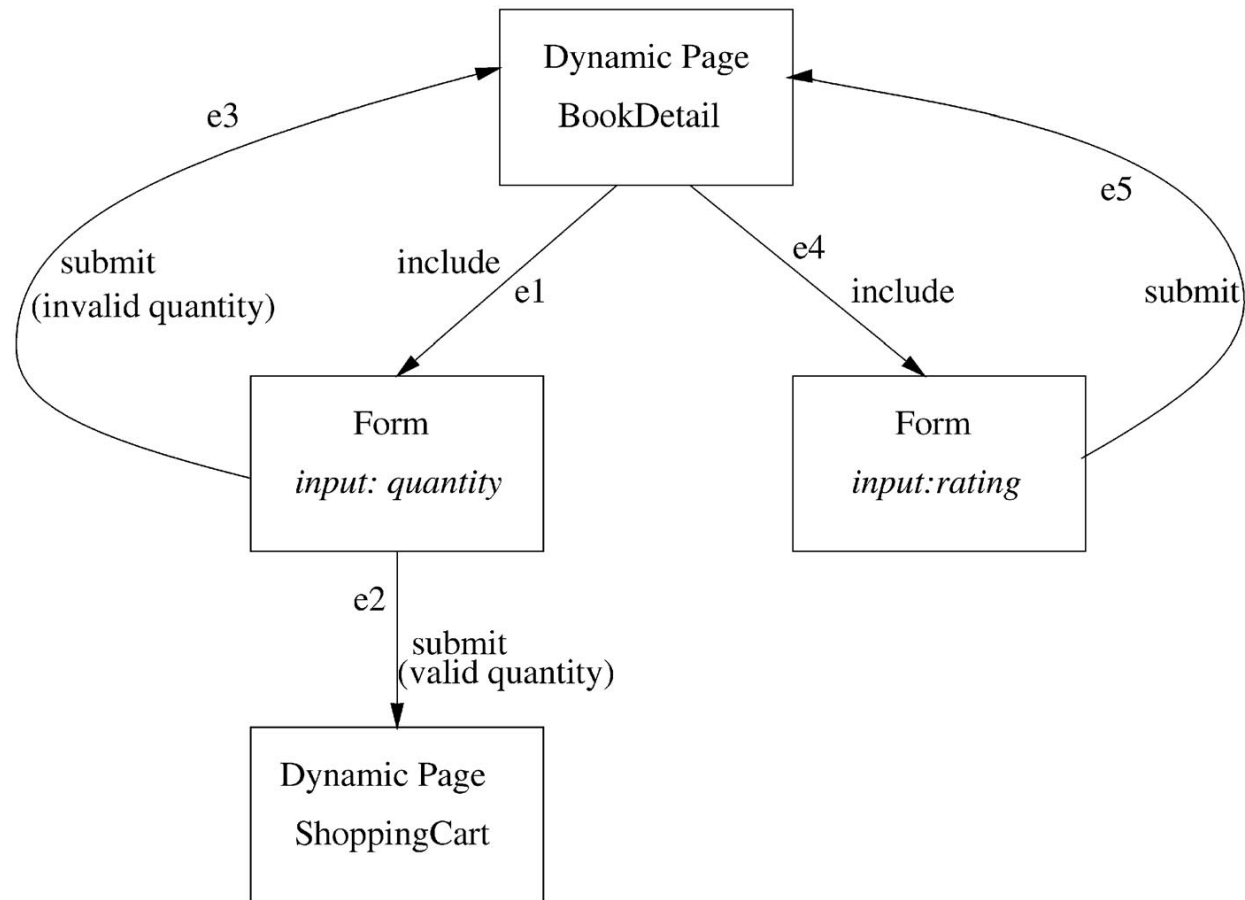
- Web Testing Techniques
  - Validation / Static analysis
  - Capture + replay tools (Selenium)
  - Conventional white-box testing techniques
    - Unit testing + integration testing
    - Dataflow
    - State-based
    - No work on fault detection, only coverage

# White Box Testing Techniques

- Ricca and Tonella
  - Models web application as graph
  - Nodes = Web objects
  - Edges = Relationships between objects
  - Test requirements and cases extracted from graph
  - Requires test engineer to create test cases

# White Box Testing Techniques

- Ricca and Tonella



# White Box Testing Techniques

- Ricca and Tonella-1 (WB-1)
  - Tests all edges
  - Uses linearly independent paths
  - Ignores circular links
- Ricca and Tonella-2 (WB-2)
  - Boundary values used as input
  - Each condition / All condition



# User-Session Testing Techniques

- User session is a TCP session
- Session is made up of several requests
- Request = URL + Name-value pairs
- Transparently collect several user sessions
- Uses sessions used to create test cases

# User-Session Testing Techniques

- Direct Reuse of User Sessions (US-1)
  - Analogous to a capture replay tool
- Combining Different User-Sessions (US-2)
  - Randomly combines two user-sessions with overlapping requests
- User Sessions with Form modifications (US-3)
  - Test cases randomly delete one value character
    - One case per name-value pair
    - One case modifying all name-value pairs

# Hybrid Testing Techniques

- Combines user-session and white box tests
- Hybrid 1 (HYB-1)
  - Match user-sessions to requirements in WB-1/2
  - If unable to match sessions to requirement, requirement is ignored
- Hybrid 2 (HYB-2)
  - Expands hybrid-1 by creating cases for unmatched requirements

# Experiment Setup

- Research Questions
  - How effective are the techniques
  - Does technique appropriateness vary with fault type?
  - Relationship between number of user sessions and test suite effectiveness
- Independent Variable = 7 test techniques
- Dependent Variable = Coverage, Fault Detection

# Experiment Setup

- Test subject is an online bookstore
  - Implemented in Perl (67 functions, 399 blocks)
  - Uses MySQL (7 tables)
  - Hosted on a Apache web server
- Fault seeding was used
  - 50 “realistic” faults added by 2 grad students
    - Scripting faults
    - Form faults
    - Database query faults

# Experiment Setup

Metric	WB-1	WB-2	US-1	US-2	US-3	HYB-1	HYB-2
Test Suite Size	28	64	85	84	407	1004	1089
Requests	99	241	1975	1919	2742	1428	1397

- Test Suite Creation
  - White box
    - 75 hours spent creating the representation model
    - Completed prior to fault seeding
  - User-session
    - 73 users navigated the website using IE
    - Sessions were recorded with Apache/Javascript
  - Oracle
    - Web application output prior to fault seeding

# Results

Metric	WB-1		WB-2		US-1		US-2		US-3		HYB-1		HYB-2	
	abs	%	abs	%	abs	%	abs	%	abs	%	abs	%	abs	%
Block Coverage	263	66	306	76	263	66	255	64	288	72	260	65	270	68
Function Coverage	65	97	66	99	65	97	64	96	65	97	65	97	64	99
Faults Detected	22	54	24	58	23	56	23	56	26	63	23	56	23	56

Technique Combination	Blocks		Functions		Faults	
	abs	%	abs	%	abs	%
$(WB-2 \cap US-3)$	273	68	65	97	23	54
$(WB-2 - US-3)$	32	8	1	2	2	5
$(US-3 - WB-2)$	14	4	0	0	4	9
$(WB-2 \cup US-3)$	319	80	66	99	29	67

	Faults ranked in tiers				
	1-5	6-10	11-15	16-20	Rest
Average sessions affected	98% (83)	81% (69)	33% (28)	1% (1)	0% (0)
WB-1 detects	100% (5)	80% (4)	60% (3)	40% (2)	40% (16)
WB-2 detects	100% (5)	60% (3)	40% (2)	40% (2)	50% (20)

# Threats to Validity

- Need to study additional websites
- Experiment users may not be representative of normal users
- No comparison to other white box techniques
- Tester may not have implemented white box testing properly
- Fault seeding may be biased
- Uneven test suite size between techniques



# Additional Considerations

- Web-Application State
  - Output depends on more than URL, name-value pairs
  - Test cases have different meaning in different states
- Non-Determinism
  - Identical sets of input can produce different outputs
- Managing Evolving Test Suites
  - Over time large number of user sessions accumulated
  - Remove redundant cases by keeping same function, page, block coverage

# Conclusion

- Pros
  - New technique for web-testing
  - Appears to complement existing techniques
  - Not dependent on underlying technology
  - Little human effort required
- Cons
  - More experimentation needed
  - Requires stable application...good for beta testing
  - Practicality?

# Future Work

- Combining traditional techniques with user-session tests
- Filtering and reducing large amounts of user-sessions
- Costs of this technique versus others

# Questions

