Simulating Real-world Load Patterns

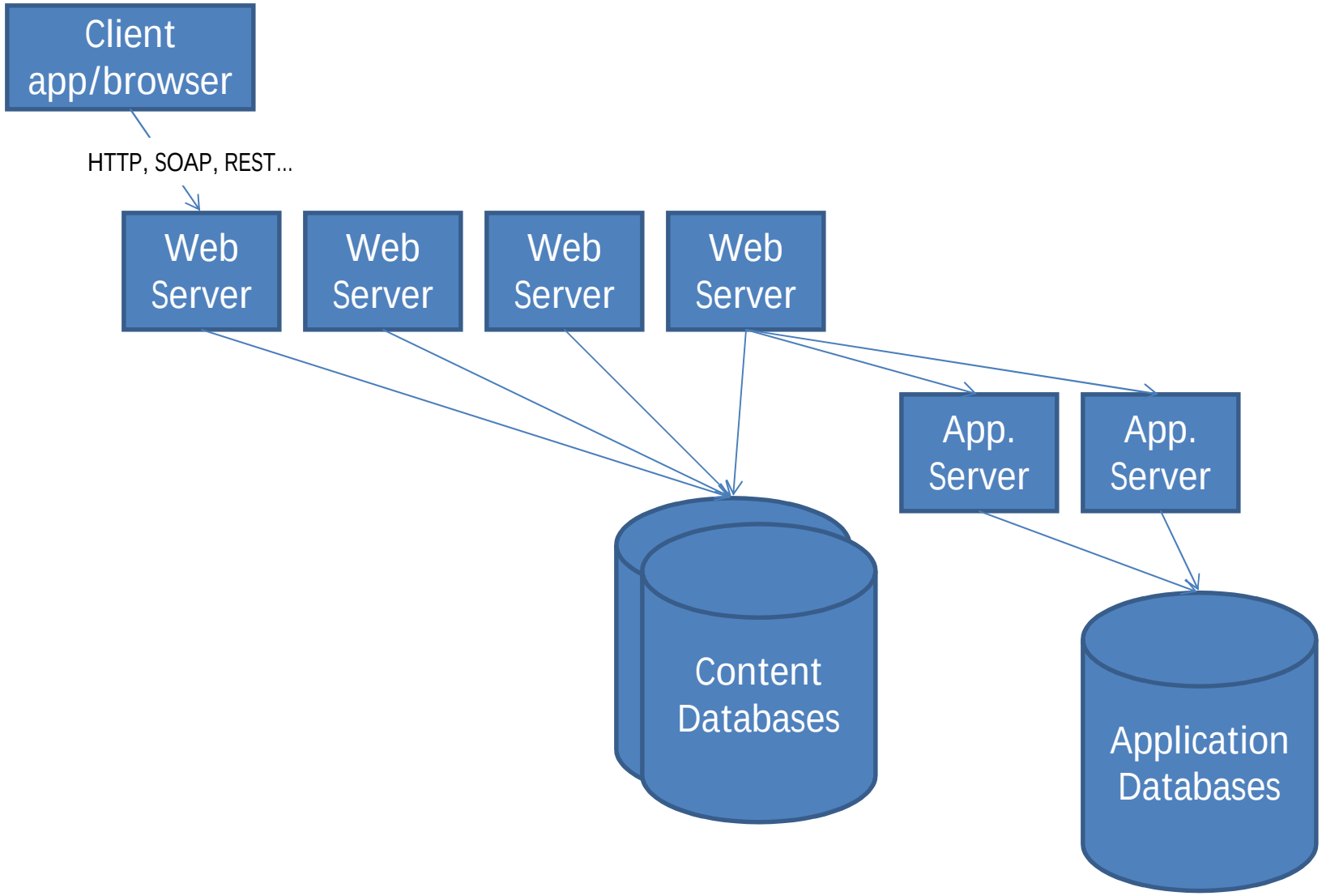
…when playback just won't cut it

Wayne Roseberry, Microsoft Corporation

Background: Microsoft SharePoint

- Web-based application server, part of Microsoft Office
  - Communication, issue tracking
  - Document management, Simple workflow
  - Enterprise search
  - Business application integration
  - Content management and publishing
  - Web browser & rich GUI client integration, web service and REST api's
- Original release 2001, current version Microsoft SharePoint 2010
- Fastest growing server product in Microsoft history

SharePoint Architecture



Background: Test Challenges

- Investigation in production is expensive, slow
- Which load patterns are typical and which are abnormal?
- Data samples are critical to performance and reliability
- Dynamic state makes playback testing ineffective

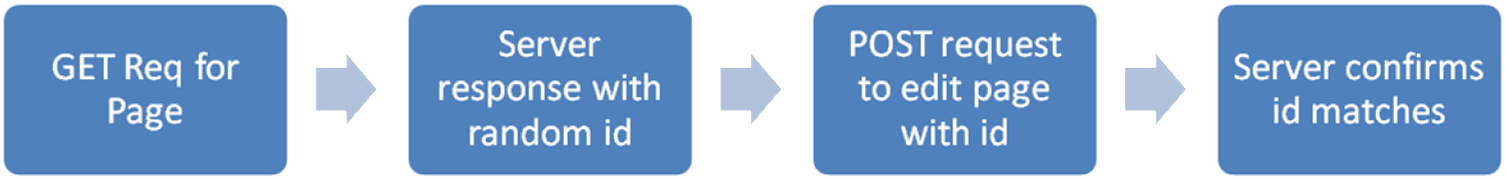
Test Challenge: Load patterns and data samples

- Extreme patterns find failures quickly, but are challenged for being unrealistic
- “Typical” patterns that mimic real usage are difficult to model, but are taken more seriously when they find failures
- Data sets on SharePoint are complex and dramatically affect the traffic pattern
  - E.g. a large document library will have larger impact on enumerations and queries that invoke conflicting locks in the database
  - E.g. very large documents will have higher cost on file manipulation actions
  - E.g. large number of unique page requests cause thrashing on in-memory caches

Test Challenge: Dynamic State

- Playback:
  - Record the exact HTTP traffic from a production sample, playback at a later time to the server as a test
- Dynamic state:
  - Random or unique values in the response calculated at runtime (document id's, security flags, session state) that must be preserved for follow up responses
  - Necessary sequences of actions (e.g. check out file, check in file) that may get captured mid-sequence

Example: Security token to block one-click attack on write operations



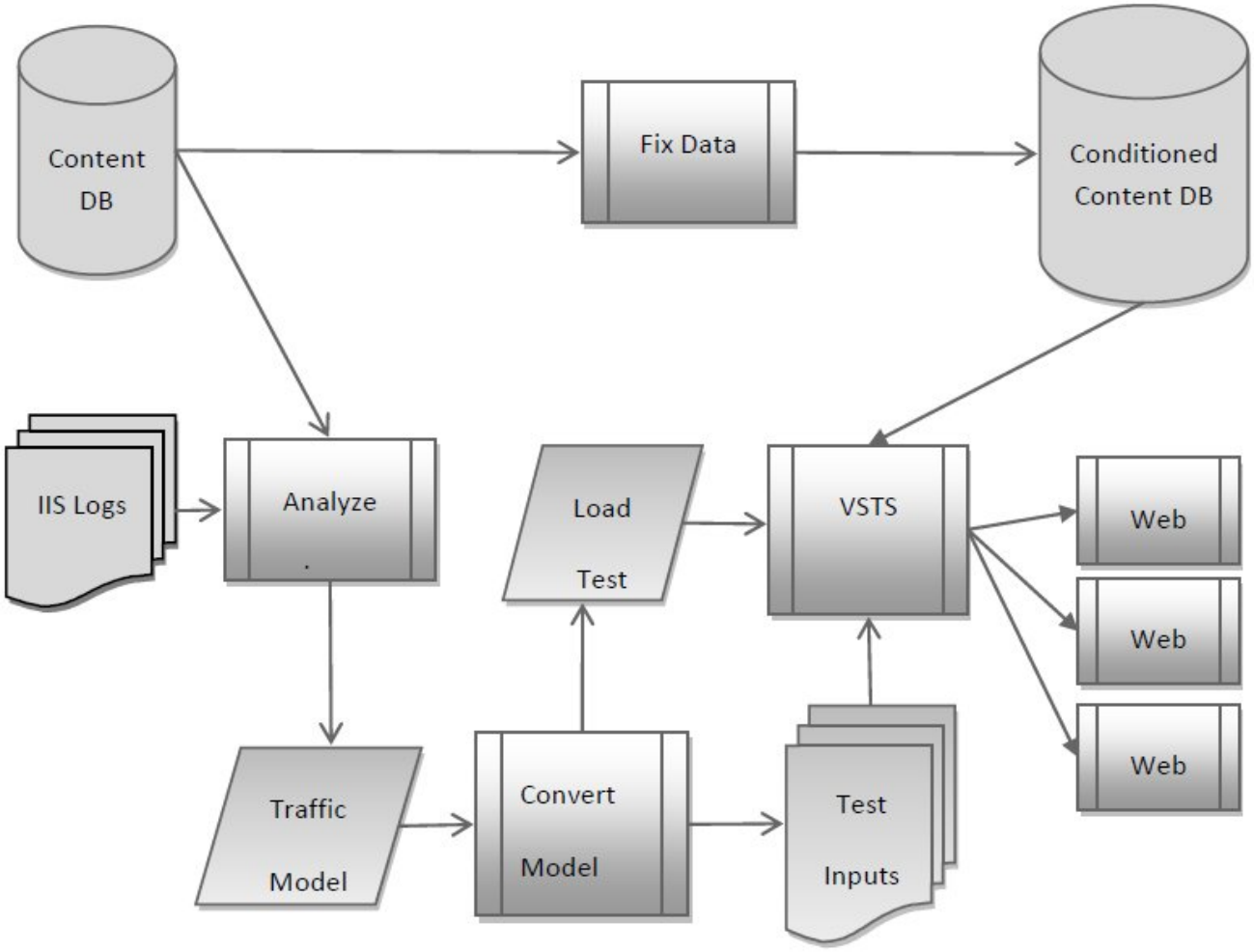
Therefore…

- Tests Need to Be Smart
  - A model of user activity, not a recording
  - Product aware, specialized to product features, not generic and blind
- Tests Need to Be Adaptable
  - System response will change, tests must respond to change
  - System state will change over time, tests must be state aware and behave appropriately
- Tests Must Be Able To Play For Variable Length
  - Different time span than original recording

What We Planned to Achieve

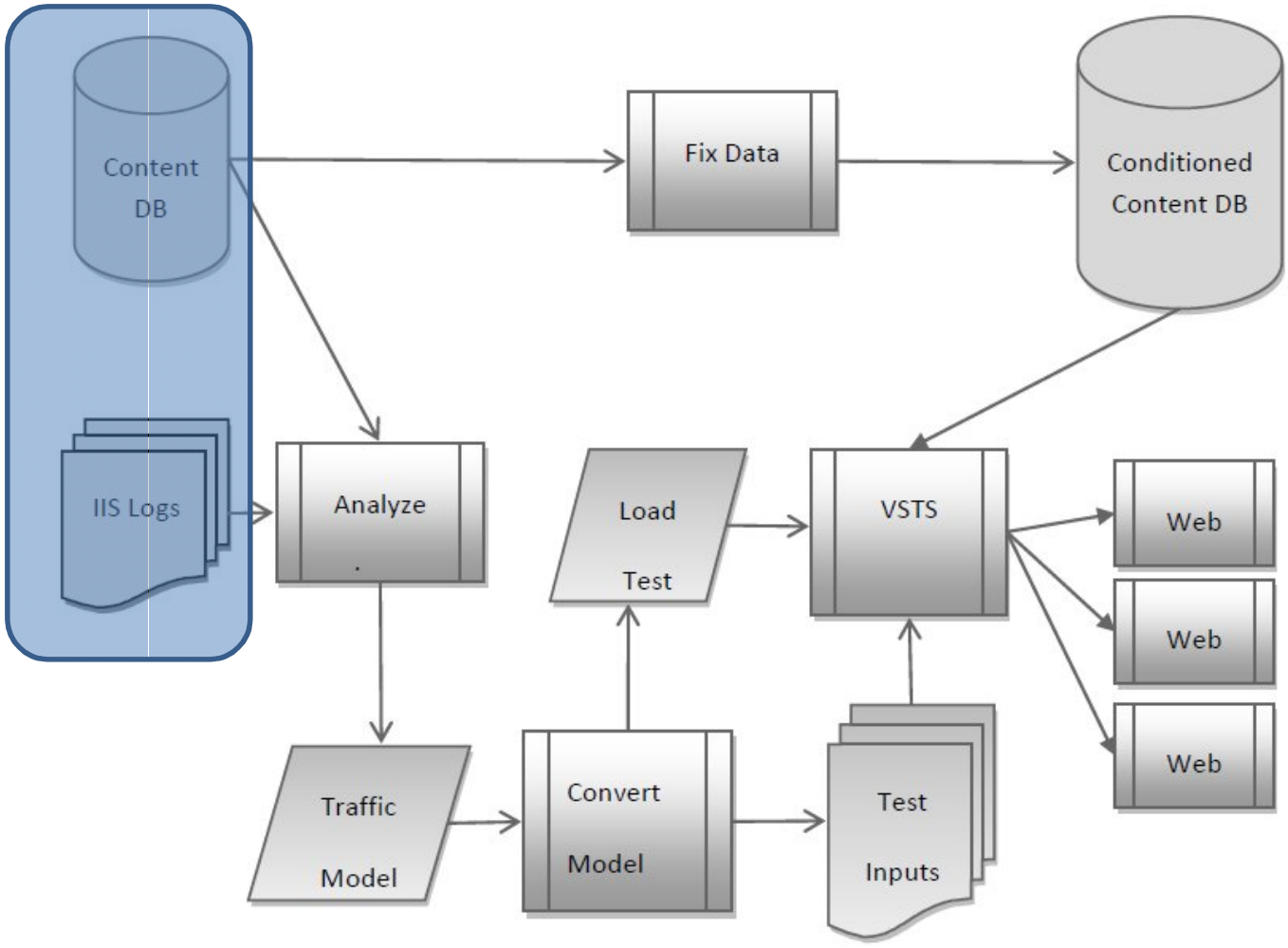
- Via tests predict performance and reliability flaws that manifest in production
- Find usage patterns from real-world that manifest bugs hard to find otherwise
- Simulate real-world traffic patterns to help prioritize bug fixes and set goals
- Create a regression suite for non-production problem investigation and fix validation
- Create a test lab environment to invent test methodologies for investigation and diagnosis
- Re-use our test solution to help customers with capacity planning and performance investigation

System Architecture



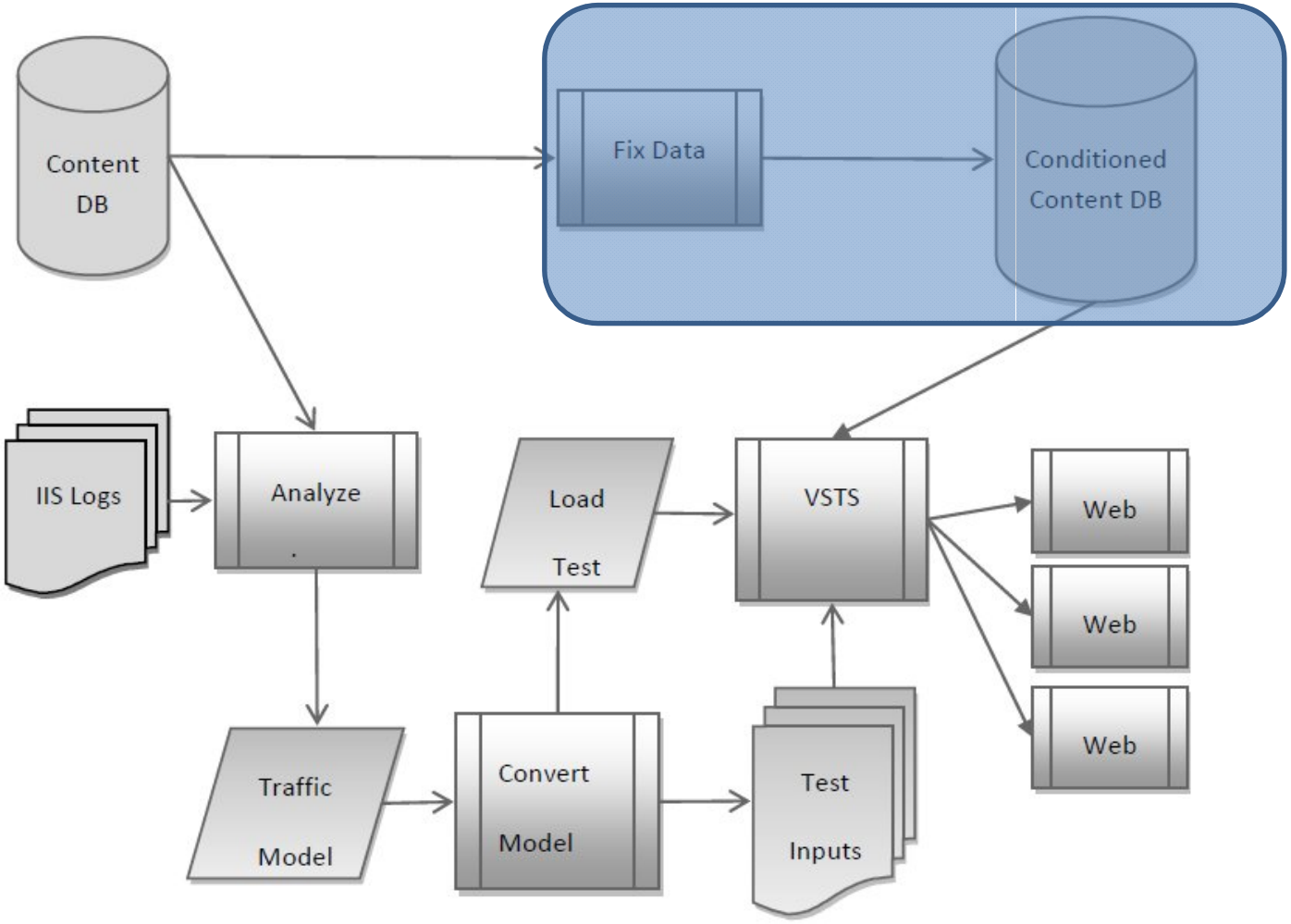
System Architecture

Get Content

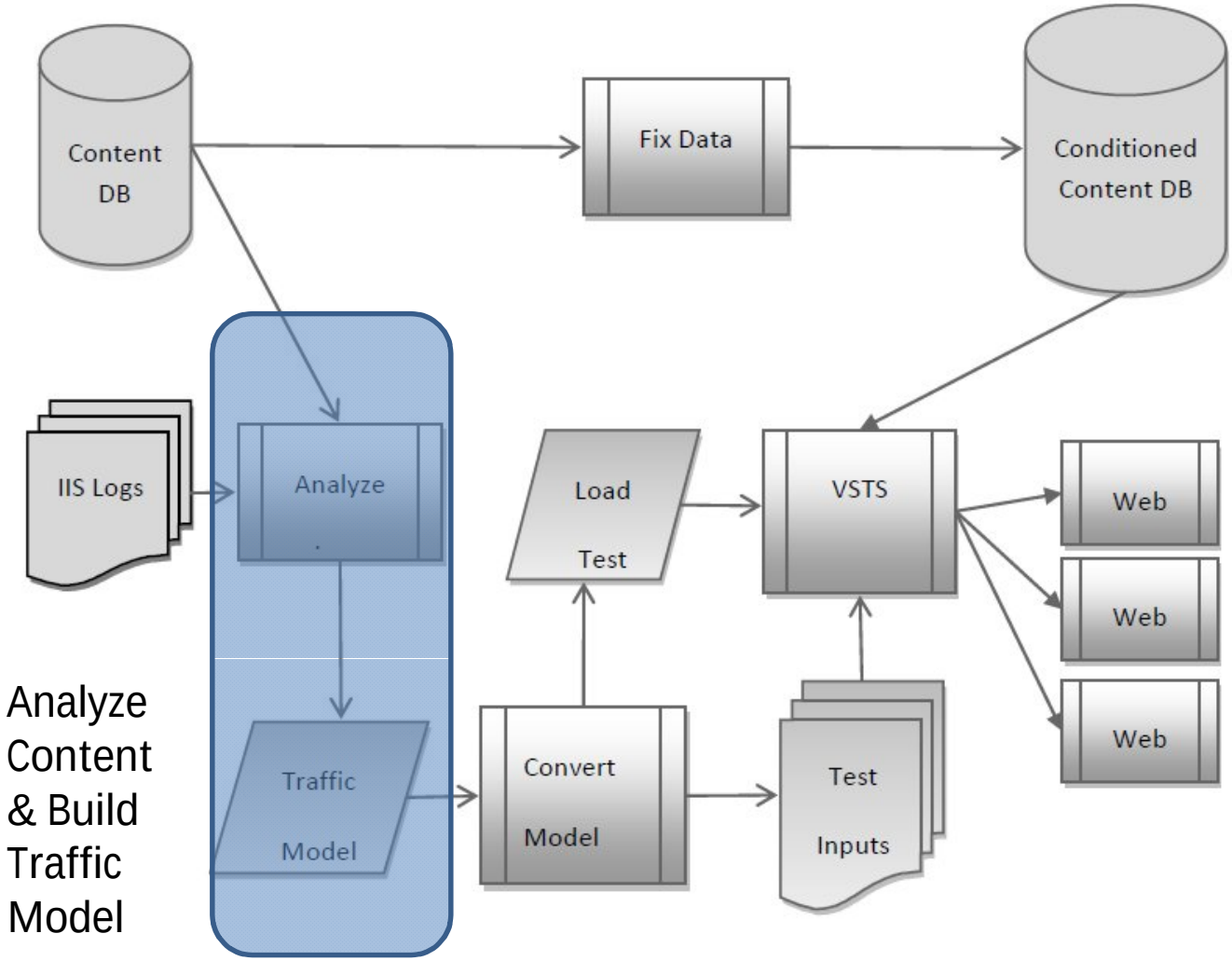


System Architecture

Copy Data And Map User permissions to Test Users

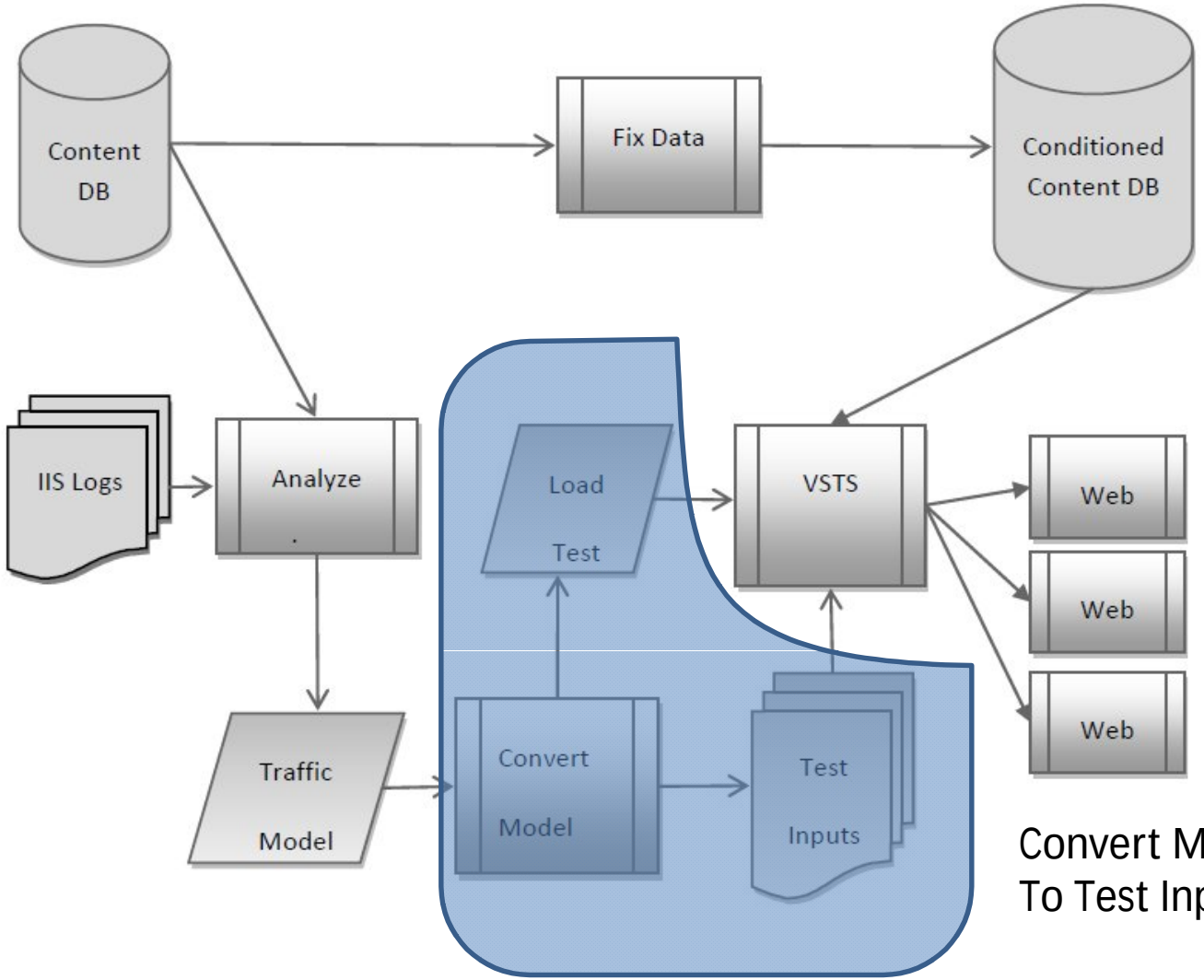


System Architecture



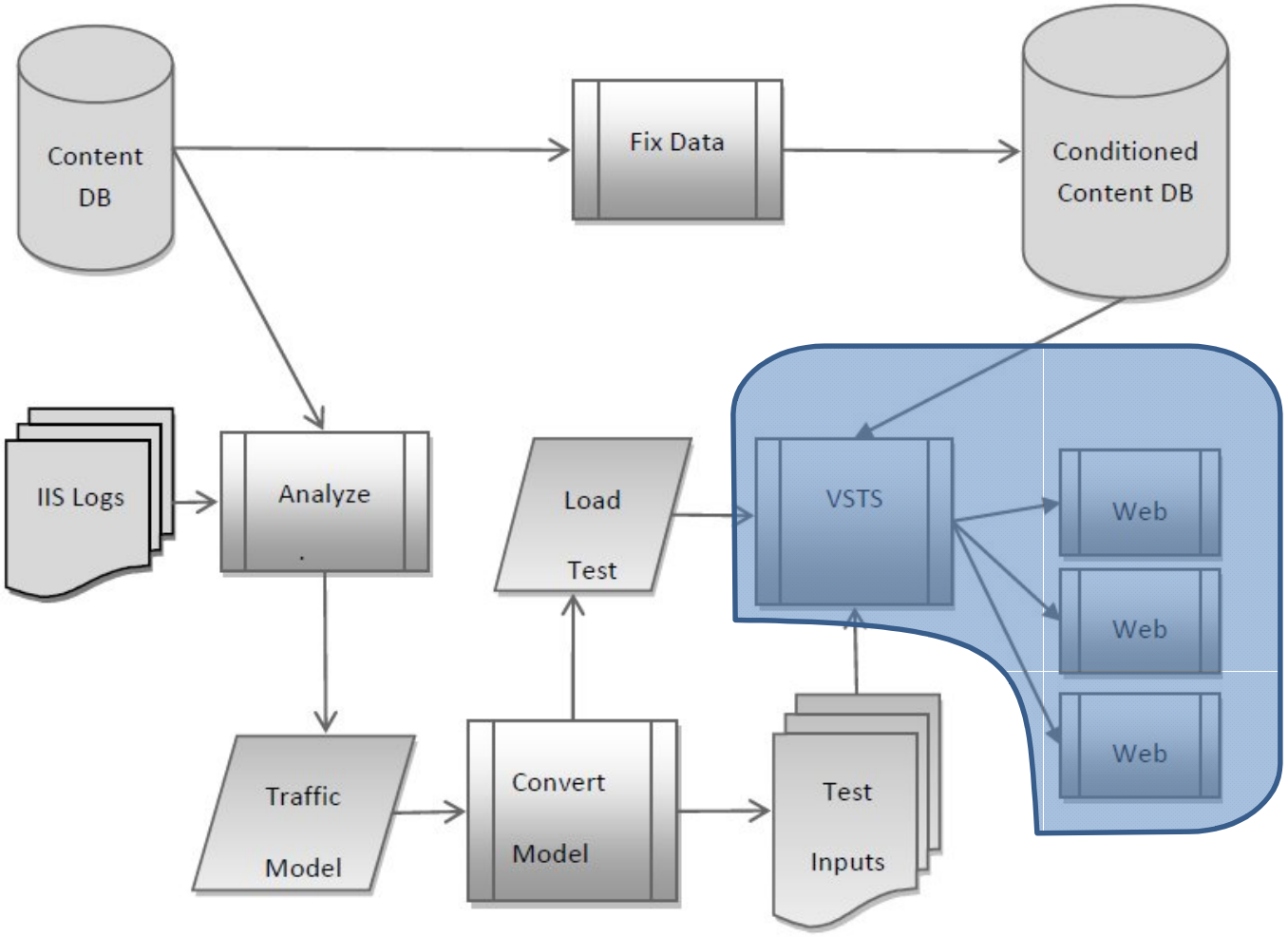
Analyze Content & Build Traffic Model

System Architecture



Convert Model
To Test Inputs

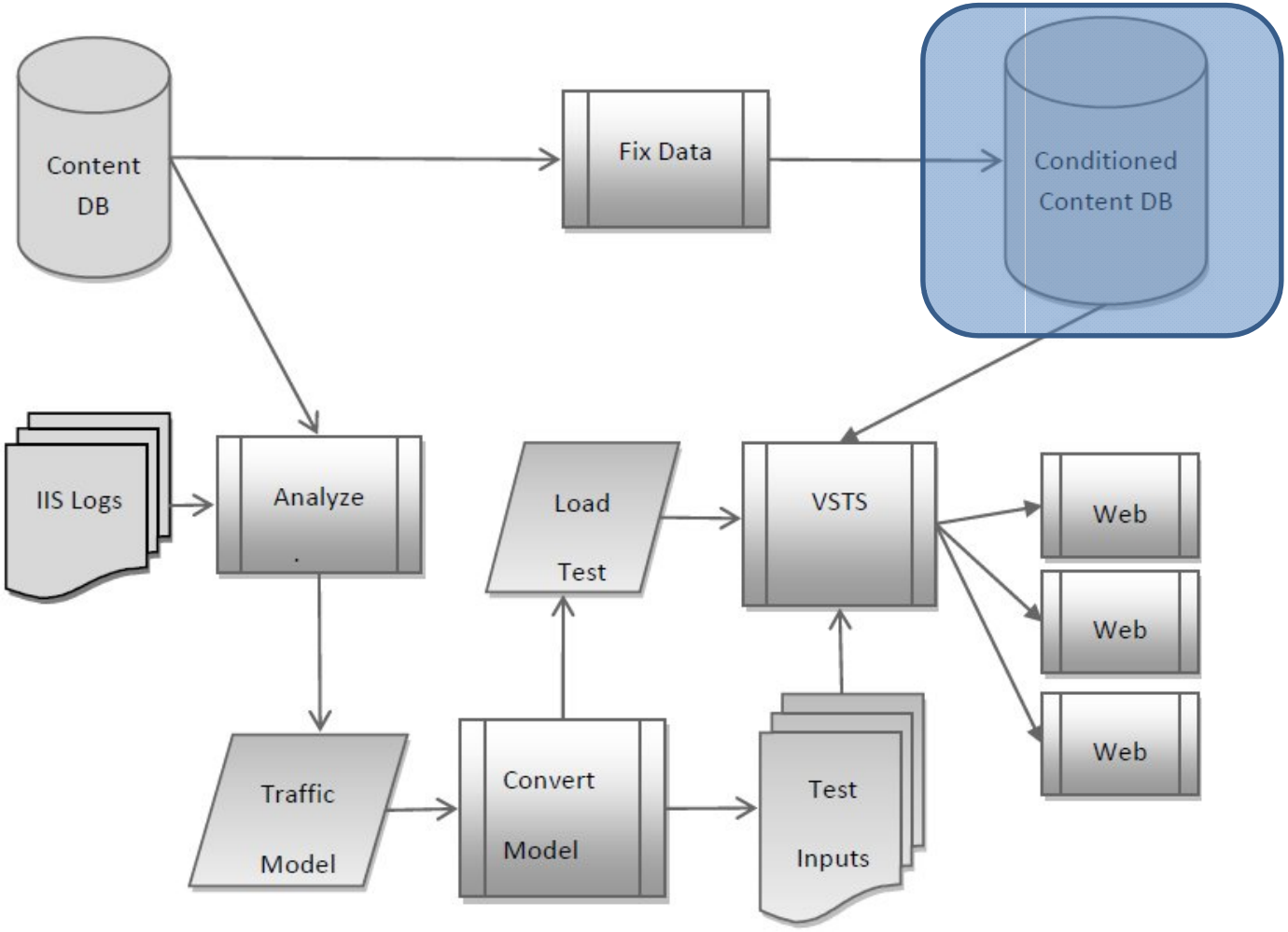
System Architecture



Visual Studio Custom Web Tests

System Architecture

Monitor
Reliability
During Test



Real-world Sites

- Office team portal (http://office)
  - 7,000 people, 7500 unique visitors per day
  - Team collaboration on documents, lists, reports, schedules
  - Seasonal workload based on Office team schedule
  - 155 requests per second peak hourly load
  - Large single document library for Office specifications and engineering documents
- Microsoft internal hosted collaboration (http://sharepoint)
  - Profile
    - Entire company, 100k + people, 80,000 unique visitors per day
    - Team collaboration, varied workload
    - World-wide use (mostly Redmond, USA)
    - 304 requests per second peak hourly load
  - Test changes
    - Changes for privacy
    - Subset of data, re-mapping load patterns
- Microsoft internal hosted personal sites (http://my)
  - Profile
    - 73,000 unique users per day
    - Peak hour 93 requests per second
    - Lots of automated access (RSS feeds, social updates in Outlook)
  - Test Changes
    - Personal sites map to real users, had to re-map to test users and permissions

Capacity Planning

•Same Workloads Used To Publish SharePoint Capacity Planning Guidance

Link to capacity Planning Material:
http://technet.microsoft.com/en-us/library/cc261716.aspx

Site From This Document	Report name on website
Office Product Group Portal	Departmental Collaboration
Microsoft IT Hosted Collaboration Portal	Intranet Collaboration
Microsoft IT Hosted Personal Site Portal	Social

•Load Test Kit Published for Customers

- Tool was re-packaged for external consumption and released to market
- Allows customer to sample their own load from existing systems and project hardware and configuration requirements to handle capacity

Defect Fix and Find Rates

Simulated Test Run Defect Find and Resolution Rates

	By Design		Dupe		Ext.		Fixed		Not Repro		Postponed		Won't Fix		Total #
	#	%	#	%	#	%	#	%	#	%	#	%	#	%	
Performance	25	8%	84	27%	11	4%	88	28%	43	14%	3	1%	55	18%	309
Other	13	10%	30	23%	5	4%	57	43%	20	15%		0%	8	6%	133
Grand Total	38	9%	114	26%	16	4%	145	33%	63	14%	3	1%	63	14%	442

Performance Defect Resolution Rate Overall (from a total set of 19269 reported defects)

	By Design	Duplicate	External	Fixed	Not Repro	Postponed	Won't Fix
%	4.21%	14.39%	1.90%	47.49%	8.31%	3.60%	19.83%

Comparison of Simulated Load to Other Performance Test Methods

- Lower: Fix Rate by 14%, Won't Fix 5%
- Higher: By Design 8%, Duplicate 15%, Not Repro 6%

Still more difficult to triage than component level performance tests

Comparable Bugs per tester: simulated run ~11 per tester (27 testers), other performance tests 12 per tester (1521 testers)

Limitations & Further Opportunities

- Production Systems Yielded Failures Not Found in Lab
  - Beta 2 until ship – most performance bugs found in production
  - We shipped with all in-production failures due to hardware/environmental failures
- Coverage Limitations
  - More, different types of operations
  - Probably biggest gap between in-lab reliability and in-production reliability
- Traffic Pattern Flattening v.s. Spiking
  - Load test maps constant percentages rather than spikes (e.g. 58.4 rps ranged from ~35 - ~65 rps spikes)
  - real-world system with 300 avg. RPS will range from 100-700 RPS on a minute-minute basis
  - Analyze as clusters of requests rather than single requests? Will it yield more failures?
- Improve Efficiency of Execution
  - Previous release, 2+ wks to build test environment every time (install, configure, upgrade data set, condition data)
  - Started this release ~ 1 wk
  - Got to 4 hours via automation
  - Fast time to start key to using as a regression tool during project end game
- Large Return From Monitoring Investments
  - Instrumentation, logging built into product, extended with tools
  - Ping-based reliability measurement used in lab and production (availability, failure rate, latency percentile spread)
  - Vast improvement on reproducibility, accounting for impact of discovered flaws, root cause investigation

Conclusions

- We proved that real-world simulation from traffic pattern models are feasible
- We proved that there is a valuable return on results in higher bug yields, better quality bugs and re-usability for customers
- Challenges still remain in increasing coverage, efficiency of execution and monitoring
- Investigation remains about value of achieving higher accuracy in simulation