Leveraging Design of Experiment (DOE) Towards a Reusable Scalability Testing Model for Defense Business Systems (DBS)

Lennore Vinnie Gunter

Dissertation Study Proposal
PhD, Management Information Systems
Walden University

ITEA Technology Review Conference
July 26, 2012
Overview

- Introduction
- Terminology
- Problem Statement
- Current State Scalability Testing for DBS
- Design of Experiment (DOE) as a Solution
- Results
- Benefits & Concerns
- Way Forward
- Wrap-Up
Introduction

• Background
  ◦ BS, MS – Computer Science, Hampton University (1993 & 1996)
  ◦ Department of Defense (DoD) IT Systems since 1995
    • Test & Evaluation (T&E) arena since 2001; Present – DT&E
  ◦ PhD–Management Information Systems; Walden Univ (pursuing)
    • Intent to use this research as dissertation study

• Purpose
  ◦ To research current statistical methods such as Design of Experiment (DOE) to determine applicability to resolving Defense Business Systems (DBS) scalability testing challenges

• Goal
  ◦ To establish a reusable, affordable, standardized developmental testing practice leveraging DOE that generates suitable DBS scalability test cases and accurate, reliable scalability test results
Terminology

- **Defense Business System (DBS)** - an information system operated by/for/on behalf of DoD including financial management, logistics and human resource management systems; non-weapon system

- **Design of Experiment (DOE)** - process of statistically determining which of many variables would most likely achieve a particular result
  
  - **Orthogonal Array Testing Strategy (OATS)** - systematic, statistical way of testing pair-wise interactions by deriving small set of test cases from large number of scenarios yet providing maximum test coverage

- **Developmental Testing** - performed early in SW development cycle to ensure SW meets functional/technical specs; large # test cases; Black Box

- **Scalability/Load Testing** - testing to determine maximum operating capacity of application or performance bottlenecks

- **Resource Utilization** - amount of resources application consuming (processor, memory, disk I/O, network I/O)

- **User Load** - number of users, concurrent users, transaction and data volumes that can run on a specific system configuration
Problem Statement

- For given DBS, user load can reach a 100,000 or more users per system
- Testing environment limitations constrain # of users and # system configurations we can test
- So, based upon testing constraints, we need a testing methodology that gives us confidence that:
  - we have adequately tested all configurations
  - we can trust results indicating full user base can be supported once system is deployed
Current State of Scalability Testing

- Testing is resource and labor intensive
  - Time constraints
  - Schedule constraints

- Testing can be costly
  - Tools
  - Personnel
  - Laboratories

- Testing is exhaustive
  - Trying to cover every test case
  - Impractical due to resource constraints
**Possible Solution: DOE - OATS**

- Many DOE Methods
  - Mathematically / Scientifically based
  - Credible, Reliable, Repeatable
  - **DOE Method: Orthogonal Array Testing Strategy (OATS)**
    - Systematic, statistical way of testing pair wise interactions
    - Very useful for integration testing of software components; tests various software configurations
    - Performs intelligent test case generation which can decrease costs and improve quality of testing
    - Drastically reduces number overall test cases
    - Detects all faults due to single parameter and 2 parameter interactions; Many multi-parameter
OATS: How Does It Work

1. Decide independent variables that will be tested for interaction = FACTORS
2. Decide the values that each FACTOR can take = LEVELS
3. Develop table of FACTORS and LEVELS
4. Generate system configuration Orthogonal Array (OA)
   - Tools/Algorithms Available: ALLPAIRS.exe
   - Open Source and Commercial Tools
5. This OA becomes your TEST CASES
### OATS: Applied to DBS Scalability Testing

1. **Decide Independent Variables → FACTORS**
   - # Users, CPU Utilization %, Transaction Type, Transaction Volume, RICE Object Priority,
   - # Cores/Processors

2. **Decide Values for each FACTOR → LEVELS**
   - # Users: 300, 600, 1000
   - CPU Use%: 30%, 60%, 100%
   - Trans Type: Online, Batch, Concurrent
   - Trans Volumes: 8500, 11000, 20,000
   - RICE Object Priority: Low, Medium, High
   - # Cores/Processors: 24, 48, 72
### 3. Develop Levels & Factors Table based upon chosen Independent Variables and Values:

<table>
<thead>
<tr>
<th>FACTORS</th>
<th># Users</th>
<th>Transaction Type</th>
<th>CPU Utilization (%)</th>
<th>Transaction Volumes (day)</th>
<th>RICE Object Priority</th>
<th>#Processors</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEVELS</td>
<td>300</td>
<td>Online</td>
<td>30</td>
<td>8500</td>
<td>Low</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>600</td>
<td>Batch</td>
<td>60</td>
<td>11000</td>
<td>Medium</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>Concurrent</td>
<td>100</td>
<td>20000</td>
<td>High</td>
<td>72</td>
</tr>
</tbody>
</table>

**Note:** 6 factors; 3 levels each = $3\times3\times3\times3\times3\times3 = 729$ test cases to cover all test configurations using non-DOE methods

### 4. Use ALLPAIRS.exe to generate OA of system test configurations = TEST CASES
OATS: Applied to Scalability Testing

5. ALLPAIRS OA Test Cases Generated = 16

<table>
<thead>
<tr>
<th>Case</th>
<th>Users</th>
<th>Type</th>
<th>CPUuse</th>
<th>Cores</th>
<th>Transactions</th>
<th>RICEPriority</th>
<th>pairings</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>300</td>
<td>online</td>
<td>30</td>
<td>48</td>
<td>8300</td>
<td>Low</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>300</td>
<td>batch</td>
<td>60</td>
<td>72</td>
<td>11000</td>
<td>Medium</td>
<td>17</td>
</tr>
<tr>
<td>3</td>
<td>300</td>
<td>both</td>
<td>100</td>
<td>72</td>
<td>20000</td>
<td>High</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>600</td>
<td>online</td>
<td>60</td>
<td>48</td>
<td>8300</td>
<td>Low</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>600</td>
<td>batch</td>
<td>30</td>
<td>24</td>
<td>11000</td>
<td>Medium</td>
<td>13</td>
</tr>
<tr>
<td>6</td>
<td>600</td>
<td>both</td>
<td>30</td>
<td>48</td>
<td>20000</td>
<td>High</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>1000</td>
<td>online</td>
<td>60</td>
<td>48</td>
<td>8300</td>
<td>Medium</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>1000</td>
<td>batch</td>
<td>60</td>
<td>72</td>
<td>11000</td>
<td>High</td>
<td>9</td>
</tr>
<tr>
<td>9</td>
<td>1000</td>
<td>both</td>
<td>30</td>
<td>48</td>
<td>20000</td>
<td>Low</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>1000</td>
<td>both</td>
<td>100</td>
<td>48</td>
<td>8300</td>
<td>Medium</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>300</td>
<td>online</td>
<td>-30</td>
<td>48</td>
<td>20000</td>
<td>High</td>
<td>14</td>
</tr>
<tr>
<td>12</td>
<td>300</td>
<td>both</td>
<td>-30</td>
<td>24</td>
<td>11000</td>
<td>Low</td>
<td>10</td>
</tr>
<tr>
<td>13</td>
<td>600</td>
<td>online</td>
<td>-30</td>
<td>48</td>
<td>20000</td>
<td>Medium</td>
<td>15</td>
</tr>
<tr>
<td>14</td>
<td>600</td>
<td>both</td>
<td>-30</td>
<td>24</td>
<td>11000</td>
<td>High</td>
<td>16</td>
</tr>
<tr>
<td>15</td>
<td>-1000</td>
<td>both</td>
<td>-100</td>
<td>-24</td>
<td>8500</td>
<td>Low</td>
<td>14</td>
</tr>
</tbody>
</table>

Note: Now we have a defined set of Test Cases to take to lab so we can focus resources, time and testing effort thus positively impacting cost, schedule & performance!!!
# Benefits & Concerns

<table>
<thead>
<tr>
<th>Benefits of Using DOE</th>
<th>Concerns with Using DOE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reduce Overall Test Cost</td>
<td>Only applicable in Black Box Testing</td>
</tr>
<tr>
<td>Can be applied to Unit, Integration, System and Acceptance Testing</td>
<td>Works best in early phases of testing</td>
</tr>
<tr>
<td>Reduce cycle time of test phase; Positively impact schedule</td>
<td>Need domain knowledge experts (SMEs, Developers, Business Analysts) involved to ensure right factor selection</td>
</tr>
<tr>
<td>Find maximum defects with minimal test cases; Enables better system performance</td>
<td>Arrays difficult to construct; Very tedious &amp; error prone by hand; Enlist OA freeware</td>
</tr>
<tr>
<td>Can aid in automated testing through generation of focused test cases</td>
<td>Can be overwhelming 1st time used but improves with continued use</td>
</tr>
</tbody>
</table>
Way Forward

- Apply initial OATS presented here to DBS entering developmental testing phase
- Document Results – Successes & Failures
  - Determine standard Independent Variable set
- Use results data to feed and inform model
- Update model; Continue iterations until confident process works and results reliable
- Document process/practice in handbook or guide; could develop as app
- Document results in a T&E/Scalability best practice database or website
- Socialize results and practice to T&E Community
Wrap-Up

- Research indicates methods such as OATS may be applicable to DBS scalability testing challenge
- Applying OATS may provide positive cost, schedule and performance impacts
- Must continue to apply, update and socialize
- Viable testing tool in developmental test toolkit
- Foundation laid for use of DOE (OATS) in DT Scalability Testing

OATS – A Reliable, Repeatable, Cost-Effective Scalability Testing Best Practice
Q & A

- For additional information contact:
  - Lennore Vinnie Gunter
    lennore.vinnie@gmail.com
    (757) 870-7163
References


