So first I will give a bit of background on what our problem was, what the process is, and the results we had.
The Challenges

• User Acceptance Testing (UAT) is part of our typical testing strategy.
• UAT is often a manual process making it expensive, time consuming, and heavily reliant on shared business resources.
• Long UAT cycles have also led to delayed deployment schedules.
• We had no test automation in place for UAT
• Existing test suites may not be optimal
The Challenges

• User Acceptance Testing (UAT) is part of our typical testing strategy.
• UAT is often a manual process making it expensive, time consuming, and heavily reliant on shared business resources.
• Long UAT cycles have also led to delayed deployment schedules
• We had no test automation in place for UAT
• Existing test suites may not be optimal
• How could we gain efficiencies and reduce the amount of time needed for UAT?
One of those solutions was using Combinatorial Test Design for UAT
COMBINATORIAL TEST DESIGN
Combinatorial Test Design

A National Institute of Standards & Technology (NIST) study concluded that the majority of software defects occurred because of the interaction between two variables.
Combinatorial Test Design

Combinatorial Test Design (CTD), or Pairwise Testing, is a test modeling approach that takes the parameters of a software application and combines the interactions between those parameters into an optimized set of test cases based on a defined level of interaction.
Benefits of CTD

• Create a large test suite from scratch without manually writing each test case

• Identify testing gaps in existing test suites

• Eliminate duplicate test interaction in existing test suites

• Allows testers to execute test cases more frequently
Combinatorial Test Design

- **Parameter** – a measurable factor of the system under test
  
  Ex. Browser
  
- **Value** – a characteristic of the parameter
  
  Ex. – Chrome, IE, Firefox, etc.

You are going to hear me repeat these phrases over and over again throughout this presentation.
This simple product of set A multiplied by set B, gives us a new set called AB. Set AB contains every interaction between the values in Set A and Set B, and every interaction is present in our new set. Test cases can be structured in the same manner – a value of one parameter being tested with the values of other parameters.

Software systems are of course not this simple!

They can be comprised of dozens of parameters, with many possible values.
How CTD Works

The CTD tool creates the Cartesian Product based on your parameters and values, and creates the lowest number of possible test cases by removing duplications and combining interactions.
CTD Example

Medicare Insurance Claims
CTD Example

Parameters

• Gender
• Primary or secondary insurance policy
• Claim Payment Status
• Correct Claims Submission
• Prior Documentation
• Claim Type
• Patient on Social Security
• Patient Disability Status
• Patient Work Status
CTD Example

Taking our test requirements and writing them in a parameter & value format simplifies the import/entry into the CTD tool

<table>
<thead>
<tr>
<th>Gender</th>
<th>Coverage</th>
<th>Claim Payment Status</th>
<th>Claim form</th>
<th>Documentation</th>
<th>Claim type</th>
<th>SSA</th>
<th>Disabled</th>
<th>Working Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>Primary</td>
<td>Fully paid</td>
<td>Property formatted</td>
<td>Record of insurance</td>
<td>Professional</td>
<td>SSA</td>
<td>Disabled</td>
<td>Working</td>
</tr>
<tr>
<td>Female</td>
<td>Secondary</td>
<td>Partially paid</td>
<td>Improperly formatted</td>
<td>No record of insurance</td>
<td>Medical</td>
<td>No SSA</td>
<td>Not Disabled</td>
<td>Not Working</td>
</tr>
<tr>
<td></td>
<td></td>
<td>None paid</td>
<td></td>
<td></td>
<td>Dental</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CTD Example

With this simple example, there are 1,152 potential interactions that need to be accounted for full test coverage

<table>
<thead>
<tr>
<th>Gender</th>
<th>Coverage</th>
<th>Claim Payment Status</th>
<th>Claim form</th>
<th>Documentation</th>
<th>Claim type</th>
<th>SSA</th>
<th>Disabled</th>
<th>Working Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>Primary</td>
<td>Fully paid</td>
<td>Properly formatted</td>
<td>Record of insurance</td>
<td>Professional</td>
<td>SSA</td>
<td>Disabled</td>
<td>Working</td>
</tr>
<tr>
<td>Female</td>
<td>Secondary</td>
<td>Partially paid</td>
<td>Improperly formatted</td>
<td>No record of insurance</td>
<td>Medical</td>
<td>No SSA</td>
<td>Not Disabled</td>
<td>Not Working</td>
</tr>
<tr>
<td></td>
<td></td>
<td>None paid</td>
<td></td>
<td></td>
<td>Dental</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
If we begin manually writing all of the possible test cases, tweaking one of the values, we see a lot of duplication right away.

How many times do we really need to test a male patient with primary coverage?
Not only are we saving time in test execution, creating this test suite takes only a couple of minutes.
CTD TOOLS
CTD Tools

HEXAWISE
MORE COVERAGE. FEWER TESTS.

Testcover.com
Your test design requirements. Covered.

IBM
CTD PROCESS
There are 6 steps to the process and I am going to talk about each one.

CTD Process Phases

1. Discovery & Test Planning
2. Entering Parameters
3. Adding Restrictions
4. Setting the Interaction Levels
5. Executing Test Case Creation
6. Test Case Coverage Analysis
We accomplish this like any test planning effort. We consult SMEs, we review product requirements and documentation, and we review historic bugs.
We take the parameters and their corresponding values, and put them into the CTD tool of our choice
Restrictions are pairs in our tests that we know for a fact will not appear in
4. Setting The Interaction Levels

Hexawise tests cover all possible 2-way interactions in just: **10 tests** of a total possible **1,152 tests**
5. Executing Test Case Creation

<table>
<thead>
<tr>
<th>Gender</th>
<th>Coverage</th>
<th>Claim Payment Status</th>
<th>Claim form</th>
<th>Documentation</th>
<th>Claim types</th>
<th>SSA</th>
<th>Disabled</th>
<th>Working</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>Primary</td>
<td>Fully paid</td>
<td>Properly formatted</td>
<td>Record of insurance</td>
<td>Professional</td>
<td>SSA</td>
<td>Disabled</td>
<td>Not Working</td>
</tr>
<tr>
<td>Female</td>
<td>Secondary</td>
<td>Partially paid</td>
<td>Improperly formatted</td>
<td>No record of insurance</td>
<td>Professional</td>
<td>No SSA</td>
<td>Not Disabled</td>
<td>Working</td>
</tr>
<tr>
<td>Female</td>
<td>Secondary</td>
<td>Fully paid</td>
<td>Properly formatted</td>
<td>No record of insurance</td>
<td>Medical</td>
<td>No SSA</td>
<td>Disabled</td>
<td>Not Working</td>
</tr>
<tr>
<td>Male</td>
<td>Primary</td>
<td>None paid</td>
<td>Improperly formatted</td>
<td>Record of insurance</td>
<td>Professional</td>
<td>SSA</td>
<td>Not Disabled</td>
<td>Working</td>
</tr>
<tr>
<td>Female</td>
<td>Primary</td>
<td>Fully paid</td>
<td>Properly formatted</td>
<td>Record of insurance</td>
<td>Dental</td>
<td>No SSA</td>
<td>Not Disabled</td>
<td>Working</td>
</tr>
<tr>
<td>Male</td>
<td>Primary</td>
<td>Partially paid</td>
<td>Improperly formatted</td>
<td>Record of insurance</td>
<td>Medical</td>
<td>SSA</td>
<td>Disabled</td>
<td>Not Working</td>
</tr>
<tr>
<td>Male</td>
<td>Secondary</td>
<td>Partially paid</td>
<td>Properly formatted</td>
<td>No record of insurance</td>
<td>Dental</td>
<td>SSA</td>
<td>Not Disabled</td>
<td>Not Working</td>
</tr>
<tr>
<td>Male</td>
<td>Primary</td>
<td>None paid</td>
<td>Properly formatted</td>
<td>No record of insurance</td>
<td>Medical</td>
<td>No SSA</td>
<td>Disabled</td>
<td>Not Working</td>
</tr>
<tr>
<td>Female</td>
<td>Secondary</td>
<td>None paid</td>
<td>Improperly formatted</td>
<td>Record of insurance</td>
<td>Dental</td>
<td>SSA</td>
<td>Not Disabled</td>
<td>Not Working</td>
</tr>
<tr>
<td>Female</td>
<td>Primary</td>
<td>Fully paid</td>
<td>Improperly formatted</td>
<td>No record of insurance</td>
<td>Medical</td>
<td>SSA</td>
<td>Not Disabled</td>
<td>Working</td>
</tr>
</tbody>
</table>
6. Test Case Coverage Analysis
ANALYSIS
# Results from Projects Using CTD

<table>
<thead>
<tr>
<th>Project</th>
<th>Original Test Case Count</th>
<th>CTD Test Case Count</th>
<th>Results &amp; Impact</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>667</td>
<td>230</td>
<td>Identified 23 missing tests and reduced number of test cases by 65%</td>
</tr>
<tr>
<td>B</td>
<td>27</td>
<td>112</td>
<td>Original test suite only had ~30% test coverage of defined test requirements. The increase of test cases closed our testing gaps.</td>
</tr>
<tr>
<td>C</td>
<td>228</td>
<td>17</td>
<td>Reduced number of test cases by 92%</td>
</tr>
<tr>
<td>D</td>
<td>121</td>
<td>84</td>
<td>Reduced number of test cases by 30%</td>
</tr>
<tr>
<td>E</td>
<td>215</td>
<td>120</td>
<td>Reduced number of test cases by 44%</td>
</tr>
<tr>
<td>F</td>
<td>61</td>
<td>41</td>
<td>Reduced number of test cases by 33%</td>
</tr>
<tr>
<td>G</td>
<td>34</td>
<td>52</td>
<td>Original test suite only had 65% test coverage of defined test requirements. The increase of test cases closed our testing gaps.</td>
</tr>
</tbody>
</table>
CONCLUSION
Conclusion

CTD is a great test planning process to supplement your testing strategy

CTD is an easy process which can be quickly adopted by any tester

It can help you save time and money in UAT by reducing the test planning phase along with the number of manual tests that need to be executed
QUESTIONS?