Combinatorial Test Design

Aviad Zlotnick

About us

- Software Performance Analysis Review and Quality Group at IBM Haifa Research Labs
  - Code review methodology and tool support
  - Design by concern methodology
  - Concurrent testing
  - Performance analysis and optimization
  - Smart system simulation
  - Code coverage analysis
  - Test selection
  - Functional modeling and test planning

Speaker introduction

Aviad Zlotnick
Research Scientist, Haifa

Dr. Zlotnick is a Research Staff Member in the Code Optimization and Quality Technologies Department at the IBM Research Haifa Labs. He received his B.Sc. degree from the Tel Aviv University, and M.Sc., and Ph.D. degrees in computer science from The Hebrew University of Jerusalem in 1978, 1981, and 1986, respectively. From 1986 to 1989 he was a member of the Digital Mapping Laboratory at the Carnegie Mellon University. In 1989, Dr. Zlotnick joined the IBM Research Haifa Labs, where he has worked on document processing, image based parcel sorting, and Storage architecture. In 2008, he joined the Software Test, Analysis and Review group. He is an author or coauthor of over 60 patents and 13 technical papers.

Introduction Objectives

- Raise awareness of an effective method and tool for test planning
- Enable practitioners to decide when it is appropriate to use the tool
- Provide an address for consulting and further education

Topics

- Motivation
- Overview of Combinatorial Test Design and the IBM Functional Coverage Unified Solution (FoCuS)
- Demo

Topics

- Motivation
- Overview of Combinatorial Test Design and the IBM Functional Coverage Unified Solution (FoCuS)
- Demo

Motivation

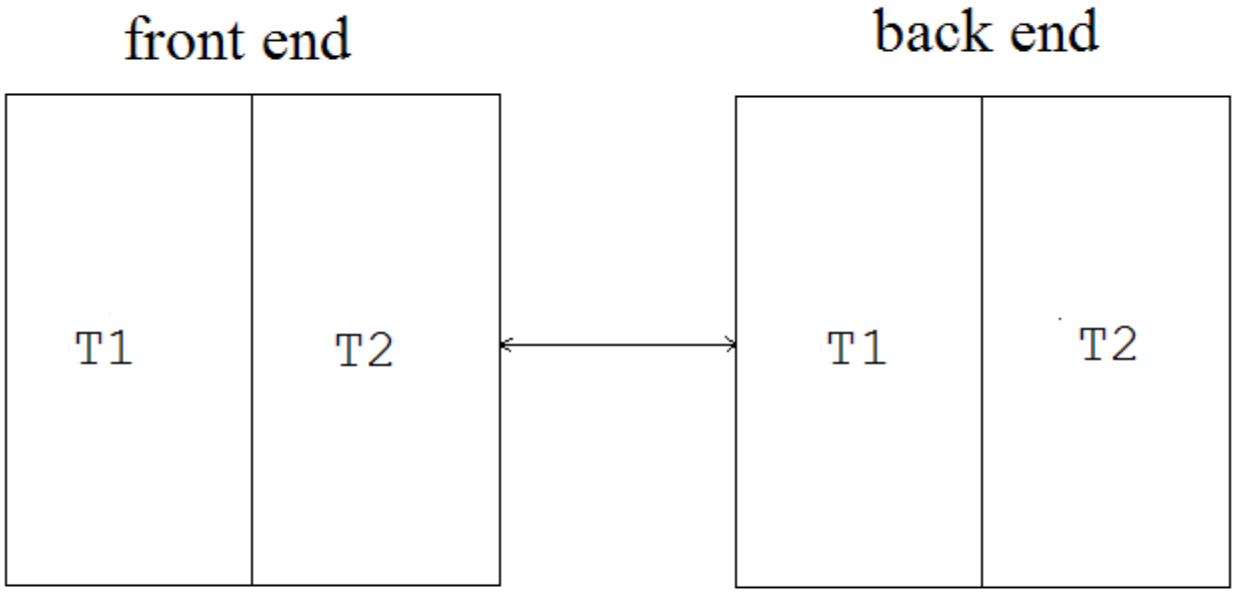
- The challenge:
  - We have too many combinations to deal with
  - We would like to use our time efficiently
  - We would like to control the risks we are taking
  - We would like to know what we tested
    - Minimize omissions

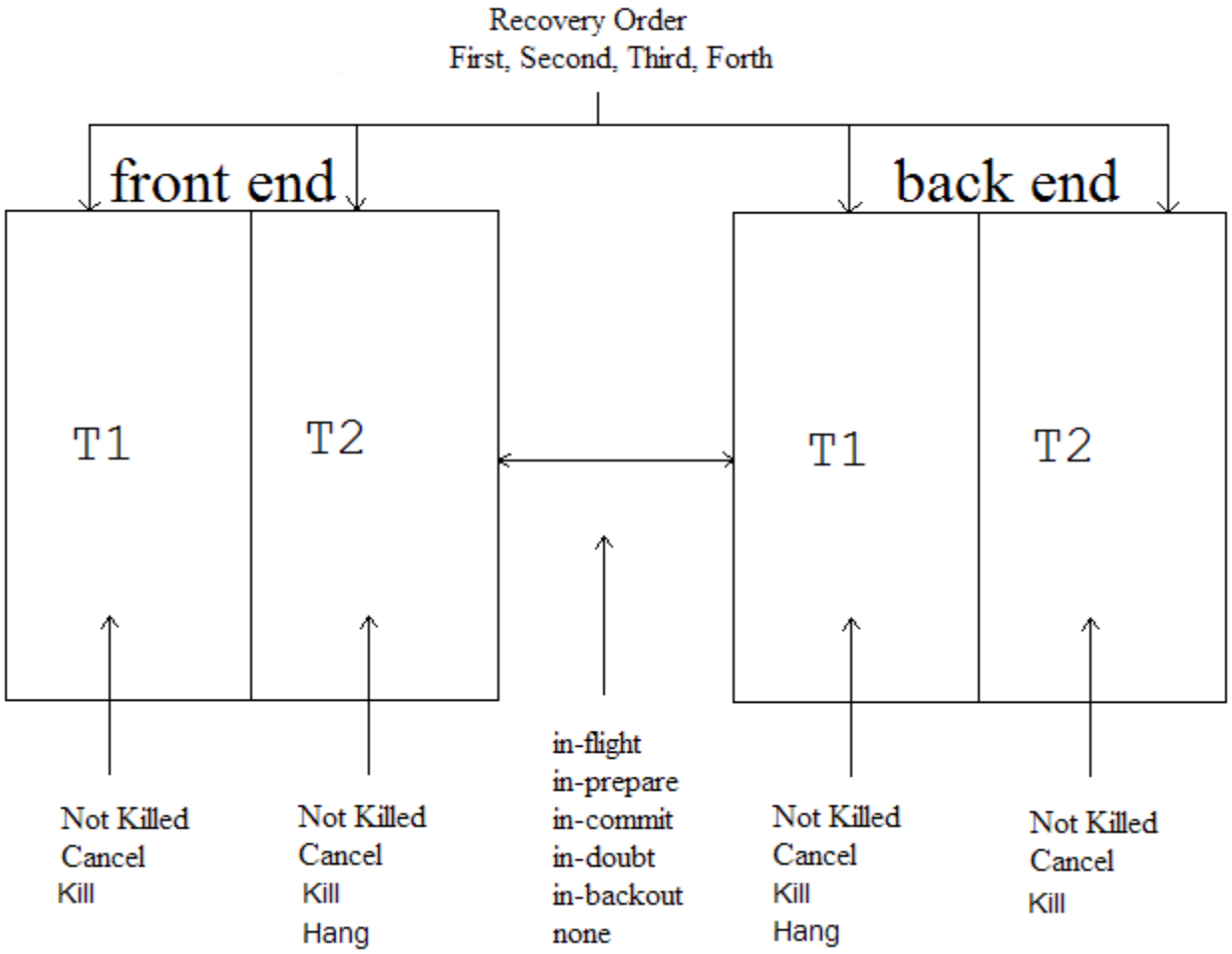
- A solution: Combinatorial Test Design (CTD)
  - Systematic planning of tests
  - Maximizes the value of each tested scenario
    - Significant reduction in the number of tests
  - Controlled risk
  - Easy to review

  → Minimizes omissions

Success stories

- For a customer in the Insurance Industry
  - The client had 15,000 tests, manually reduced to 6000 based on risk estimates
  - IBM modeled the claims adjudication process using CTD
  - IBM identified 41 test cases to perform system test with better coverage
- For a customer in the Telecommunication Industry
  - IBM reverse-engineered the model present in 117 hand-written test cases
  - Concluded that these tests could be replaced by 12 test cases
- IBM internal – System recovery
  - The test team suggested ~50 tests
  - After holes were found and a model was created, there were ~7,800 tests
  - CTD suggested only 17
  - Out of the 17 tests, 14 revealed unknown defects
  - A total of 20 new defects identified
  - No more outages for over two years





Topics

- Motivation
- Overview of Combinatorial Test Design and the IBM Functional Coverage Unified Solution (FoCuS)
- Demo

The Cartesian Product

- The Cartesian product of two sets *X* and *Y*, denoted *X* × *Y*, is the set of all possible ordered pairs whose first component is a member of *X* and whose second component is a member of *Y*.

- For example, let X be {Ace, 2, 3, …, 9, 10, Jack, Queen, King} and Y be {Diamond, Heart, Club, Spade}, then *X* × *Y* is the 52-element set of all possible playing cards.

- Thank you, Wikipedia!

- 52 = 13 * 4

- Adding a third set, e.g., Z = {Deck1, Deck2, Deck3}, we have *X* × *Y* × Z, with 13 * 4 * 3 = 156 elements.

- And so on.

Toy Example – Online Shopping System

- Parameters:
  - Availability
  - Payment Method
  - Carrier
  - Delivery Schedule
  - Export Control

- The parameters are the sources of variability in the system. We also call them variables or attributes.

Toy Example – Online Shopping System – cont.

Availability	Payment	Carrier	Delivery Schedule	Export Control
Available	Credit	Mail	One Day	True
Not in Stock	Paypal	UPS	2-5 Working Days	False
Discontinued	Gift Voucher	Fedex	6-10 Working Days	
No Such Product			Over 10 Working Days	

4 x 3 x 3 x 4 x 2 = 288 combinations

Do we really need to test all combinations?

Levels of interaction

- Suppose there is a bug, and Credit does not work well with One Day delivery
- Any combination that includes Credit and a One Day delivery will expose that bug
  - There are 24 such combinations
- Suppose Credit does not work well with a One Day delivery, but only with Fedex
- Any combination that includes Credit, a One Day delivery, and Fedex will expose that bug
  - There are 8 such combinations
- We call the first case a level two interaction, and the second case a level three interaction

Do we really need to test all combinations?

The root cause analysis of many bugs shows they depend on a value of one variable (20%-68%)

Most defects can be discovered in tests of the interactions between the values of two variables (65-97%)

Table 1. Number of variables involved in triggering software faults

Vars	Medical Devices	Browser	Server	NASA GSFC	Network Security
1	66	29	42	68	20
2	97	76	70	93	65
3	99	95	89	98	90
4	100	97	96	100	98
5		99	96		100
6		100	100		

- Source http://csrc.nist.gov/groups/SNS/acts/ftfi.html

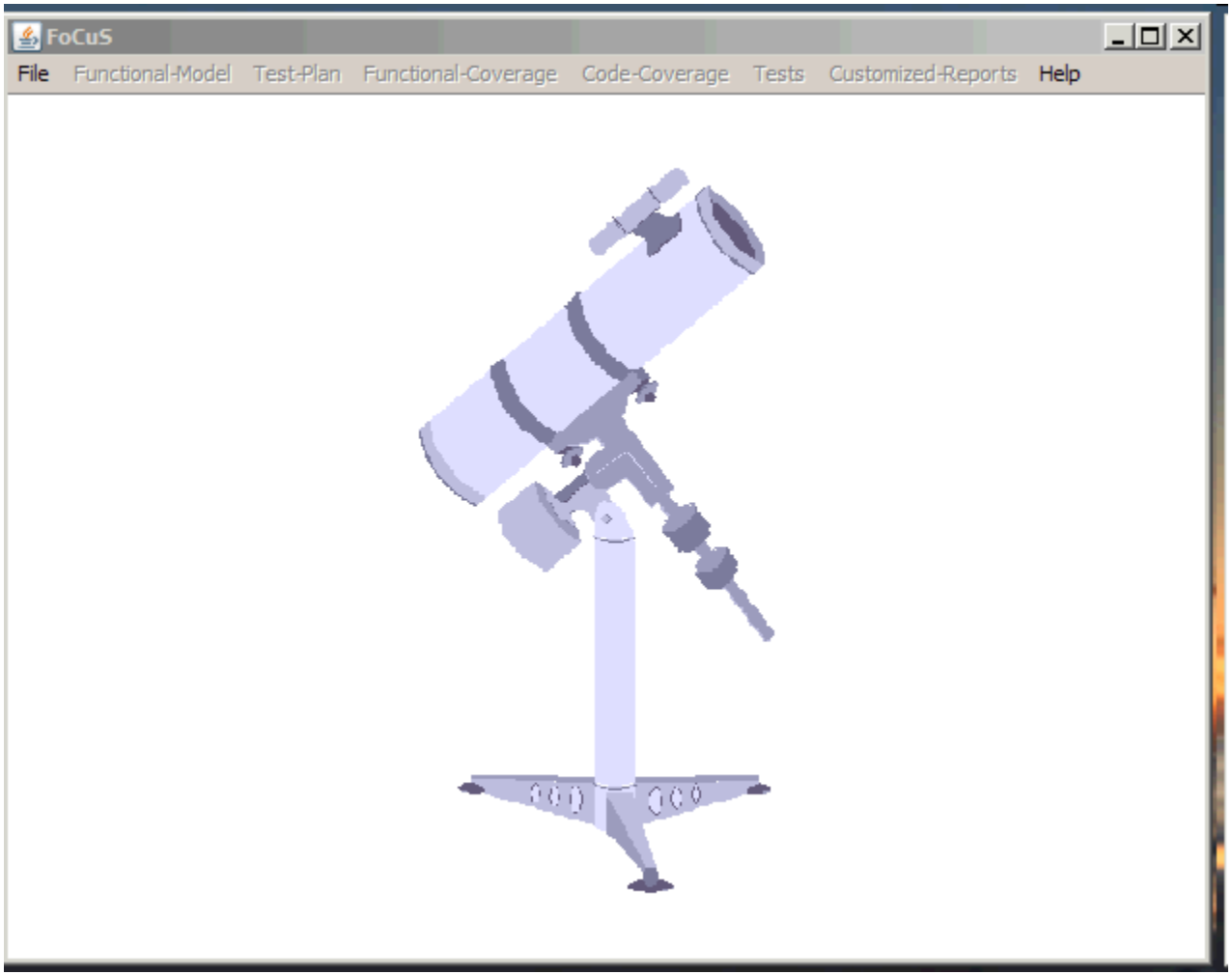
Coverage of Interactions

- Let's take interaction level 2 for example:
  - There are 101 different pairs of values:
    - Payment = Credit, Delivery = One Day
    - Payment = Credit, Delivery = 2-5 Days
    - …
    - Availability = Available, Delivery = One Day
    - …
  - A given test plan covers x% of interaction level 2 if it covers x% of these 101 pairs
  - 100% pairwise coverage means that each pair appears at least once
  - A test plan that gives 100% pairwise coverage will reveal all defects that result from an interaction level of 2 (expected 65-97% of the defects)

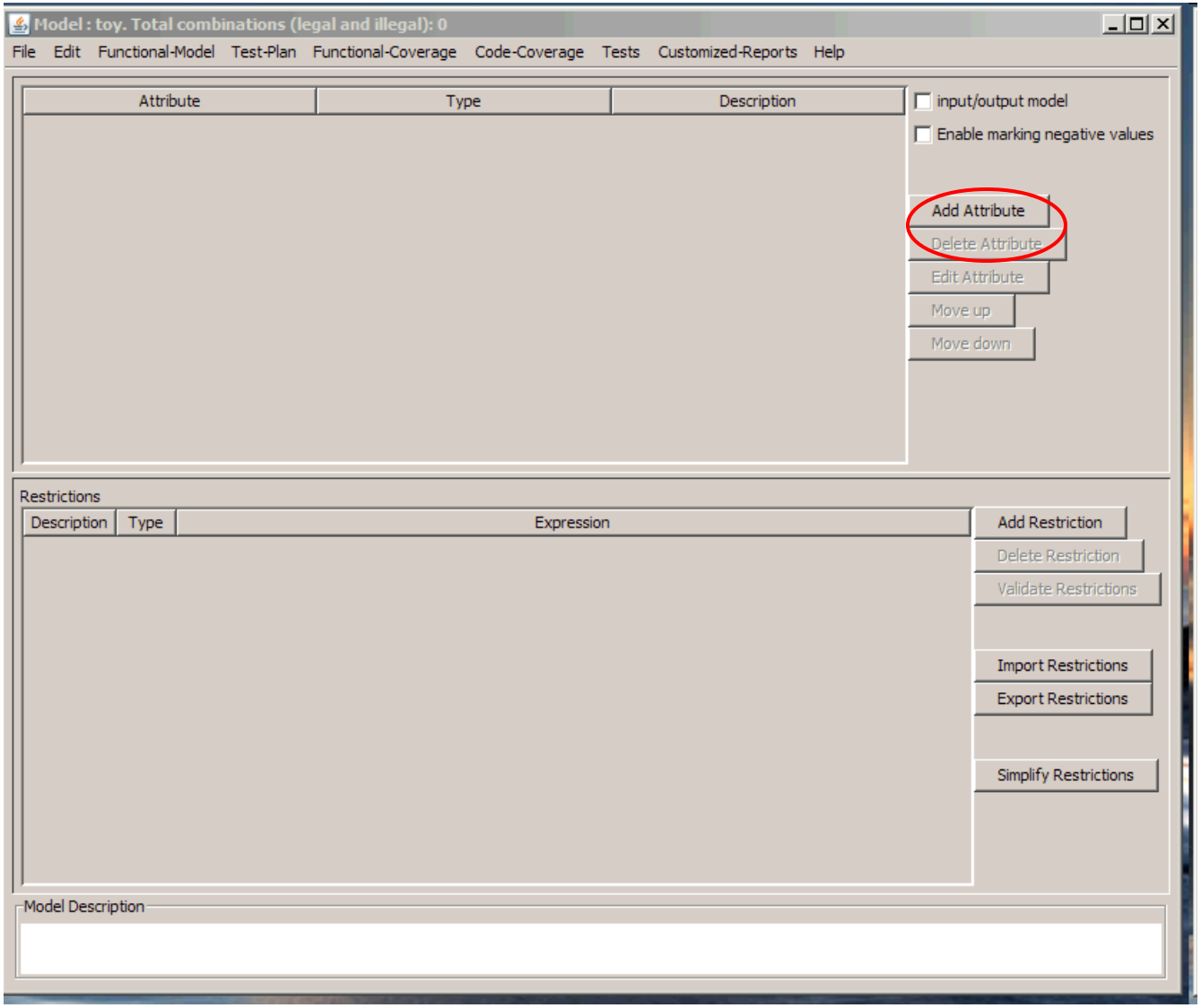
Combinatorial Test Design (CTD)

- To balance cost and risk, we select a subset of tests that covers all the interactions of variables at some level of interaction (pairs, three-way, etc.)
- A combinatorial test design (CTD) algorithm finds a small test plan that covers 100% of a given interaction level

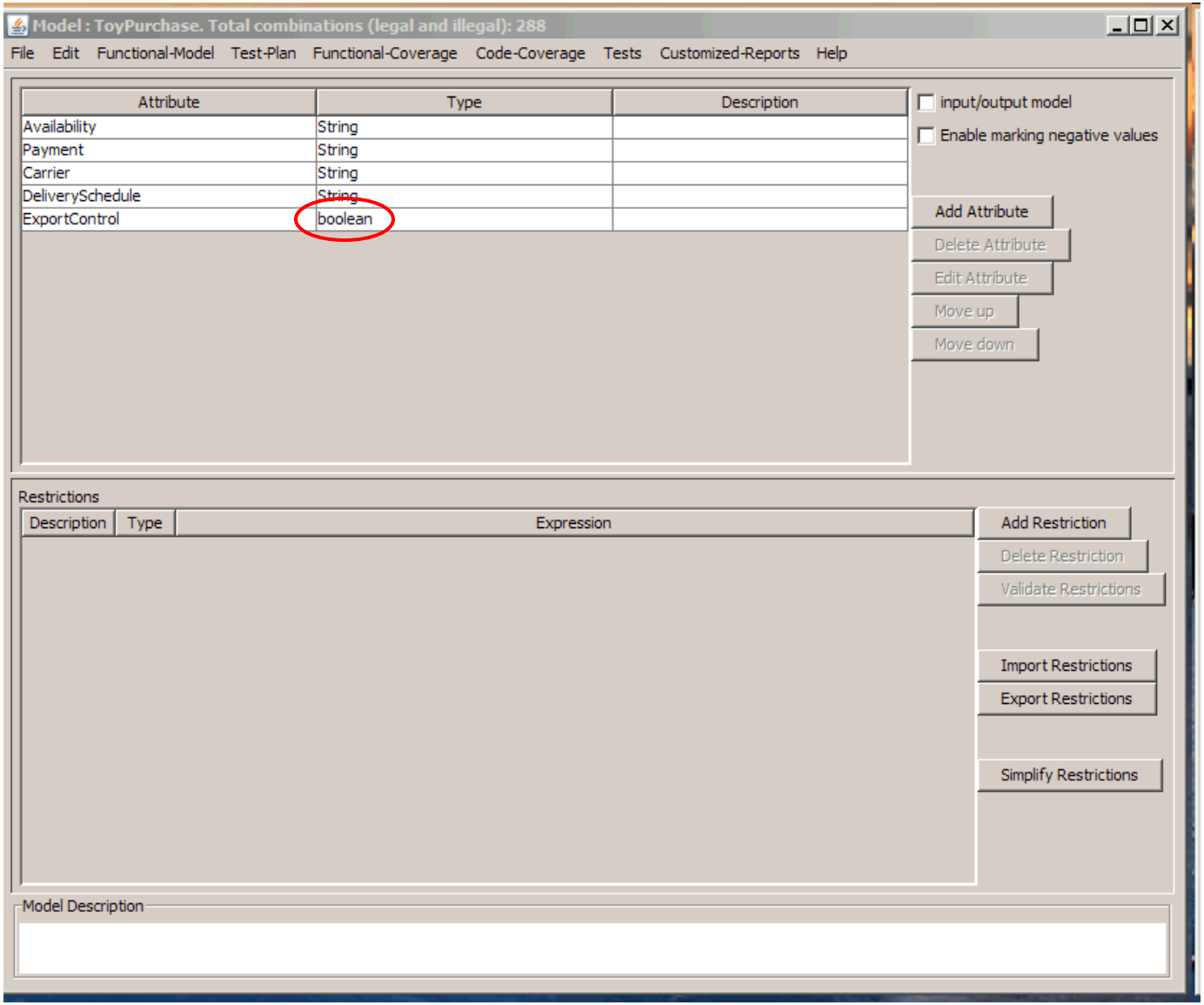
The IBM Functional Coverage Unified Solution (FoCuS) - welcome screen



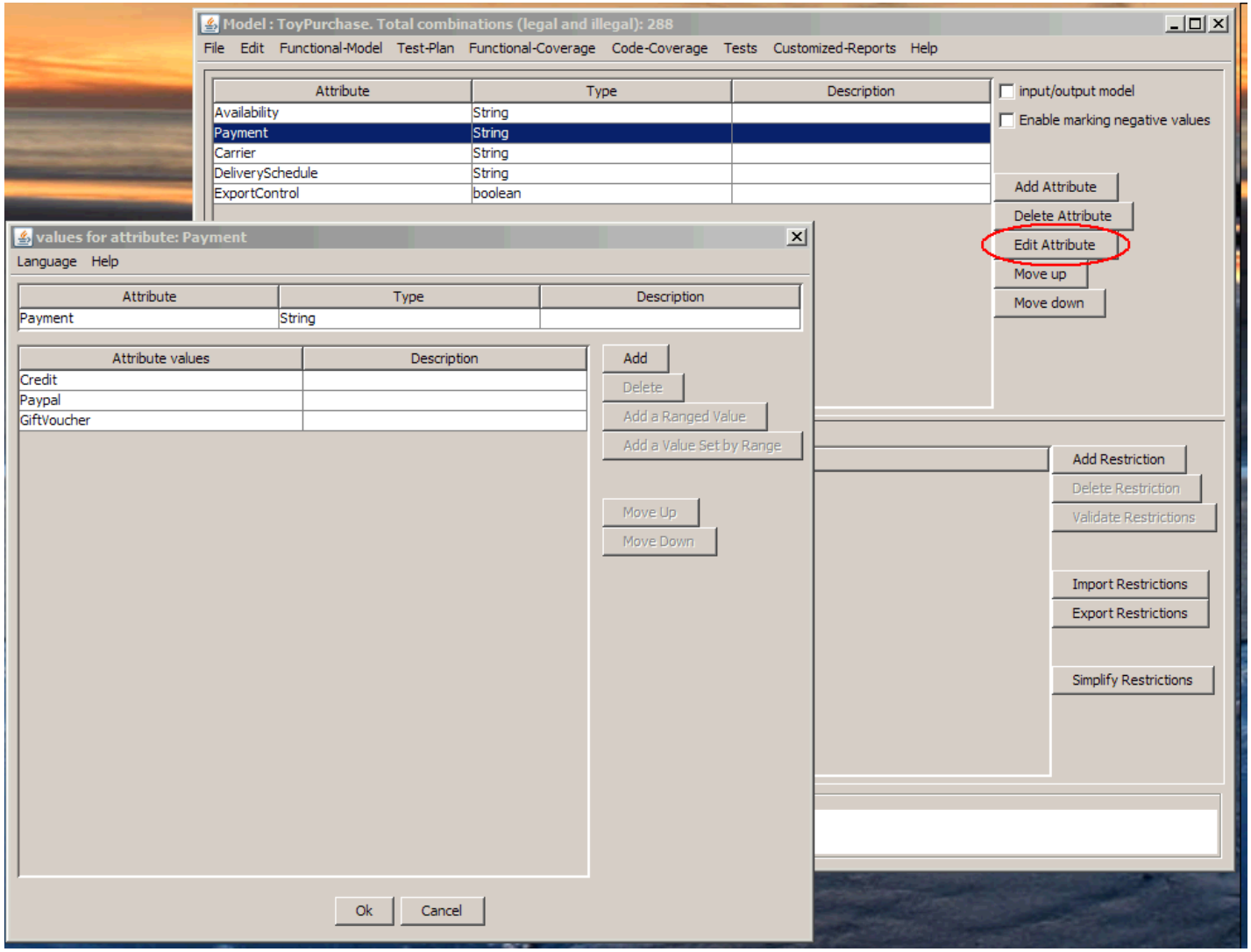
New model



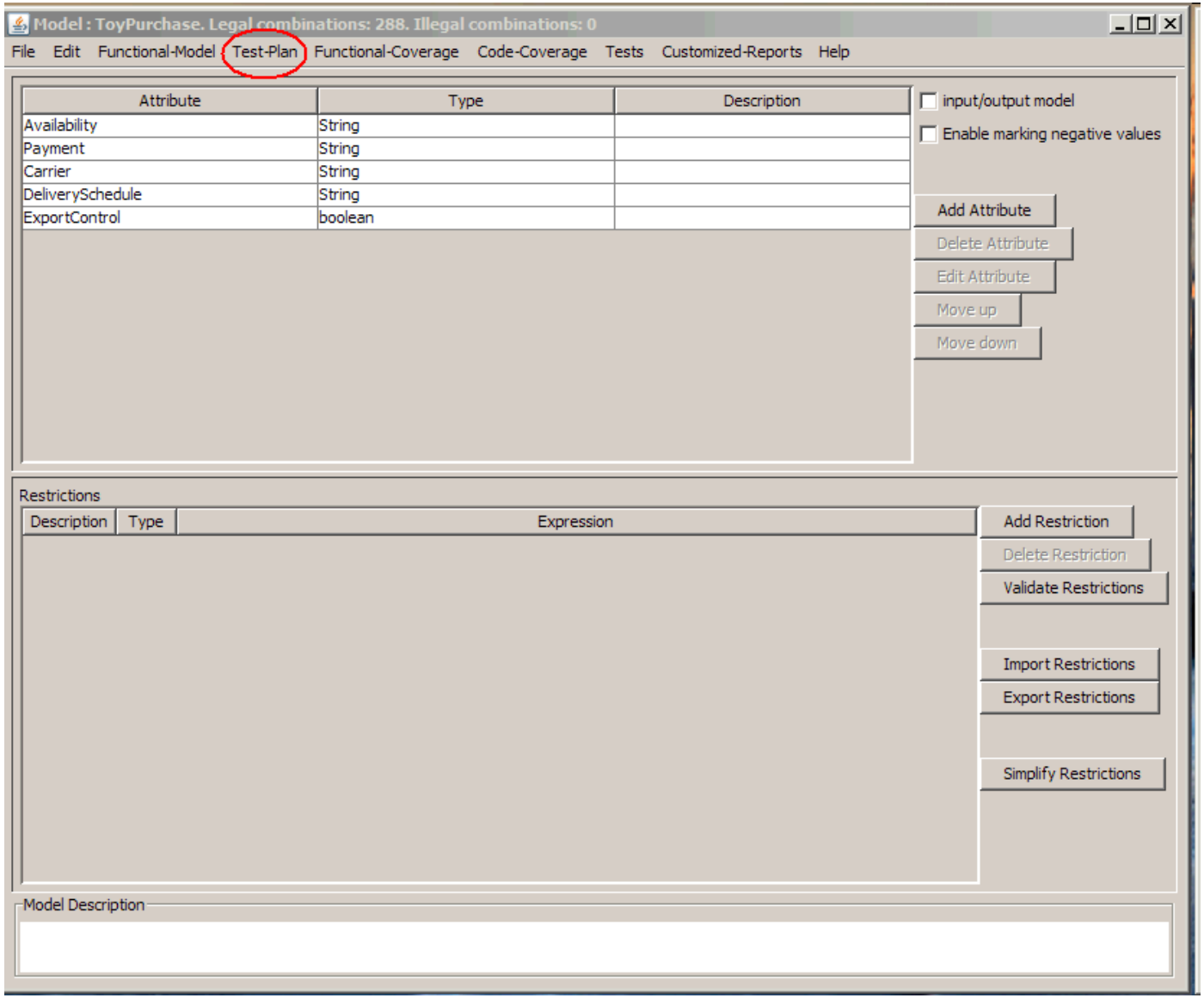
Model



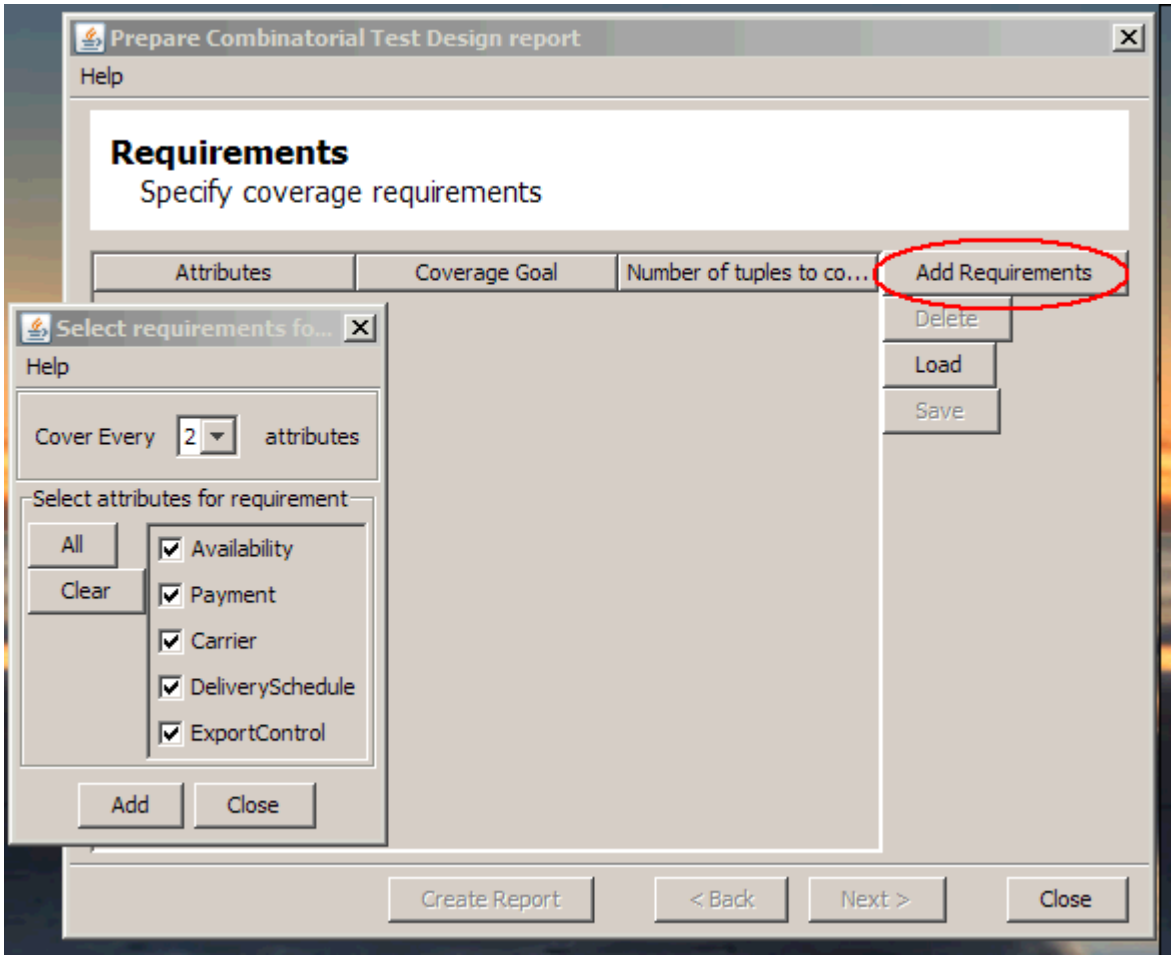
Edit attribute



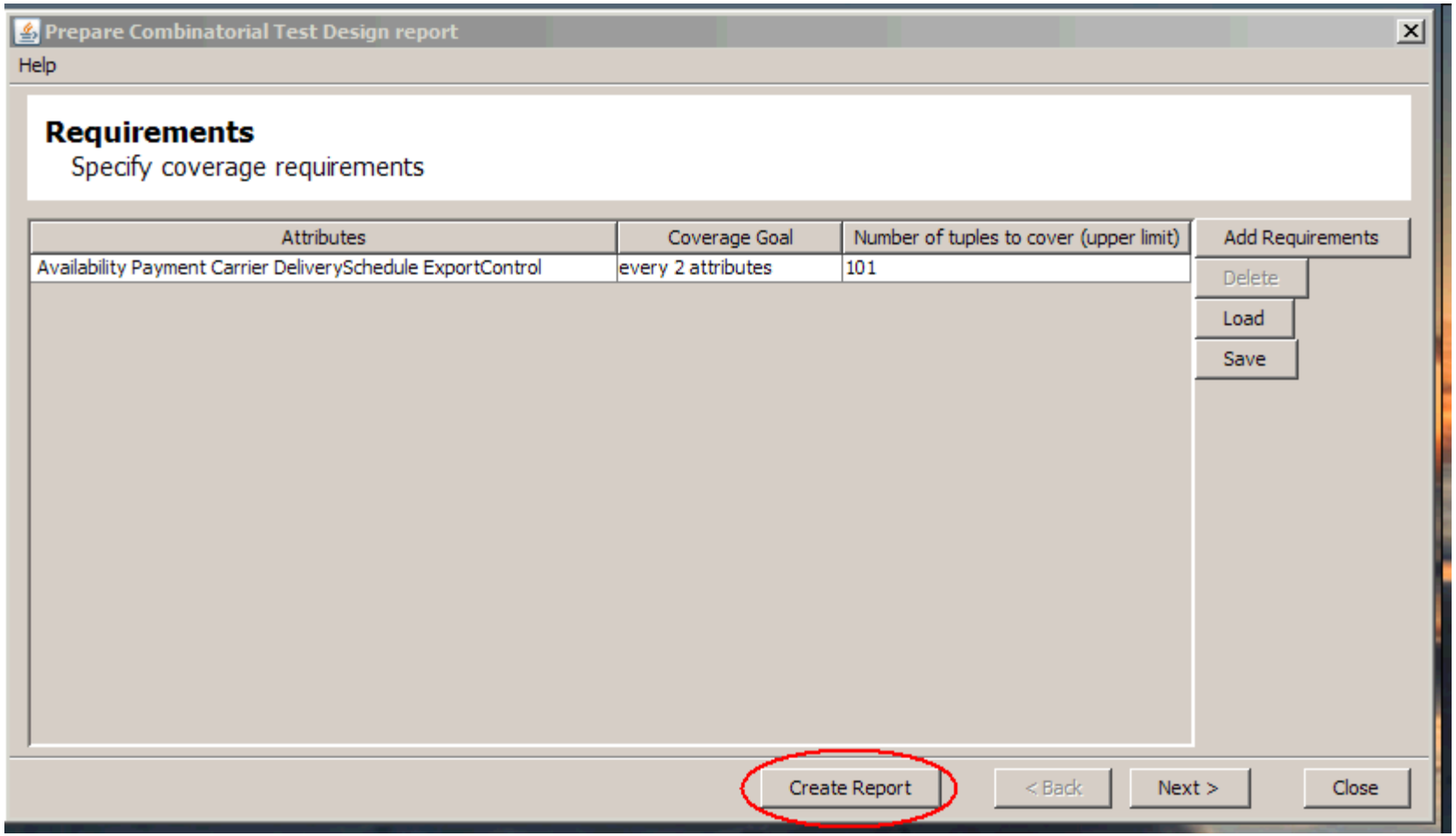
Test Planning



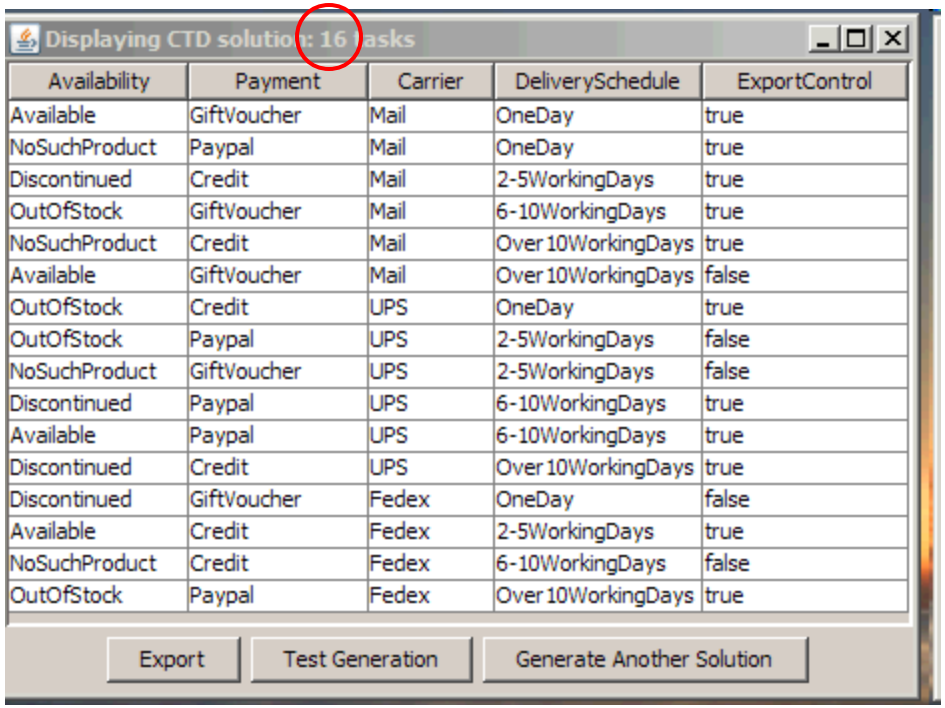
Interaction level



Create report



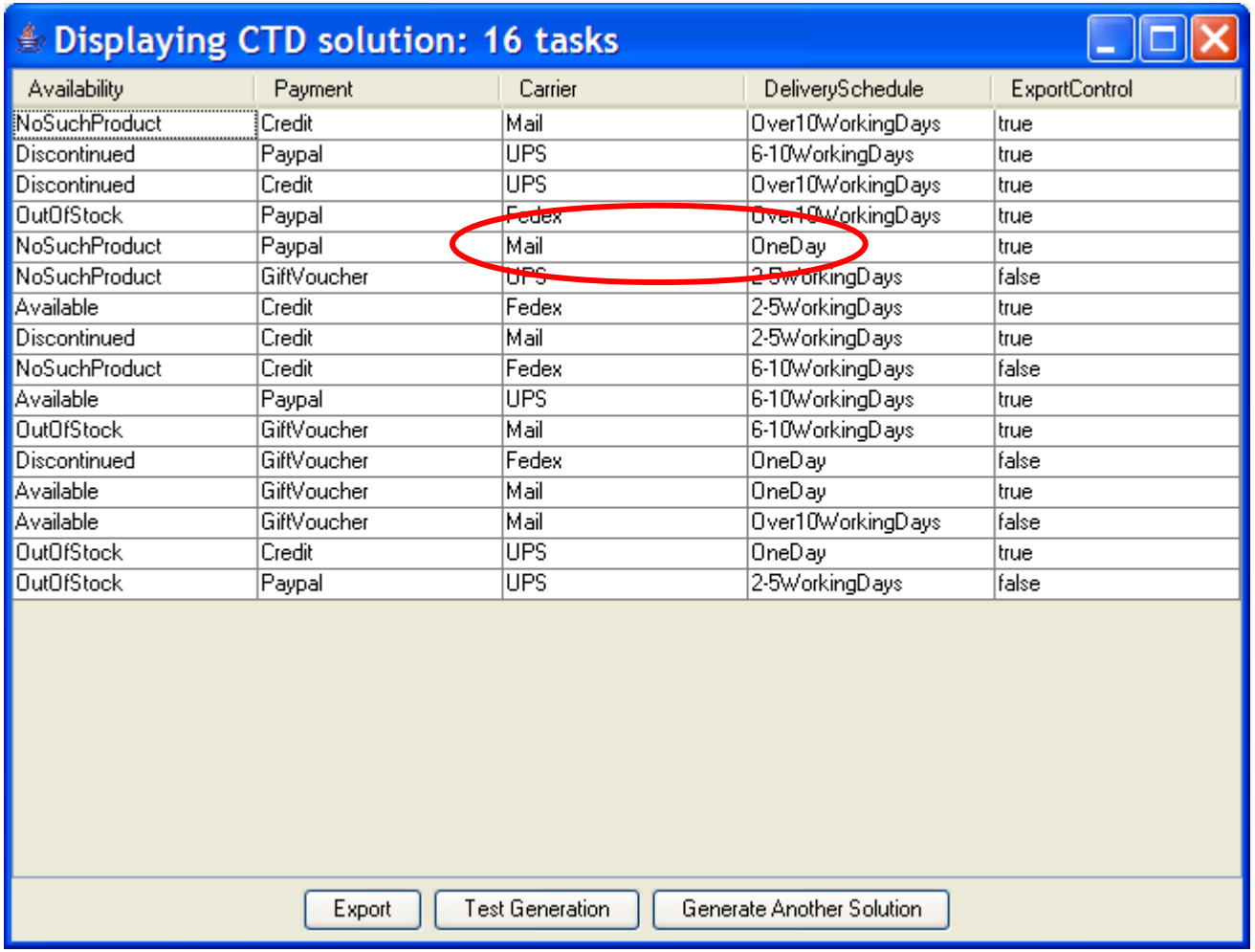
Complete pairwise coverage (one of many)



Test Plans vs. Actual Tests

- The CTD tool generates a test plan, not actual tests
- Extracting actual tests from the generated test plan may be a laborious task – generate data, generate test environments, etc.

Complete pairwise coverage



Restrictions

Why do we need restrictions?

- Impossible or irrelevant combinations, for example:
  - Mail Carrier with One Day Delivery Schedule
  - Fedex Carrier with Over 10 Working Days Delivery Schedule
  - and more..

- Naturally we cannot create and run actual tests that contain impossible combinations, so we need to state in advance what should be excluded

Why not just skip tests that contain impossible/irrelevant combinations?

- Assume we skip all tests with mail carrier in one day:

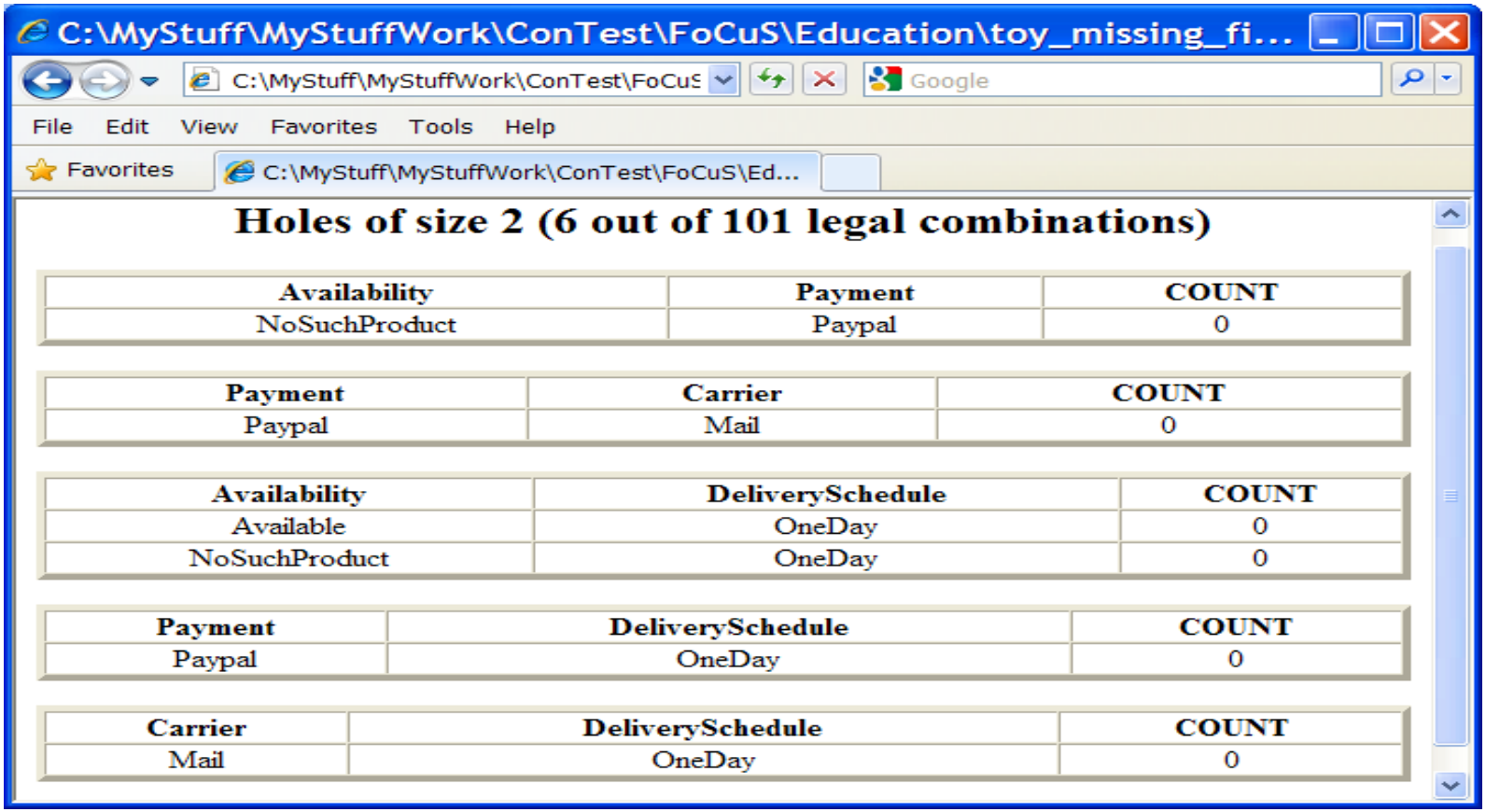
Displaying CTD solution: 16 tasks

Availability	Payment	Carrier	DeliverySchedule	ExportControl
NoSuchProduct	Credit	Mail	Over10WorkingDays	true
Discontinued	Paypal	UPS	6-10WorkingDays	true
Discontinued	Credit	UPS	Over10WorkingDays	true
OutOfStock	Paypal	Fedex	Over10WorkingDays	true
NoSuchProduct	Paypal	Mail	OneDay	true
NoSuchProduct	GiftVoucher	UPS	2-5WorkingDays	false
Available	Credit	Fedex	2-5WorkingDays	true
Discontinued	Credit	Mail	2-5WorkingDays	true
NoSuchProduct	Credit	Fedex	6-10WorkingDays	false
Available	Paypal	UPS	6-10WorkingDays	true
OutOfStock	GiftVoucher	Mail	6-10WorkingDays	true
Discontinued	GiftVoucher	Fedex	OneDay	false
Available	GiftVoucher	Mail	OneDay	true
Available	GiftVoucher	Mail	Over10WorkingDays	false
OutOfStock	Credit	UPS	OneDay	true
OutOfStock	Paypal	UPS	2-5WorkingDays	false

Export | Test Generation | Generate Another Solution

Why not just skip tests that contain impossible/irrelevant combinations?

- Now let's run hole analysis on the test plan without the two skipped tests:



Holes of size 2 (6 out of 101 legal combinations)

Availability	Payment	COUNT
NoSuchProduct	Paypal	0

Payment	Carrier	COUNT
Paypal	Mail	0

Availability	DeliverySchedule	COUNT
Available	OneDay	0
NoSuchProduct	OneDay	0

Payment	DeliverySchedule	COUNT
Paypal	OneDay	0

Carrier	DeliverySchedule	COUNT
Mail	OneDay	0

5 legal pairs are now uncovered, in addition to the excluded pair!

Why not just skip tests that contain impossible/irrelevant combinations?

- Each test in the CTD test plan may cover multiple unique legal combinations

- By skipping a test we will lose all these combinations, and no longer have 100% interaction coverage

What are restrictions?

- Restrictions are rules that determine which combinations are included and which are excluded from the model
- Combinations that are excluded from the model will never appear in the test plan
  - So it is important to define them carefully
  - FoCuS enables viewing the excluded combinations

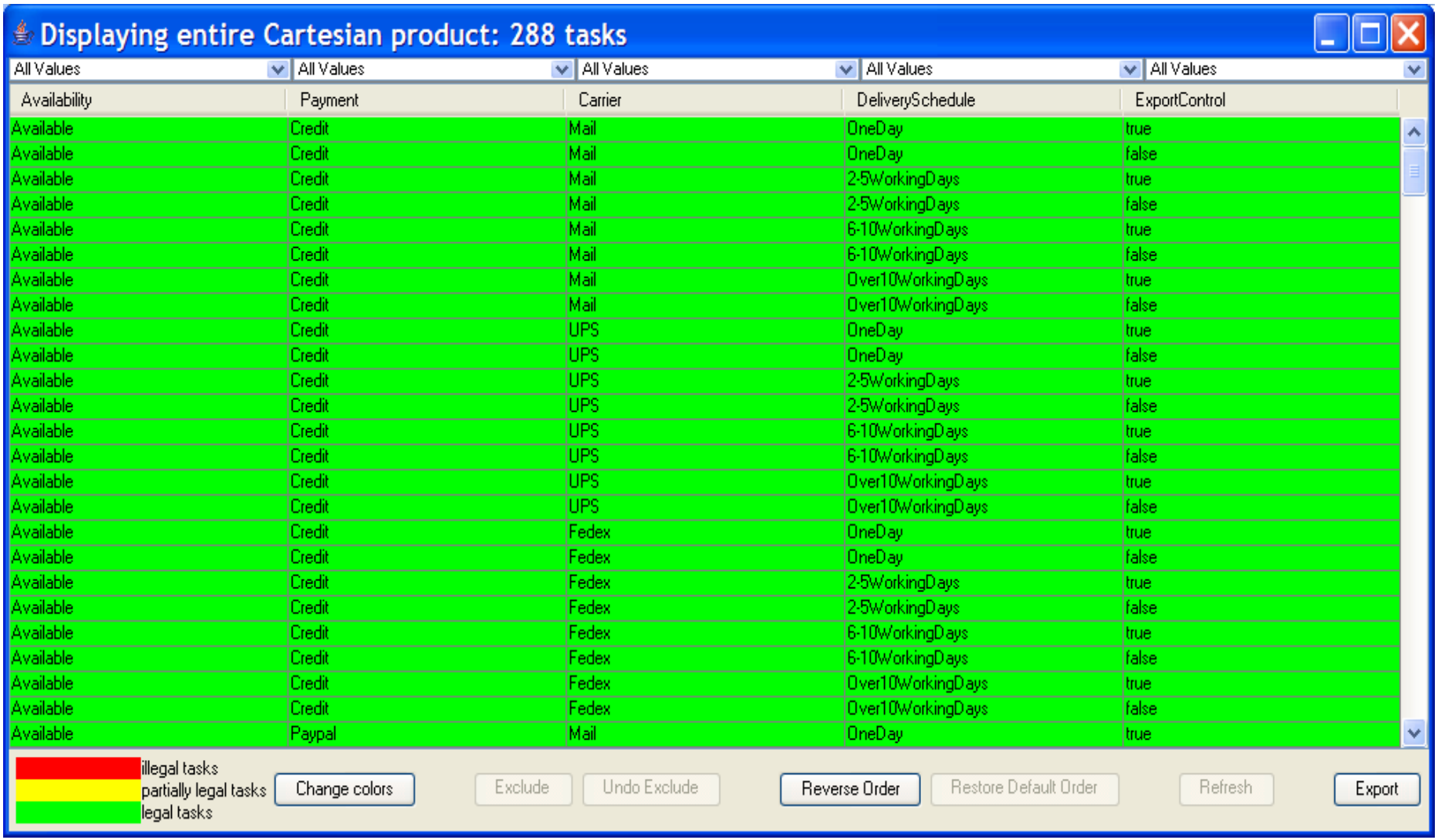
How do I define restrictions in FoCuS?

By marking and excluding combinations in the Cartesian product report

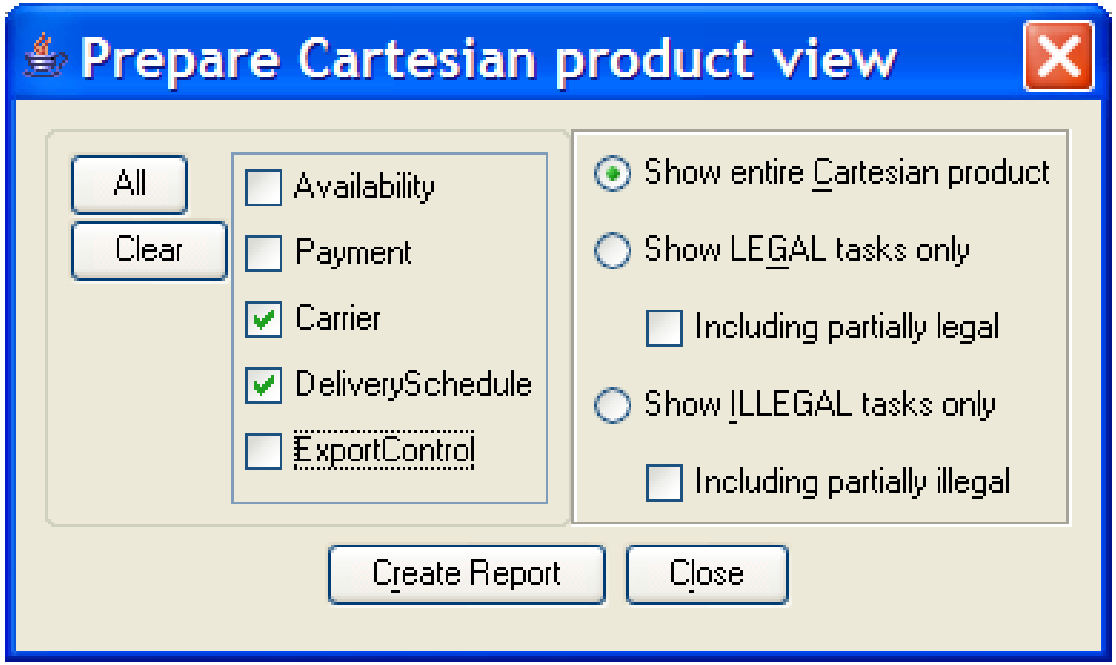
Or

By writing explicit conditions on what combinations should be included/excluded (advanced)

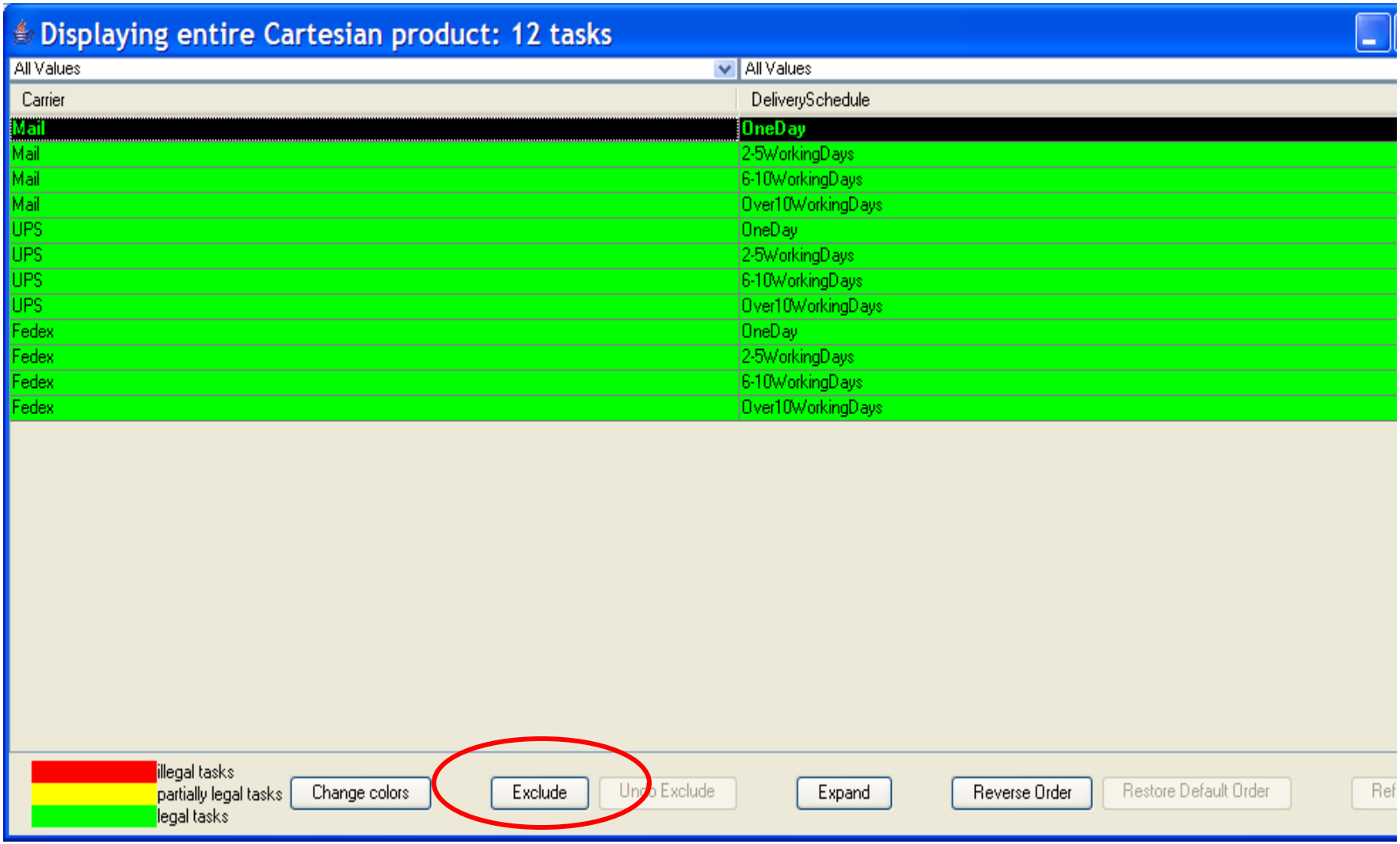
Cartesian product report – all 288 combinations are legal



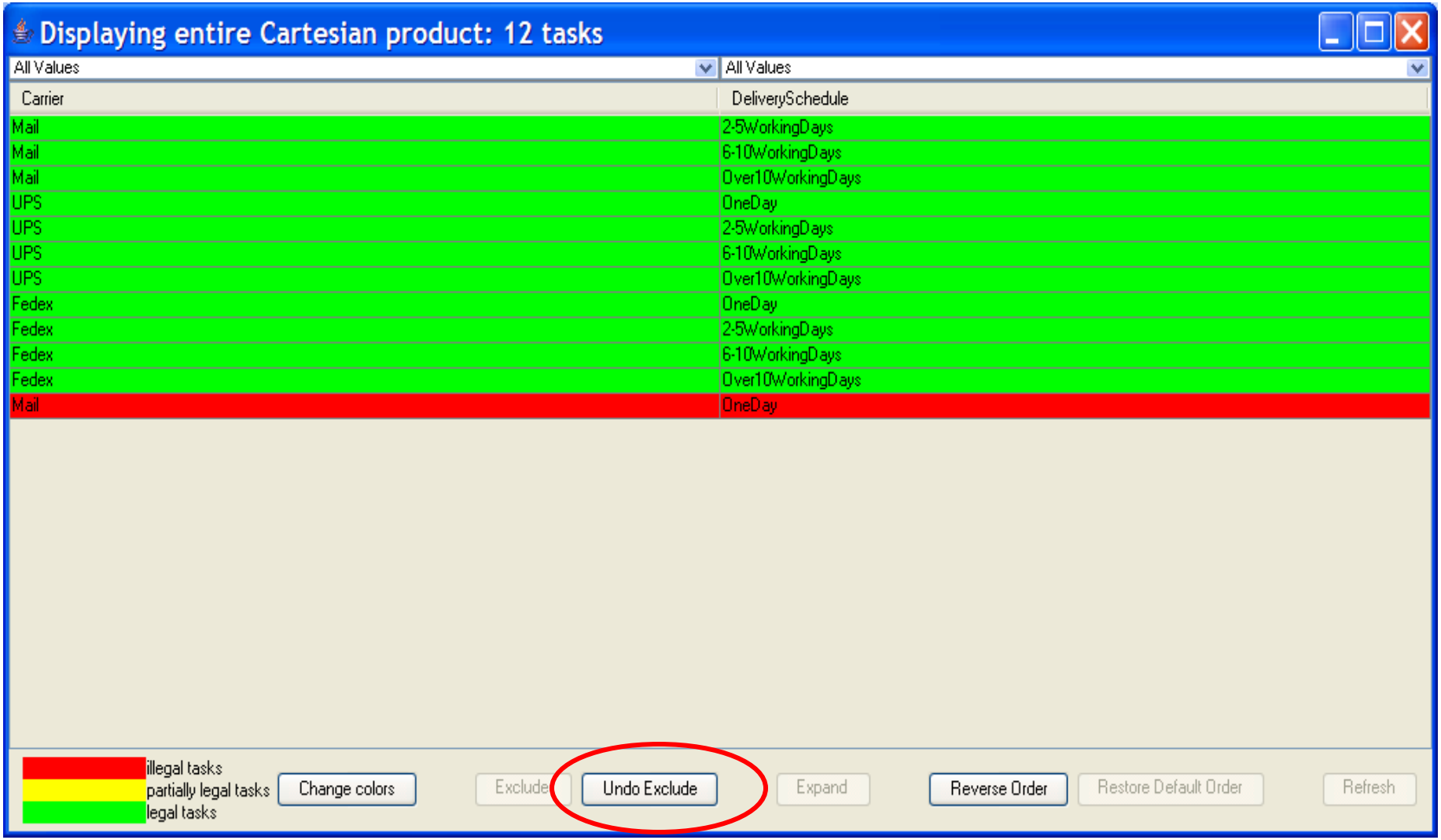
Choose to view only part of the attributes (projection)

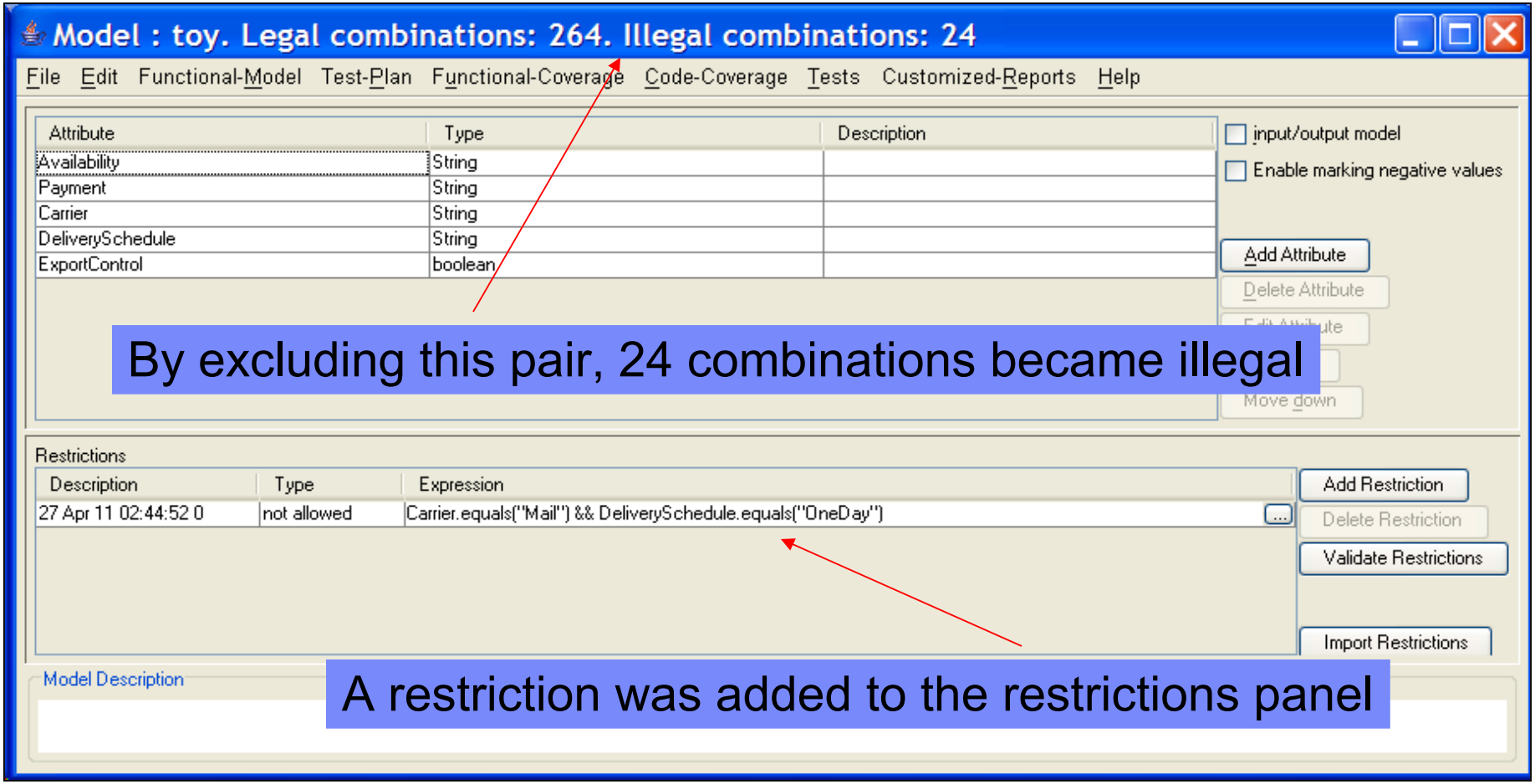


12 value pairs in the projection of the selected attributes

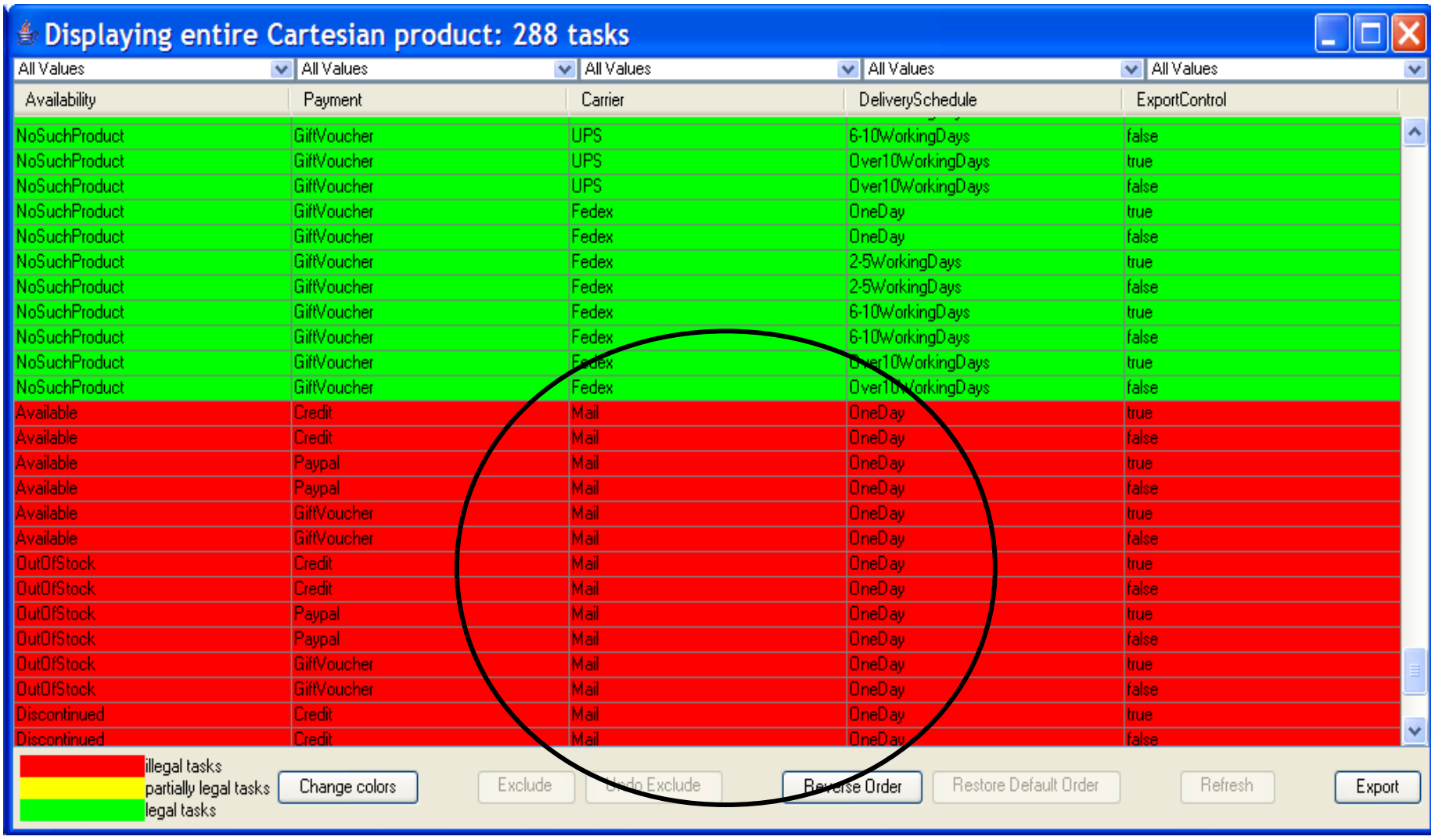


Excluding the invalid combination

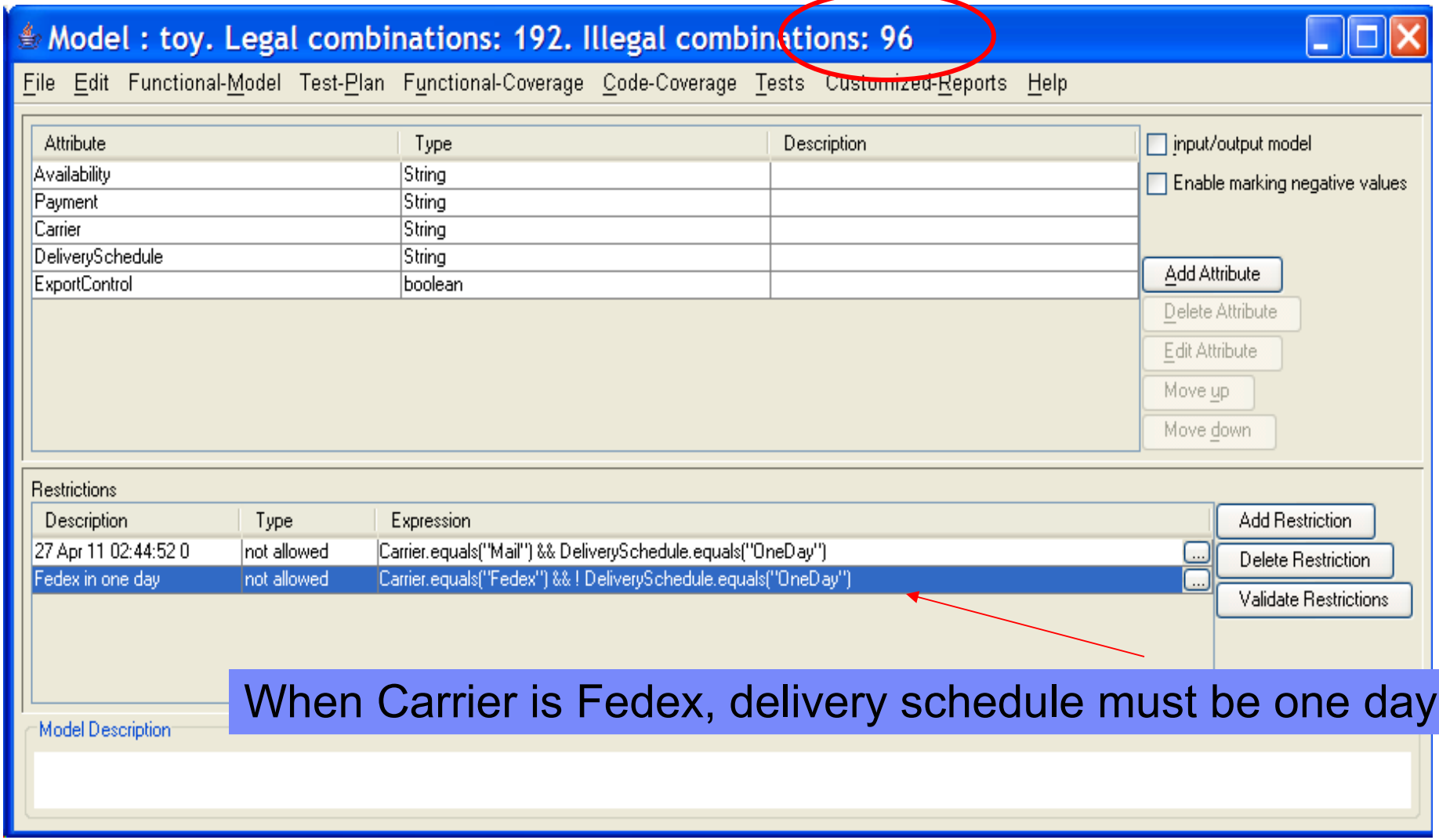




The Cartesian product displays all legal and illegal combinations



Explicitly adding a restriction to the model

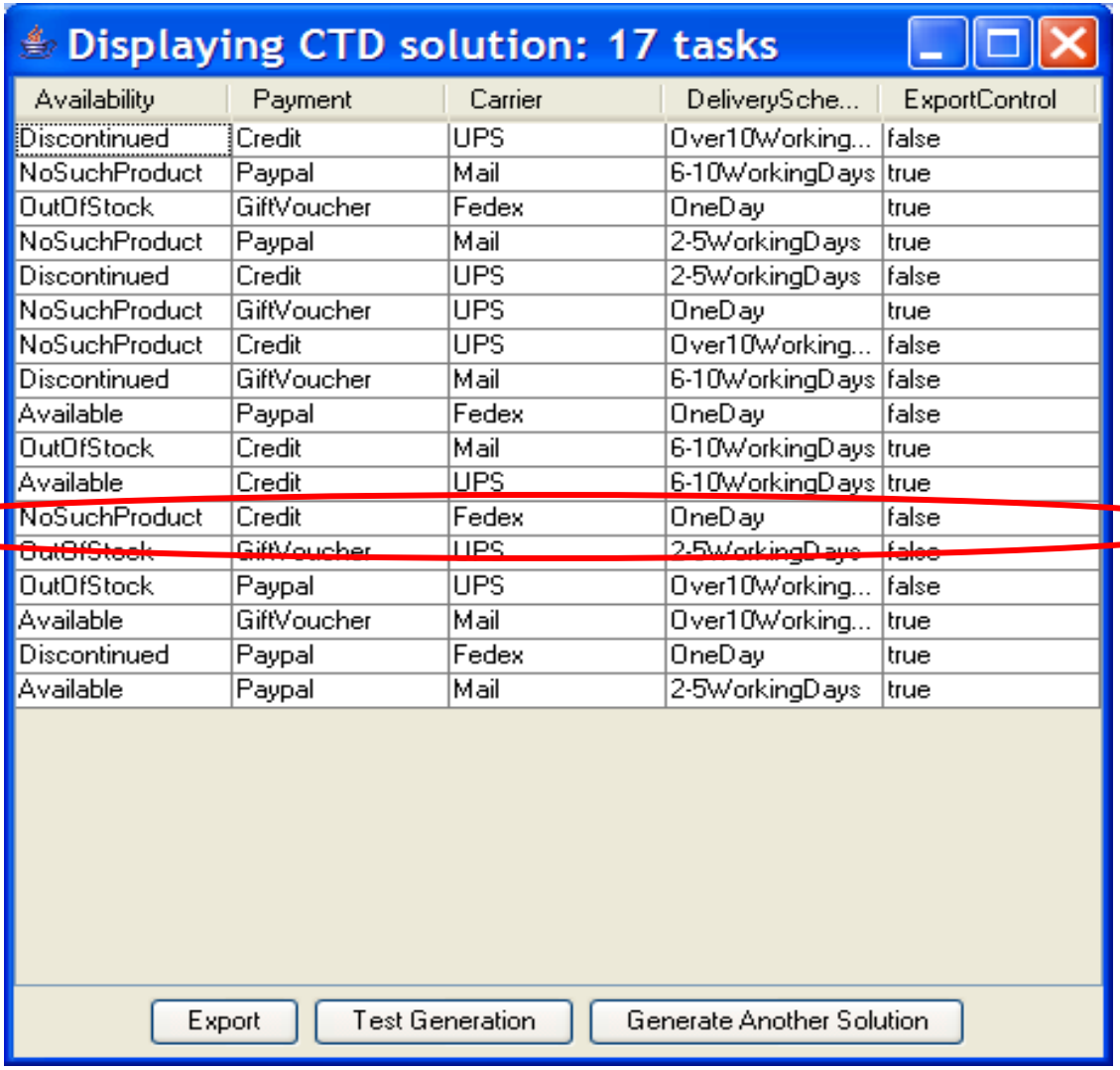


Complete pairwise coverage of the legal pairs

- Note that after adding restrictions there are less combinations to cover, but more tests in the test plan (17 instead of 16)

- This happens since some tests that previously covered many new combinations may now become illegal, and cannot be used

A closer look at the resulting test plan



Negative Testing

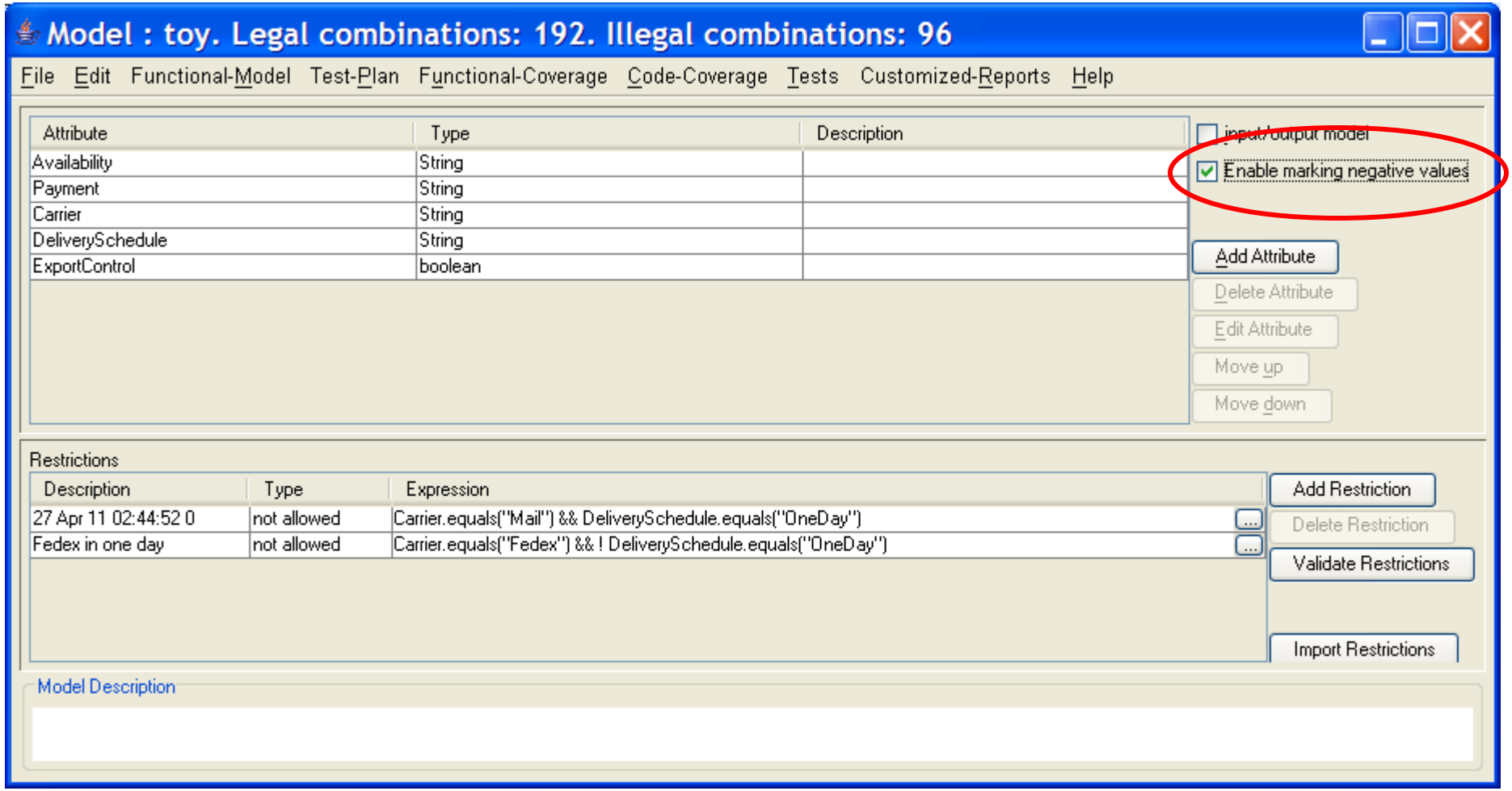
Negative Values

- When no such product exists, the test will terminate prematurely
  - The interactions between the other attributes will not be actually tested by this test
- The combination Payment=Credit, Carrier=Fedex appears only in one test, and this test is failing
  - It is covered by the test plan, but will not be reached by the executed code
- To really achieve 100% interaction coverage, negative values must be identified and considered
- FoCuS supports testing of bad paths
  - Indication of the failure values in the model
  - Creation of separate test plans for good paths and bad paths

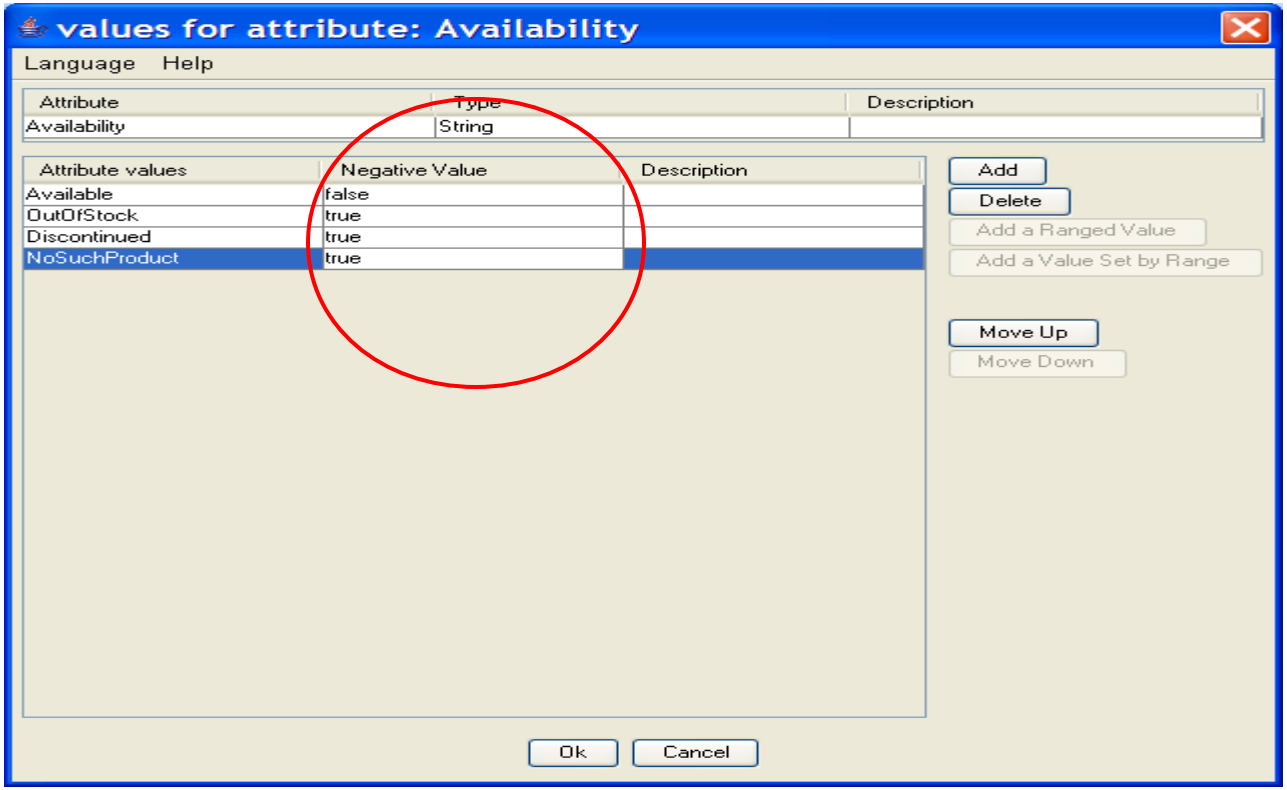
Negative Testing

- Testing of what happens when things go wrong
- There can be many ways to fail
  - Wrong inputs, unexpected conditions, unavailable resources..
- Testers tend to concentrate on the good path, and neglect the bad paths
  - Failure scenarios are less intuitive to consider
  - Bad path tests can be more difficult to implement
  - Results in incomprehensible error messages, unnecessary crashes, and chain reactions of failures..
- Especially important to consider when using CTD, as otherwise might result in false coverage of interactions

Marking negative values



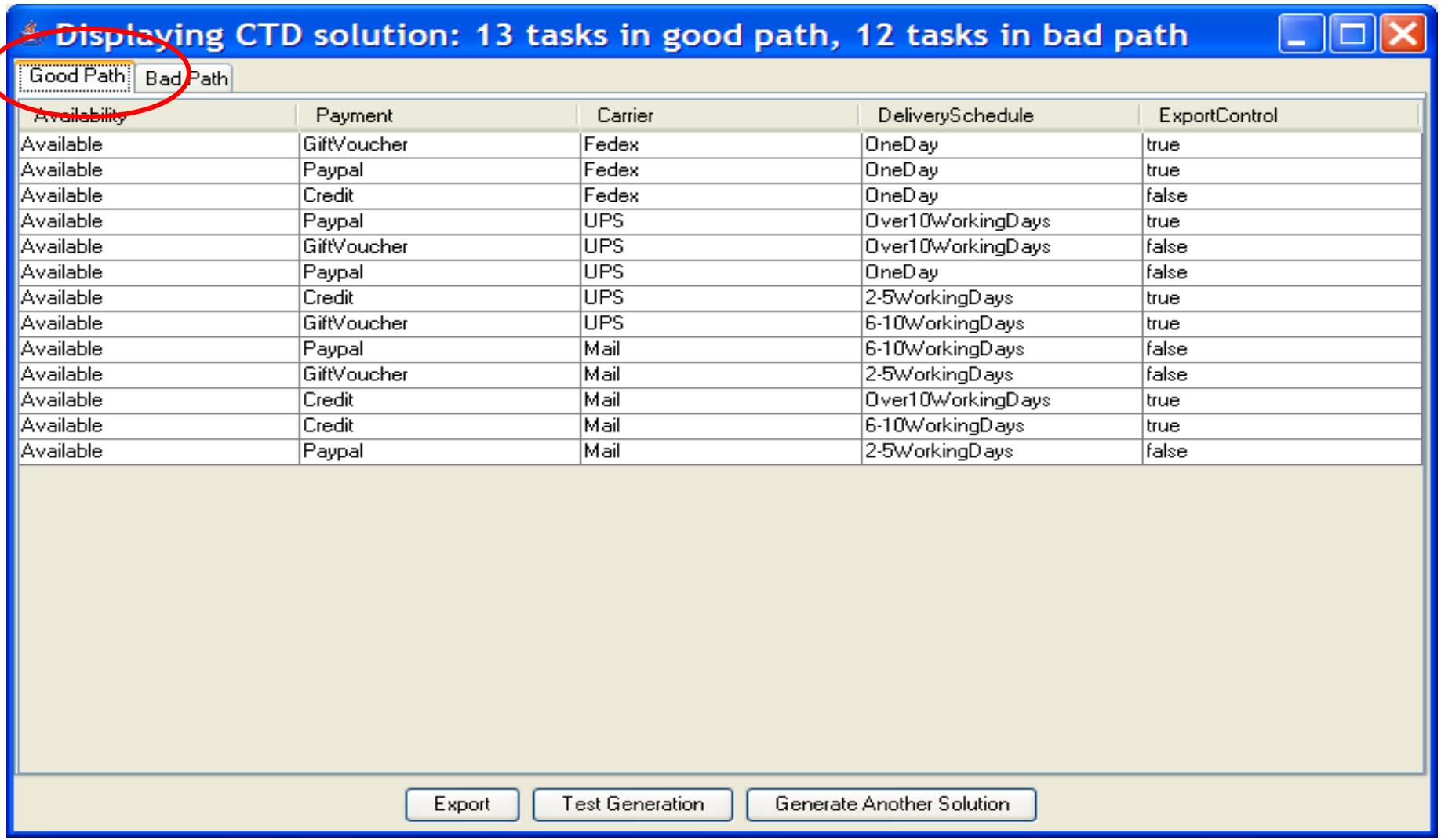
Marking negative values



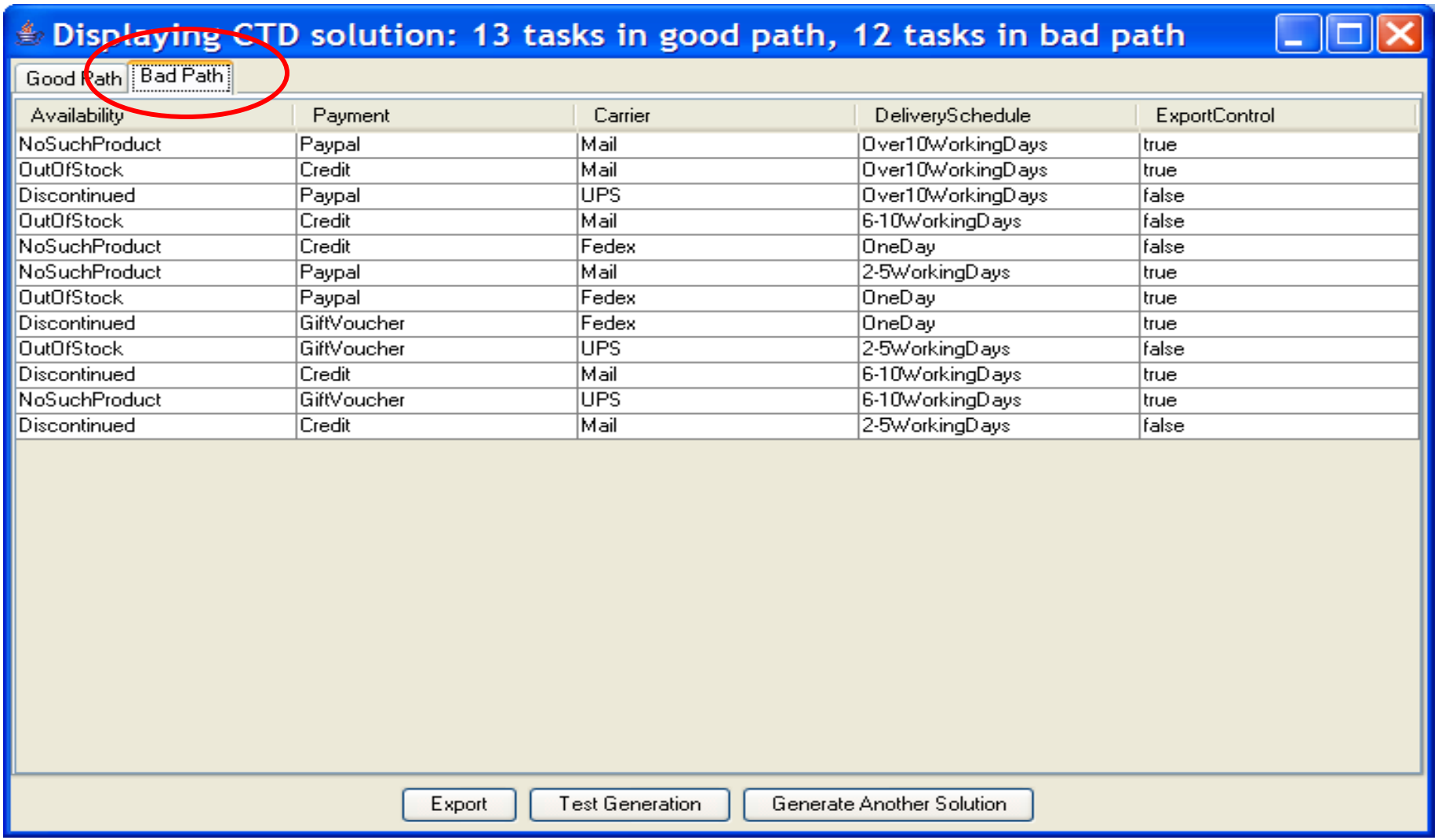
Alternatively, OutOfStock can be considered non-negative, and restricted to a late delivery

- This depends on how the specific system under test works
- Requires understanding the details of the system through interviews and documents

Good path test plan does not contain negative values



Bad path test plan contains exactly one negative value in each test



Topics

- Motivation
- Overview of Combinatorial Test Design and the IBM Functional Coverage Unified Solution (FoCuS)
- Demo

Uncovered topics

- Engagement steps
- Traces
- Hole analysis
- Test selection
- Test enhancements
- Modeling patterns
- Advanced restrictions

Contacts

- Aviad Zlotnick/Haifa
- Rachel Tzoref/Haifa
- Itai Segall1/Haifa

Summary

- A test project can be modeled with attributes, values, and restrictions
- Combinatorial Test Design lets you improve quality and reduce effort, ensuring 100% coverage of required interaction levels
- It has been applied successfully, using the IBM Functional Coverage Unified Solution (FoCuS), in many real projects
- FoCuS supports modeling, creating a test plan from scratch, selecting tests from an existing test suite, and enhancing an existing test suite
- This workshop was only an introduction. Please feel free to contact us to help you start deployment