3rd UCAAT User Conference on Advanced Automated Testing

ETSI World Class Standards

Rachid Kherrazi
UCAAT 2015

Sophia Antipolis, French Riviera
20-22 October 2015

USING TASK MODELS IN MODEL-BASED TESTING

by Rachid Kherrazi

NSPYRE

CONTENT

- Introduction to Model Based Testing
  - Why MBT?
  - What is MBT?
  - What are benefits and challenges?
- User Experience and Task Models
- From Task model to Test Model using UXSpec tool
- Case study and results

Rachid Kherrazi

Sr. Consultant @ Nspyre

Domain

Process and Product Improvement

Model Based Testing and Model Driven Engineering

iSQI Accredited Certified Model Based Tester Trainer

Also involved in this project

Anita Wierda User Experience specialist

Neda Noroozi MBT specialist

Arjan van der Meer MDE specialist

Typical issues with traditional testing

- Many problems discovered at a late stage of development process, often costly to resolve.

- In addition, they affect the planning of development processes in that they must be resolved 'In between'

➔Several pre and post release defects not found with traditional testing, which results in

➔Several delays in planning

➔With related quality/cost impact

Typical what we hear from our customers

Root Cause

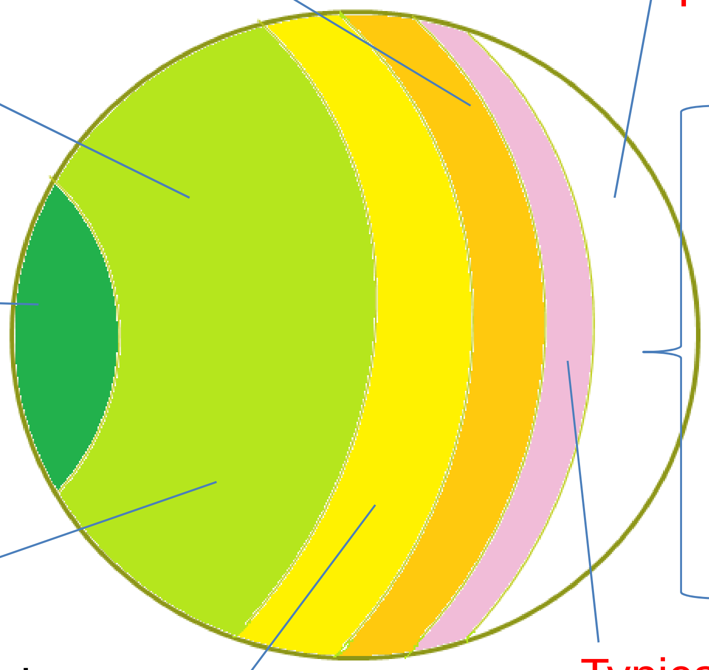
- Unit testing is insufficient
  - Single unit may work properly in isolation
  - Incorrect interaction between units may cause serious reliability failures
- System-level testing
  - Requires **model** of system behavior
  - Behavior is often reactive/nondeterministic
    - Implementation is multi-threaded or distributed
    - Thread scheduling hard to control
  - State space is typically infinite
  - Traditional FSM-based testing does not scale

MBT helps to avoid integration issues and post release defects

Manual exhaustive
exploratory testing

Manual testing
happy flows

Review
Static testing
Unit testing



Manual testing
alternative flows

Manual testing
bad weather flows

Typical uncovered
Post release defects

This is typical domain where **Model Based Testing helps**

You can try to avoid integration issues and post release defects by adding manual test case, but it will be complex will take lot of effort and finally costly

Typical uncovered
Integration issues

Increased complexity and effort of testing

Other Drivers for MBT

Our world evolves….and becomes complex

- Cyber-Physical Systems/EoT
- Machine2Machine
- Digital Transformation

➔Increased need to manage complexity

Cyber-Physical Systems

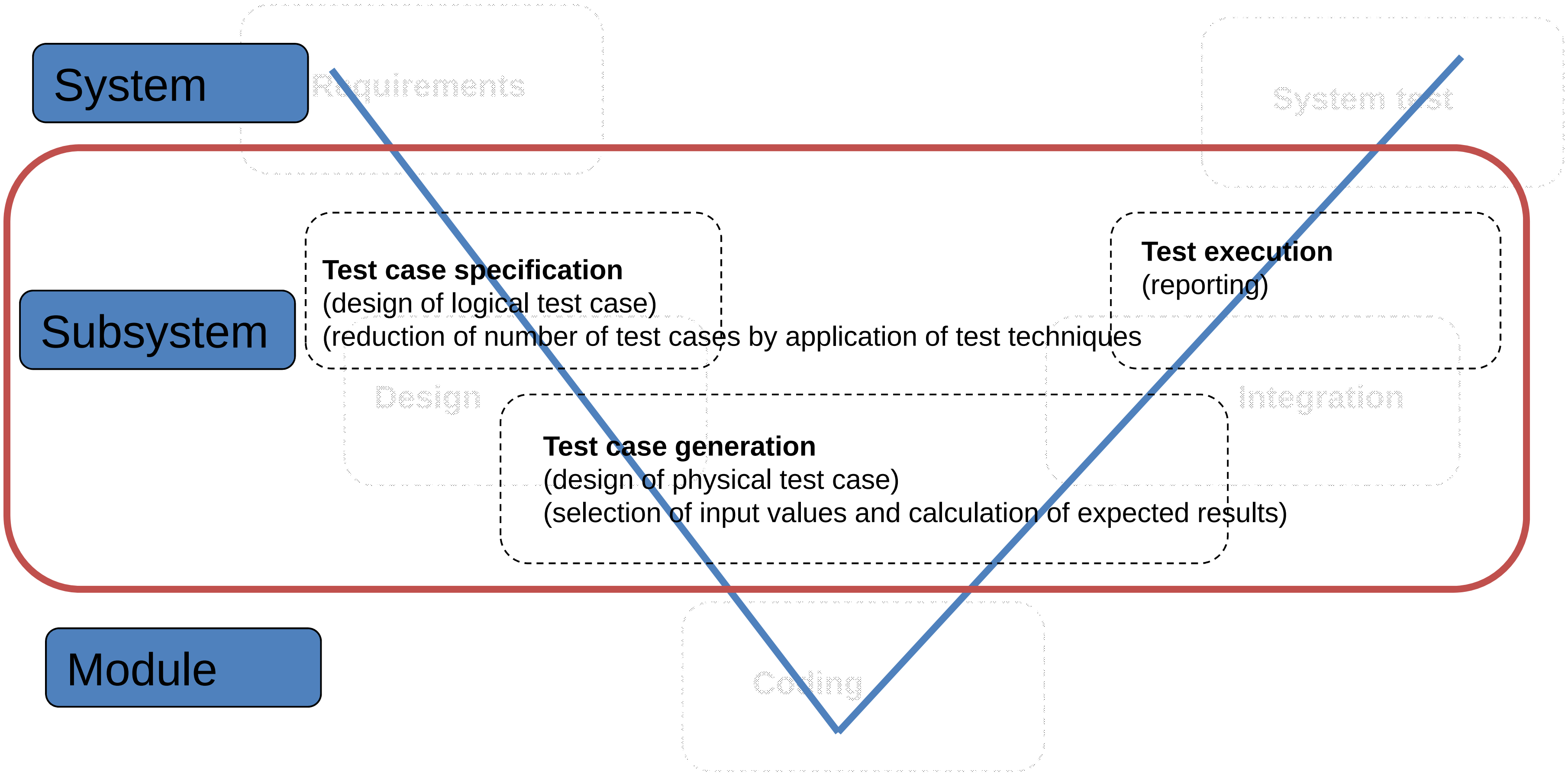
➔ Increased need to technology enablers

Innovative company's need innovative tools and approaches to develop innovative products and services

- ➔ Model Based System Engineering
- ➔ Model Driven Engineering
- ➔ Model Based Testing

Test process

3 main steps in test process



System

Subsystem

Module

Requirements

System test

Design

Integration

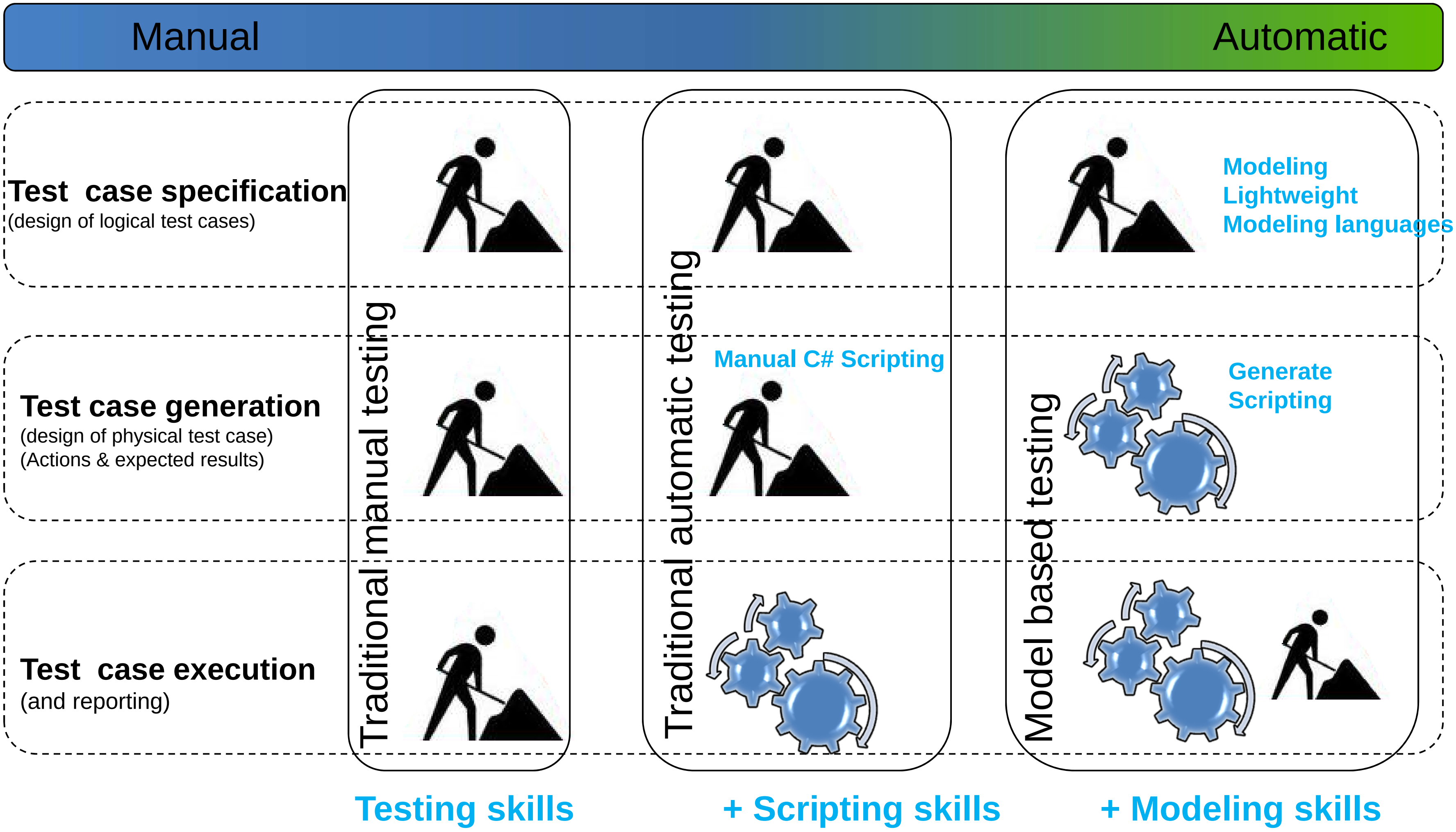
Coding

Test case specification
(design of logical test case)
(reduction of number of test cases by application of test techniques

Test execution
(reporting)

Test case generation
(design of physical test case)
(selection of input values and calculation of expected results)

MBT is the automation of test case generation



Manual

Automatic

Test case specification
(design of logical test cases)

Test case generation
(design of physical test case)
(Actions & expected results)

Test case execution
(and reporting)

Traditional manual testing

Traditional automatic testing

Manual C# Scripting

Model based testing

Modeling
Lightweight
Modeling languages

Generate
Scripting

Testing skills

+ Scripting skills

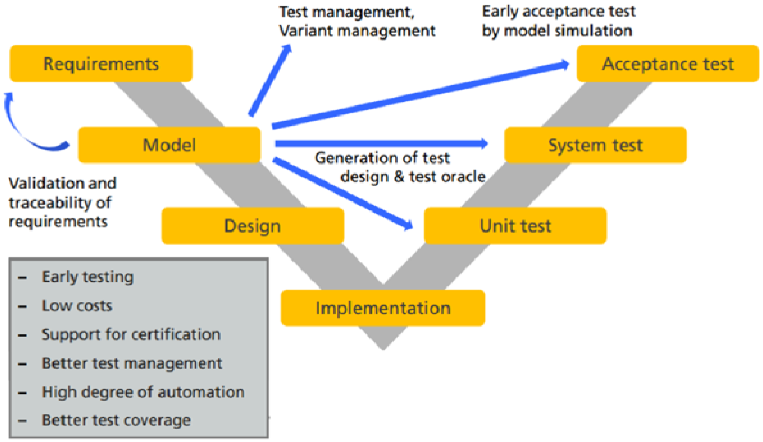
+ Modeling skills

Model Based Testing

+ better communication using models
+ abstract tests
+ auto design of tests, variety of test suites
+ model checking
+ early exposure of ambiguities in spec/design
+ ease of updating of test suites
+ systematic coverage

o automatic execution
o auto regression testing

- modelling overhead
- complexity of the models



Test management,
Variant management

Early acceptance test
by model simulation

Requirements

Acceptance test

Model

Generation of test
design & test oracle

System test

Validation and
traceability of
requirements

Design

Unit test

Implementation

- Early testing
- Low costs
- Support for certification
- Better test management
- High degree of automation
- Better test coverage

Example Spec Explorer and possible extensions

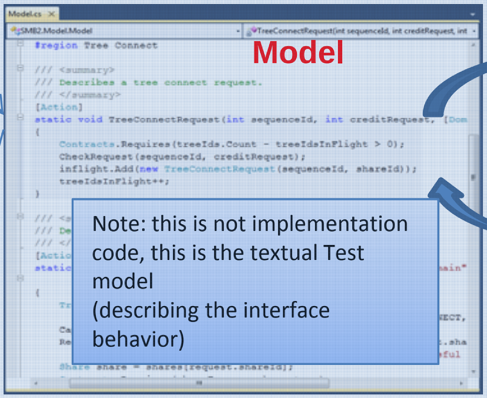
UML Extension Microsoft

Sequence Diagrams

UXSpec
(converts Task Models to SE test models)

ECLIPSE MODELING FRAMEWORK

Model



```
#region Tree Connect

/// <summary>
/// Describes a tree connect request.
/// </summary>
[Action]
static void TreeConnectRequest(int sequenceId, int creditRequest, [Dom
{
    Contracts.Requires(treeIds.Count - treeIdsInFlight > 0);
    CheckRequest(sequenceId, creditRequest);
    inflight.Add(new TreeConnectRequest(sequenceId, shareId));
    treeIdsInFlight++;
}
```

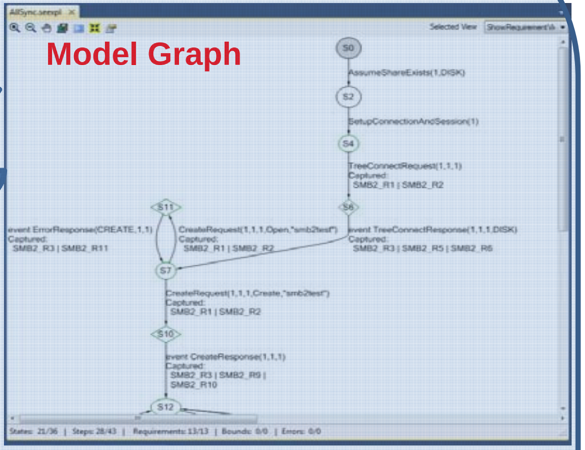
Note: this is not implementation code, this is the textual Test model
(describing the interface behavior)

Explore & Analyze

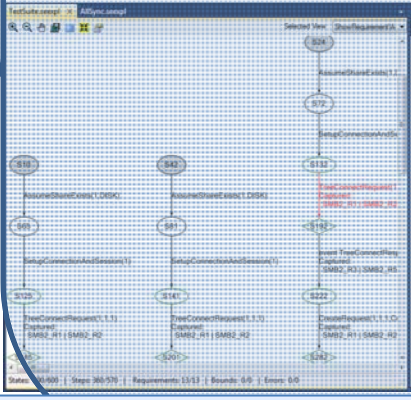
Remodel

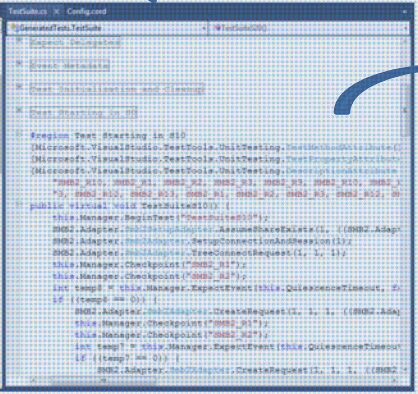
Generate

Model Graph


AllSync.seexpl
Selected View
S0
AssumeShareExists(1,DISK)
S2
SetupConnectionAndSession(1)
S4
TreeConnectRequest(1,1,1)
Captured:
SMB2_R1 | SMB2_R2
S11
S6
event ErrorResponse(CREATE,1,1)
Captured:
SMB2_R3 | SMB2_R11
CreateRequest(1,1,1,Open,"smb2test")
Captured:
SMB2_R1 | SMB2_R2
event TreeConnectResponse(1,1,1,DISK)
Captured:
SMB2_R3 | SMB2_R5 | SMB2_R6
S7
CreateRequest(1,1,1,Create,"smb2test")
Captured:
SMB2_R1 | SMB2_R2
S10
event CreateResponse(1,1,1)
Captured:
SMB2_R3 | SMB2_R9 |
SMB2_R10
S12
States: 21/36 | Steps: 28/43 | Requirements: 13/13 | Bounds: 0/0 | Errors: 0/0

Test Suite





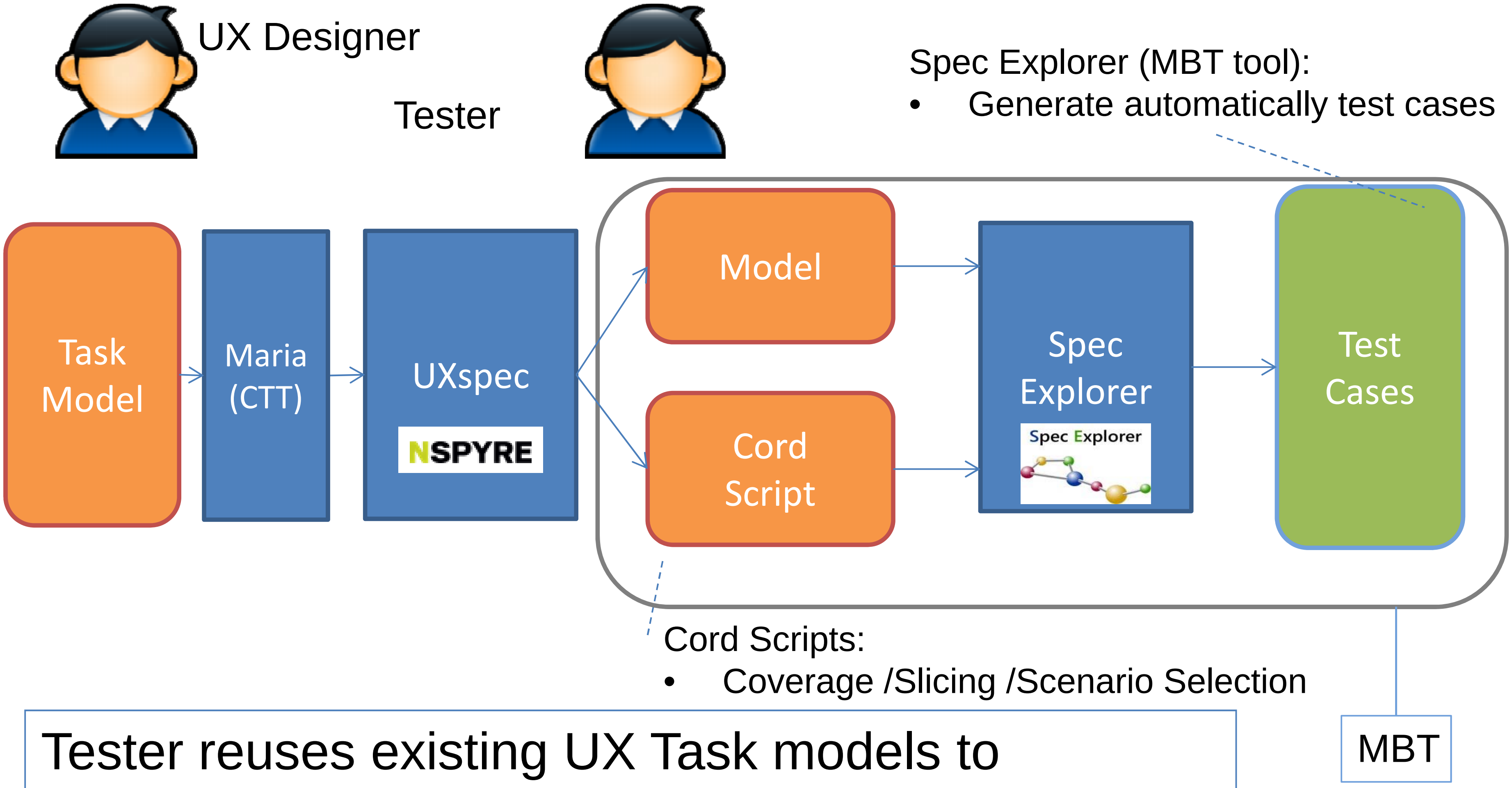
```
Expect Delegates

Event Metadata

Test Initialization and Cleanup

Test Starting in S0

#region Test Starting in S10
[Microsoft.VisualStudio.TestTools.UnitTesting.TestMethodAttribute()
[Microsoft.VisualStudio.TestTools.UnitTesting.TestPropertyAttribute
[Microsoft.VisualStudio.TestTools.UnitTesting.DescriptionAttribute
    "SMB2_R10, SMB2_R1, SMB2_R2, SMB2_R3, SMB2_R9, SMB2_R10, SMB2_
    "3, SMB2_R12, SMB2_R13, SMB2_R1, SMB2_R2, SMB2_R3, SMB2_R12, SM
public virtual void TestSuiteS10() {
    this.Manager.BeginTest("TestSuiteS10");
    SMB2.Adapter.Smb2SetupAdapter.AssumeShareExists(1, ((SMB2.Adap
    SMB2.Adapter.Smb2Adapter.SetupConnectionAndSession(1);
    SMB2.Adapter.Smb2Adapter.TreeConnectRequest(1, 1, 1);
    this.Manager.Checkpoint("SMB2_R1");
    this.Manager.Checkpoint("SMB2_R2");
    int temp8 = this.Manager.ExpectEvent(this.QuiescenceTimeout, fa
    if ((temp8 == 0)) {
        SMB2.Adapter.Smb2Adapter.CreateRequest(1, 1, 1, ((SMB2.Adap
        this.Manager.Checkpoint("SMB2_R1");
        this.Manager.Checkpoint("SMB2_R2");
        int temp7 = this.Manager.ExpectEvent(this.QuiescenceTimeout
        if ((temp7 == 0)) {
            SMB2.Adapter.Smb2Adapter.CreateRequest(1, 1, 1, ((SMB2
```

Test Execute Tool

FAT

MBT extension to overcome modeling /scripting skills and reuse available models

MBT: workflow



UX Designer

Tester

Spec Explorer (MBT tool):

- Generate automatically test cases

Task Model

Maria (CTT)

UXspec

NSPYRE

Model

Cord Script

Spec Explorer

Spec Explorer

Test Cases

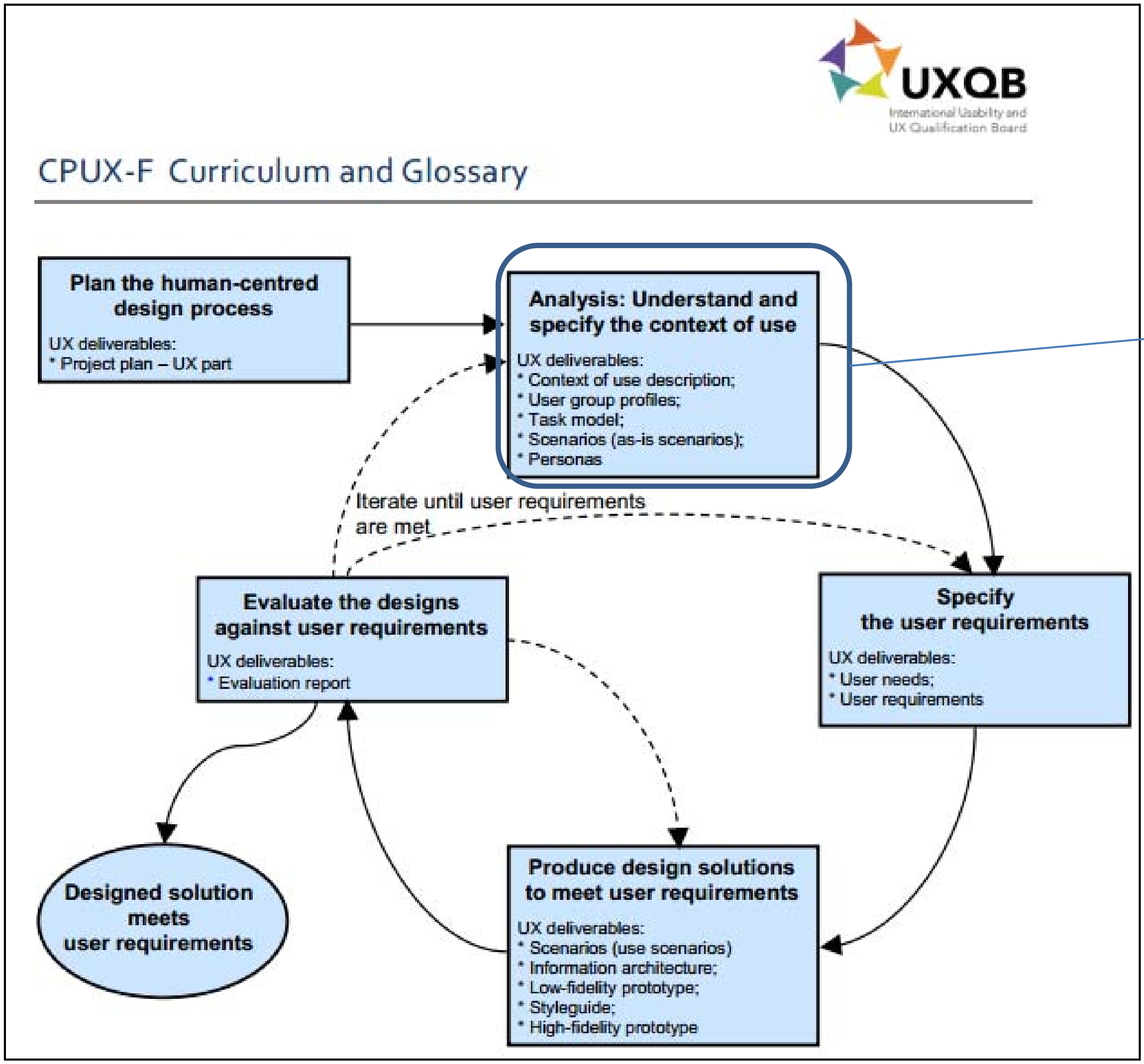
Cord Scripts:

- Coverage /Slicing /Scenario Selection

MBT

Tester reuses existing UX Task models to generate Spec Explorer MBT models

Task Model in human-centered design activities



UXQB
International Usability and UX Qualification Board

CPUX-F Curriculum and Glossary

Plan the human-centred design process

UX deliverables:
* Project plan – UX part

Analysis: Understand and specify the context of use

UX deliverables:
* Context of use description;
* User group profiles;
* Task model;
* Scenarios (as-is scenarios);
* Personas

Iterate until user requirements are met

Evaluate the designs against user requirements

UX deliverables:
* Evaluation report

Specify the user requirements

UX deliverables:
* User needs;
* User requirements

Designed solution meets user requirements

Produce design solutions to meet user requirements

UX deliverables:
* Scenarios (use scenarios)
* Information architecture;
* Low-fidelity prototype;
* Styleguide;
* High-fidelity prototype

This figure shows the interdependence of human-centered design activities according to the ISO 9241-210 standard.

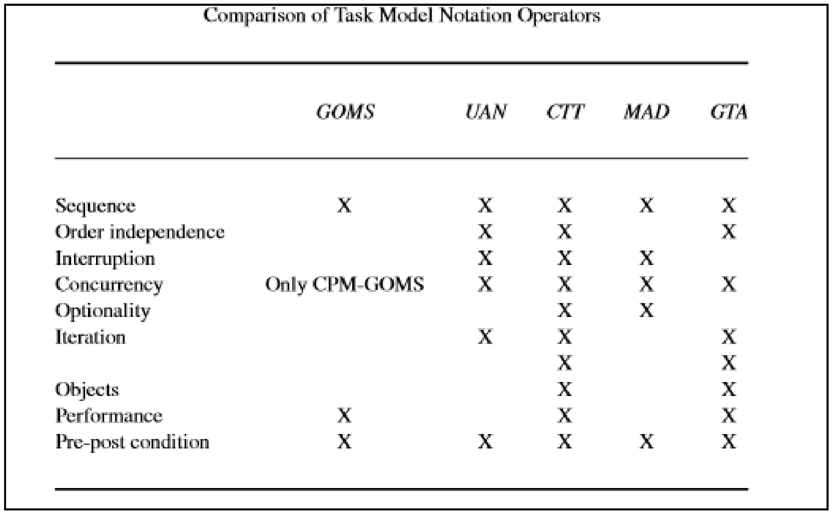
Blue rectangles show the 5 key design activities in an iterative, human-centered design process.
"UX deliverables" are key deliverables from the corresponding design activity.

During Analysis phase, Task Model is one of the deliverables

Task Model: A description of the subtasks that have to be carried out in order to reach the user’s goals

What are task models, anyway?

- Description of the process a user takes to reach a goal in a specific domain
- Typically have hierarchical structure
- Number of different task modeling languages
  - GOMS
  - UAN
  - ConcurTaskTrees (CTT)



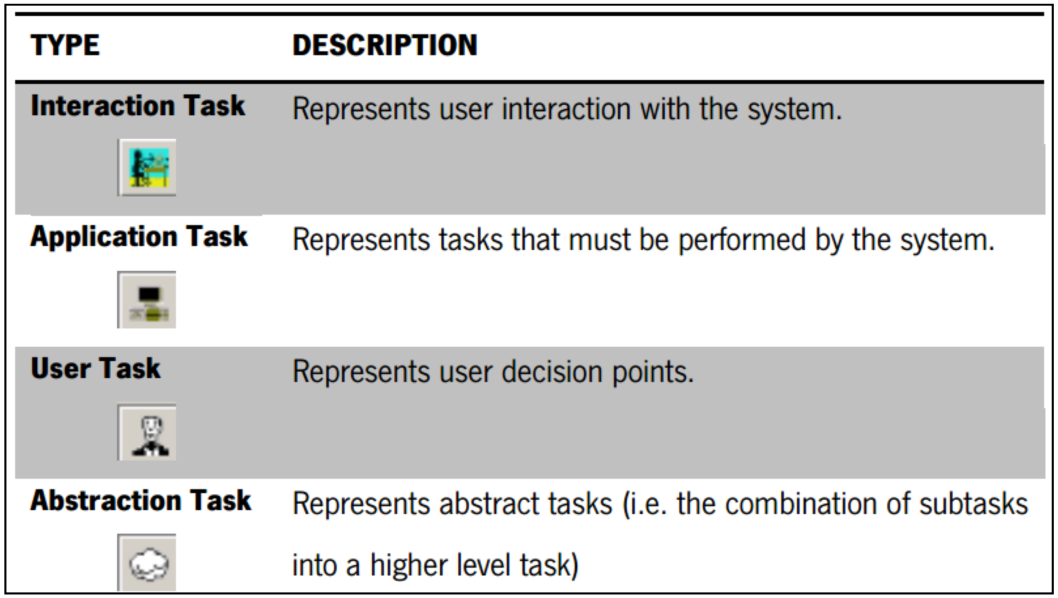
Comparison of Task Model Notation Operators

	GOMS	*UAN*	*CTT*	*MAD*	*GTA*
Sequence	X	X	X	X	X
Order independence		X	X		X
Interruption		X	X	X	
Concurrency	Only CPM-GOMS	X	X	X	X
Optionality			X	X	
Iteration		X	X		X
			X		X
Objects			X		X
Performance	X		X		X
Pre-post condition	X	X	X	X	X

http://giove.isti.cnr.it/attachments/publications/2003-A1-07.pdf

ConcurTaskTrees

- Developed by Fabio Paterno et al. for the design of user interfaces
- Goals
  - Graphical for easy interpretation
  - Concurrent model for representing UI tasks
  - Different task types
    - Represent all tasks, including those performed by the system



TYPE	DESCRIPTION
Interaction Task	Represents user interaction with the system.
Application Task	Represents tasks that must be performed by the system.
User Task	Represents user decision points.
Abstraction Task	Represents abstract tasks (i.e. the combination of subtasks into a higher level task)

Task Building Process

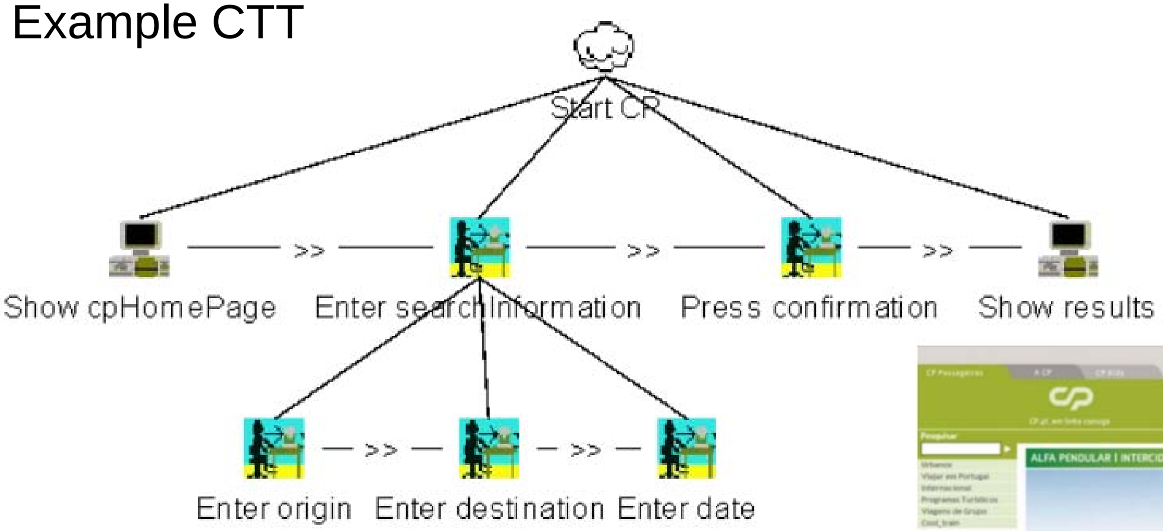
Three phases

- Hierarchically decompose the tasks
- Identify the temporal relationships among tasks at same level
- Identify what objects are manipulated and what actions can be performed on them, and assign these to the tasks as appropriate.

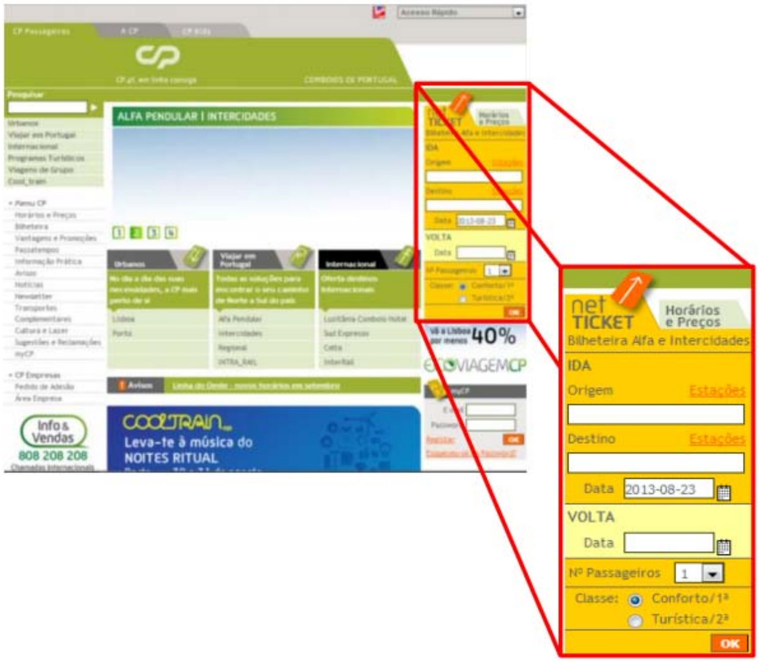
Temporal Relationships

- **T1 [] T2** - Choice
- **T1 ||| T2** - Interleaving
- **T1 |[]| T2** - Synchronization
- **T1 >> T2** - Enabling
- **T1 []>> T2** - Enabling with Information Passing
- **T1 [> T2** - Deactivation
- **T1*** - Iteration
- **T1(n)** - Finite Iteration
- **[T1]** - Optional
- **T** – Recursion

Example CTT



Ref: Paulo Filipe de Jesus Cruz


ALFA PENDULAR I INTERCIDADES
Info & Vendas
808 208 208
COOLTRAIN
Leva-te à música do NOITES RITUAL
40%
net TICKET
Horários e Preços
Bilheteira Alfa e Intercidades
OK

IDA

Origem Estações

Destino Estações

Data 2013-08-23

VOLTA

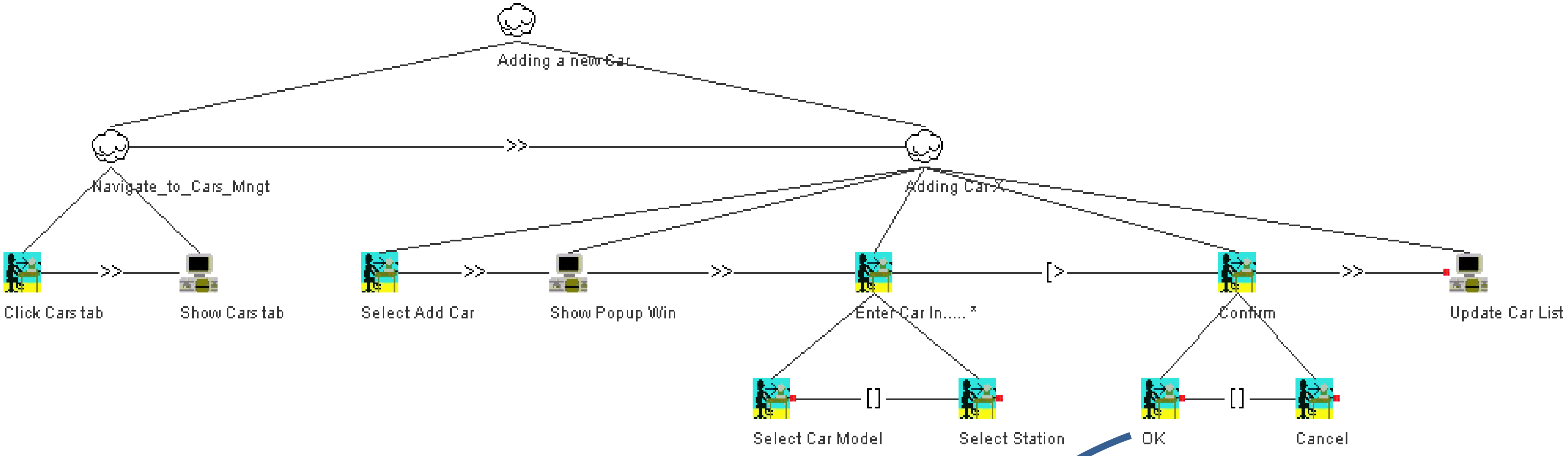
Data

Nº Passageiros 1

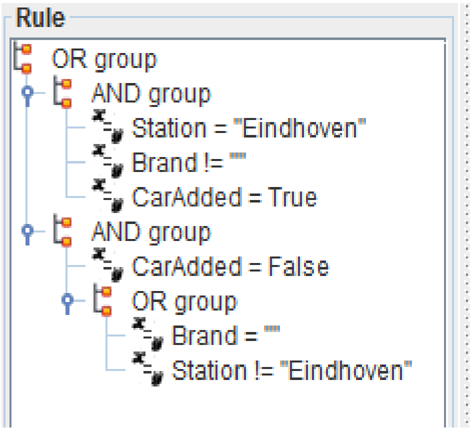
Classe: ● Conforto/1ª ○ Turística/2ª

Figure 5 - Searching Alpha and Intercity trains in CP web page

Case study:
Car reservation system

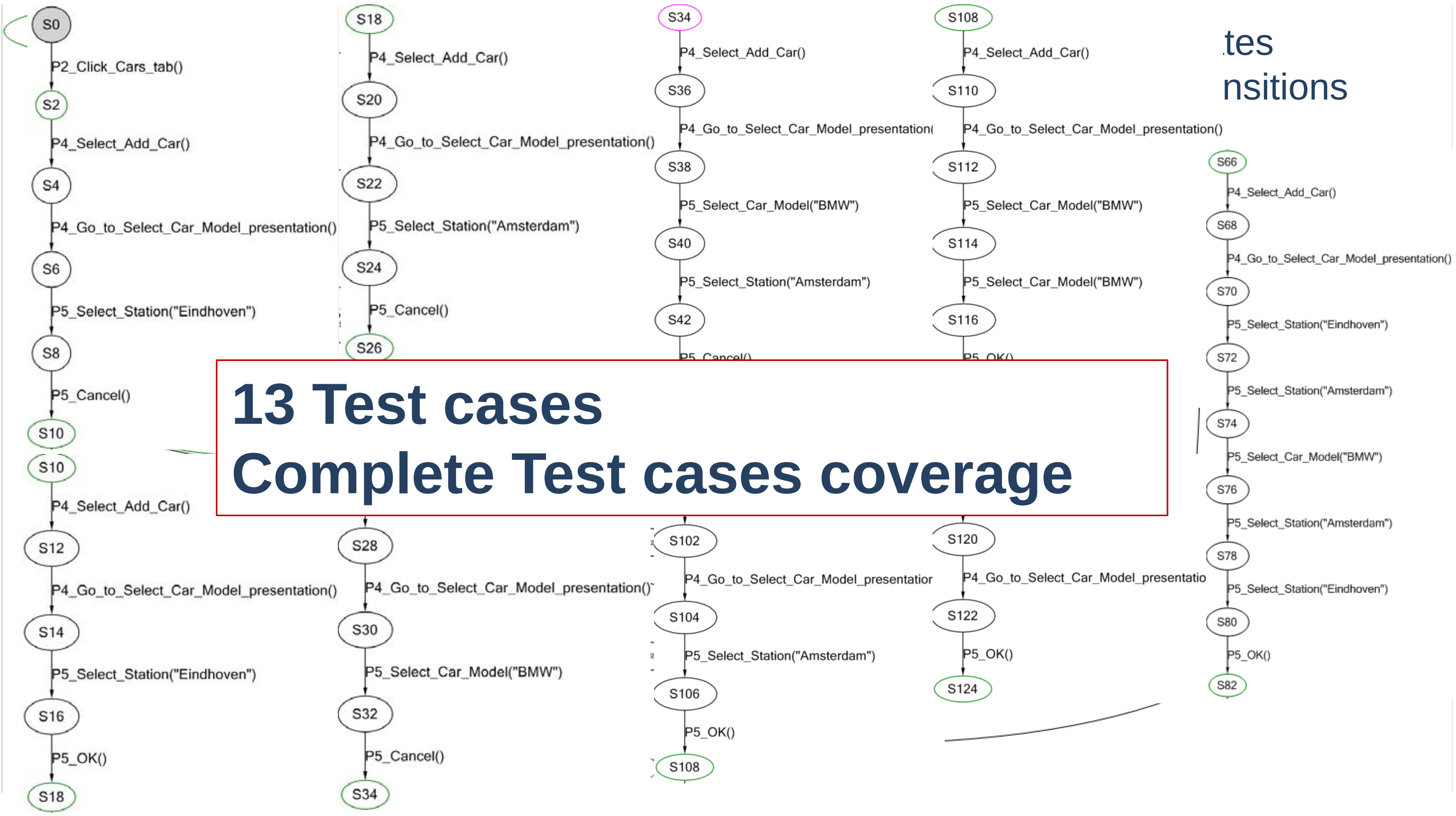


Postcondition


Rule

- OR group
  - AND group
    - Station = "Eindhoven"
    - Brand != ""
    - CarAdded = True
  - AND group
    - CarAdded = False
    - OR group
      - Brand = ""
      - Station != "Eindhoven"

MBT + UXspec: generated Model


S0
P2_Click_Cars_tab()
S2
P4_Select_Add_Car()
S4
P4_Go_to_Select_Car_Model_presentation()
S6
P5_Select_Station("Eindhoven")
S8
P5_Cancel()
S10
S10
P4_Select_Add_Car()
S12
P4_Go_to_Select_Car_Model_presentation()
S14
P5_Select_Station("Eindhoven")
S16
P5_OK()
S18
S18
P4_Select_Add_Car()
S20
P4_Go_to_Select_Car_Model_presentation()
S22
P5_Select_Station("Amsterdam")
S24
P5_Cancel()
S26
S28
P4_Go_to_Select_Car_Model_presentation()
S30
P5_Select_Car_Model("BMW")
S32
P5_Cancel()
S34
S34
P4_Select_Add_Car()
S36
P4_Go_to_Select_Car_Model_presentation(
S38
P5_Select_Car_Model("BMW")
S40
P5_Select_Station("Amsterdam")
S42
P5_Cancel()
S102
P4_Go_to_Select_Car_Model_presentatior
S104
P5_Select_Station("Amsterdam")
S106
P5_OK()
S108
S108
P4_Select_Add_Car()
S110
P4_Go_to_Select_Car_Model_presentation()
S112
P5_Select_Car_Model("BMW")
S114
P5_Select_Car_Model("BMW")
S116
P5_OK()
S120
P4_Go_to_Select_Car_Model_presentatio
S122
P5_OK()
S124
S66
P4_Select_Add_Car()
S68
P4_Go_to_Select_Car_Model_presentation()
S70
P5_Select_Station("Eindhoven")
S72
P5_Select_Station("Amsterdam")
S74
P5_Select_Car_Model("BMW")
S76
P5_Select_Station("Amsterdam")
S78
P5_Select_Station("Eindhoven")
S80
P5_OK()
S82

tes
nsitions

13 Test cases

Complete Test cases coverage

Next Steps

- Run project based on real product from industry
- Tool improvements
  - Support data handling/configuration aspects in UXSpec, instead of relying on manual additions in Spec Explorer,
  - Supporting other MBT tools.

Questions?

MBT is…..

Strategies, tools and artifacts

Manage complexity

Reduces the need of manual or human involvement of interaction

Avoids spending time in unskilled repetitive error prone or redundant tasks

Provides bandwidth to Innovate!!

Benefits of (CTT) Task Modeling

- Translation of design to Concur Task Tree seems straight forward
- Clear visual way of specifying all interaction details (e.g. all input fields in a screen) and user interaction flows
- Changes in requirements: Only keep this model up to date, not have to change all design mockups
- Model can be maintained by others than UX (free tooling)

Results

	Manual test design and manual scripting (automatic test execution)		MBT+ UXspec
SUT	Car reservation system (2 requirements)		Car reservation system (2 requirements)
Test Design (logical test case)	0,5 h for 2 test specification		½ h for creating task model
Test Generation (physical test case)	0,5 h for 2 scripting the test cases		Automatic generation of 13 test cases
Test Execution	Automatic execution (MS UI automation)		Automatic execution (MS UI automation)
Number of Test Cases	2 Test cases (2 test case/hrs)		13 Test cases (26 test case/hrs)
Coverage	2 scenarios 2 req		All scenarios 2 req
Maintenance Effort (change /new requirements)	1 h for adding test scenario (new requirement)		½ h for updating task model