A Structured Approach for Efficient Model-Based Testing in Large IT Projects

UCAAT 2013
22 – 24 October - Paris

Jean-Pierre Schoch – Bruno Legeard
{jean-pierre.schoch, bruno.legeard}@smartesting.com

Agenda

smartesting®
Optimize your Test Center

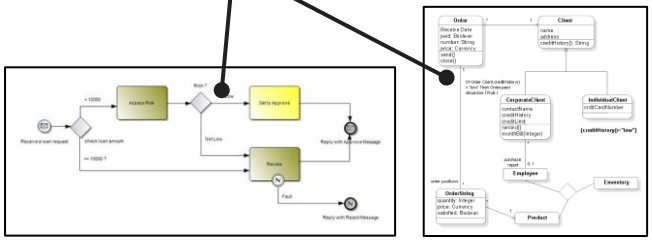
- MBT for Large IT Systems
  - Current challenges of testing large-scale applications
  - Levels of testing addressed by model-based testing
- Building the Test Generation models
  - Understanding and controlling your test requirements
  - Understanding and composing business process models
  - Designing the test generation models
- Reuse and Multi-Model Systems
  - Enabling reuse and collaborative work
  - Structuring models as a layered architecture

Large-scale Enterprise Information Systems

- **System of systems & Complex composite systems**
  - Multiple applications
    - Mix of Bespoke and Packaged applications
    - Mix of data-oriented and process-oriented applications
  - Multiple targeted platforms (PC, Smartphone, Pad)
- **Testing needs**
  - Business workflow and business rules oriented
  - **Application testing**, but also **end-to-end testing**
  - Requirements and Business Process **coverage**
  - 80% of test execution still **manual** (and for some part will remain manual)

Model-Based Testing in a Nutshell

Model Assets for Automated Test Generation



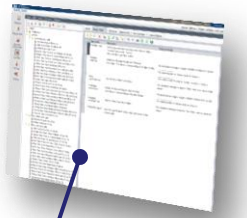
Business Needs

MBT Test Production

Test Design Automation

MBT Automated Traceability

Test Repository

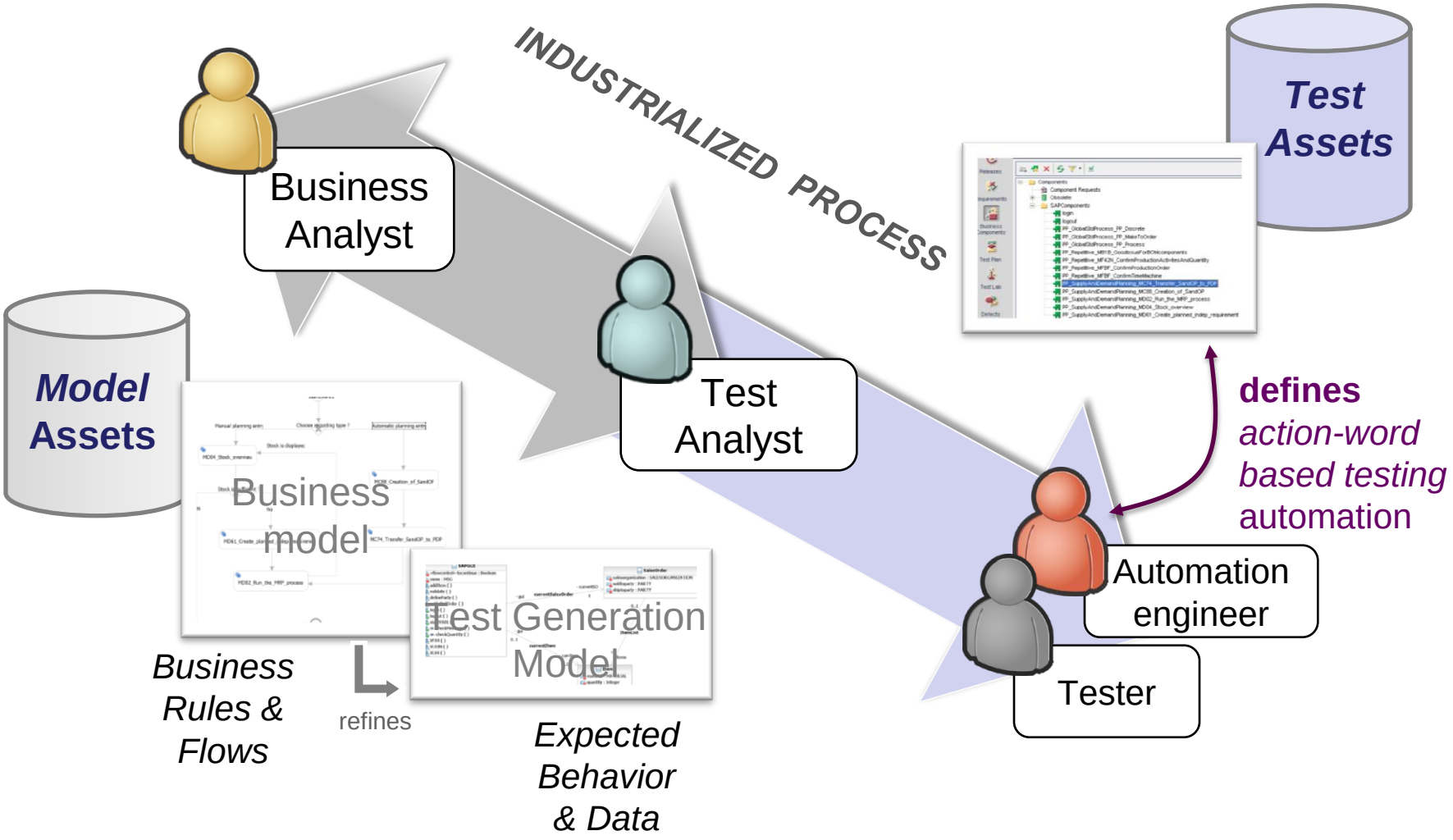


Test Execution

automated

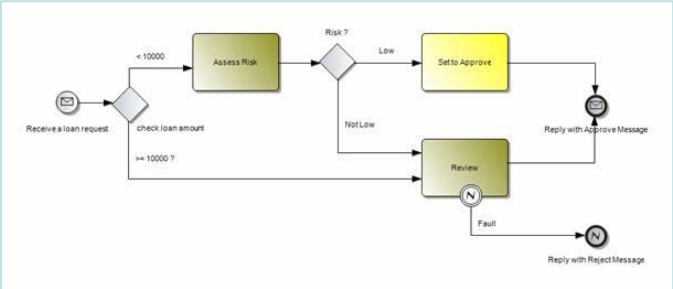
manual

Roles in the Model-Based Testing Process

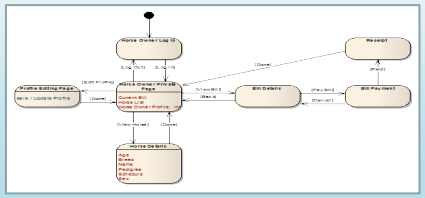


Models for Automated Test Generation

Business Process Model (BPMN)

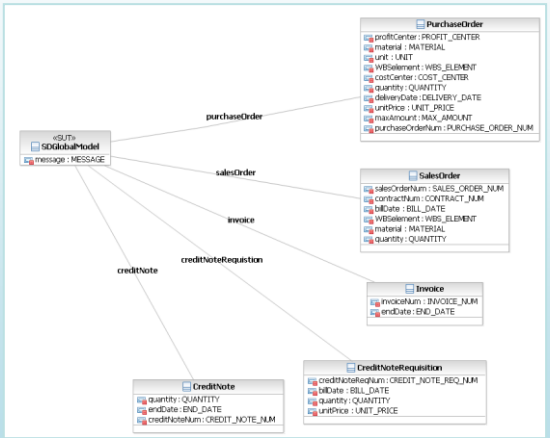


Business Rules and Behavioral Model (UML/OCL)



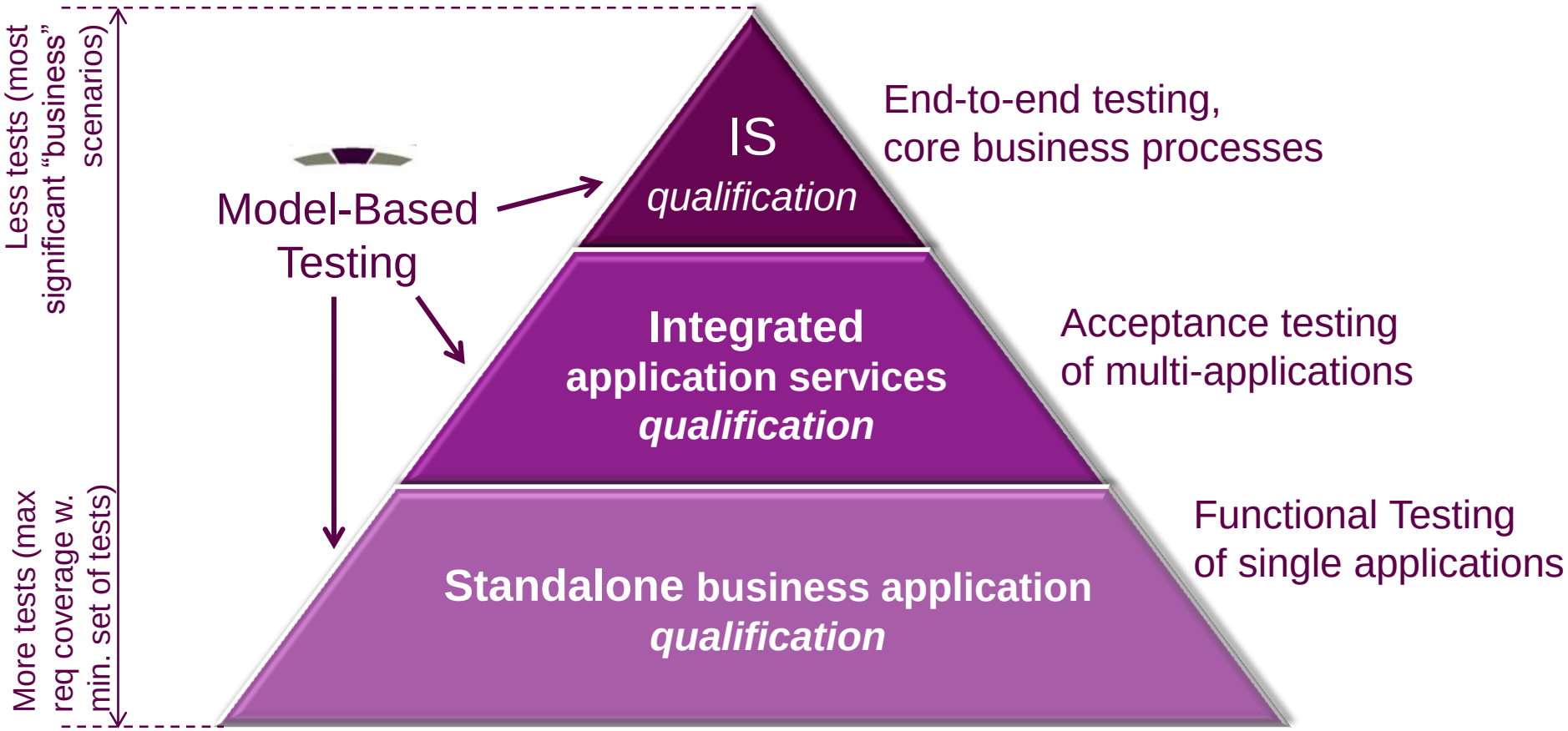
Conditions	Rules				
< 5 years	✓	×	×	×	×
>= 5 and < 18	×	✓	×	×	×
>= 18 and < 55 with concession card	×	×	✓	×	×
>= 18 and < 55 no concession card	×	×	×	✓	×
>= 55	×	×	×	×	✓

Business Entities and Logical Test Data (UML)



Modeling notations

What Types of Tests?

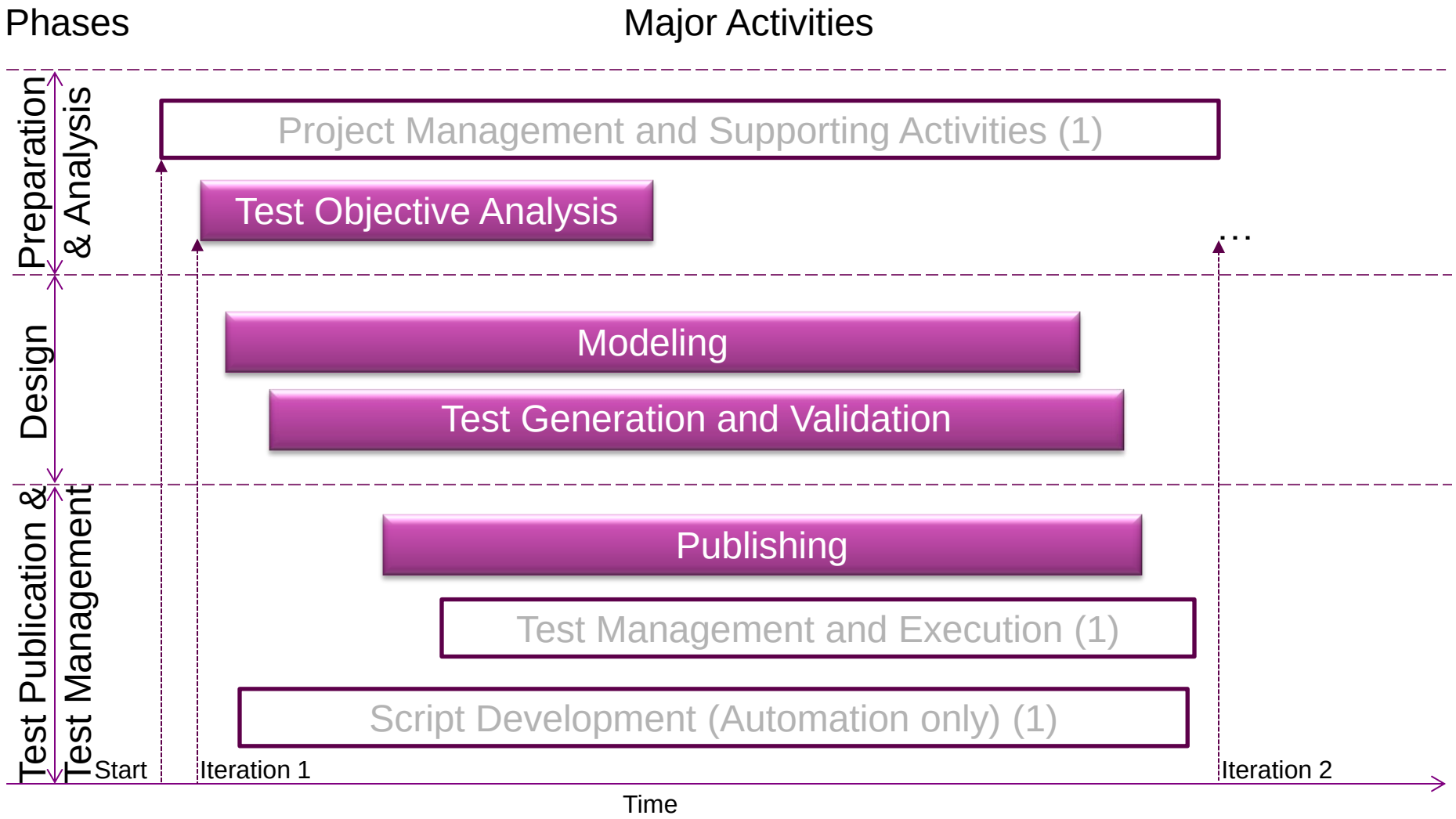


Agenda

smartesting
Optimize your Test Center

- MBT for Large IT Systems
  - Current challenges of testing large-scale applications
  - Levels of testing addressed by model-based testing
- Building the Test Generation models
  - Understanding and controlling your test requirements
  - Understanding and composing business process models
  - Designing the test generation models
- Reuse and Multi-Model Systems
  - Enabling reuse and collaborative work
  - Structuring models as a layered architecture

The Test Generation Process avec Smartesting CertifyIt



(1) Not covered in this tutorial

Preparation & Analysis
1. Defining Test Requirements

- Tests typically created to verify specific requirements
  - Formal or not
  - Capture all test requirements in a Test Objective Charter (next slide)
  - The TOC is used as the "contract" between:

    The different stakeholders, typically represented by business analysts and functional experts

    AND

    The test analysts responsible for designing the behavioral model(s)
- References in the models to the covered requirements provide the basis for:
  - Automatic traceability between requirements and generated tests
    - Traceability links are part of the info published into the test environment
  - Accurate progress tracking

Preparation & Analysis
2. Test Objective Charter

Use Cases

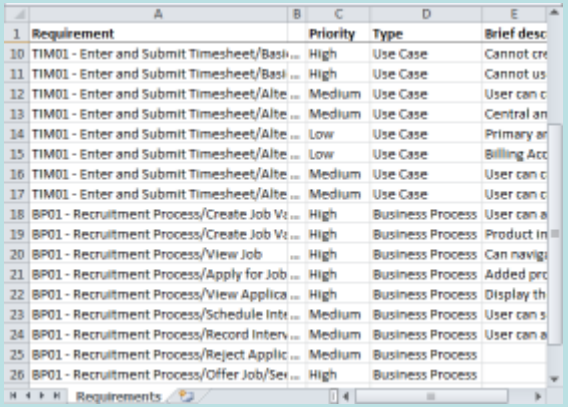
Textual Requirements

Application Mockups

Business Processes

And all Other Sources…

Test Objective Charter



	A	B	C	D	E
1	Requirement		Priority	Type	Brief desc
10	TIM01 - Enter and Submit Timesheet/Basi...		High	Use Case	Cannot cre
11	TIM01 - Enter and Submit Timesheet/Basi ...		High	Use Case	Cannot us
12	TIM01 - Enter and Submit Timesheet/Alte ...		Medium	Use Case	User can c
13	TIM01 - Enter and Submit Timesheet/Alte ...		Medium	Use Case	Central an
14	TIM01 - Enter and Submit Timesheet/Alte ...		Low	Use Case	Primary ar
15	TIM01 - Enter and Submit Timesheet/Alte ...		Low	Use Case	Billing Acc
16	TIM01 - Enter and Submit Timesheet/Alte ...		Medium	Use Case	User can c
17	TIM01 - Enter and Submit Timesheet/Alte ...		Medium	Use Case	User can c
18	BP01 - Recruitment Process/Create Job Ve ...		High	Business Process	User can a
19	BP01 - Recruitment Process/Create Job Ve ...		High	Business Process	Product in
20	BP01 - Recruitment Process/View Job ...		High	Business Process	Can navig
21	BP01 - Recruitment Process/Apply for Job ...		High	Business Process	Added pro
22	BP01 - Recruitment Process/View Applica ...		High	Business Process	Display th
23	BP01 - Recruitment Process/Schedule Inte ...		Medium	Business Process	User can s
24	BP01 - Recruitment Process/Record Interv ...		Medium	Business Process	User can a
25	BP01 - Recruitment Process/Reject Applic ...		Medium	Business Process	
26	BP01 - Recruitment Process/Offer Job/Ser ...		High	Business Process	

Requirements

- Unique reference for “test” requirements
- Can be exported from existing requirement repositories
- Includes attributes such as priority, criticality, target release, etc.
- The “contract” between the BAs and the modeling team

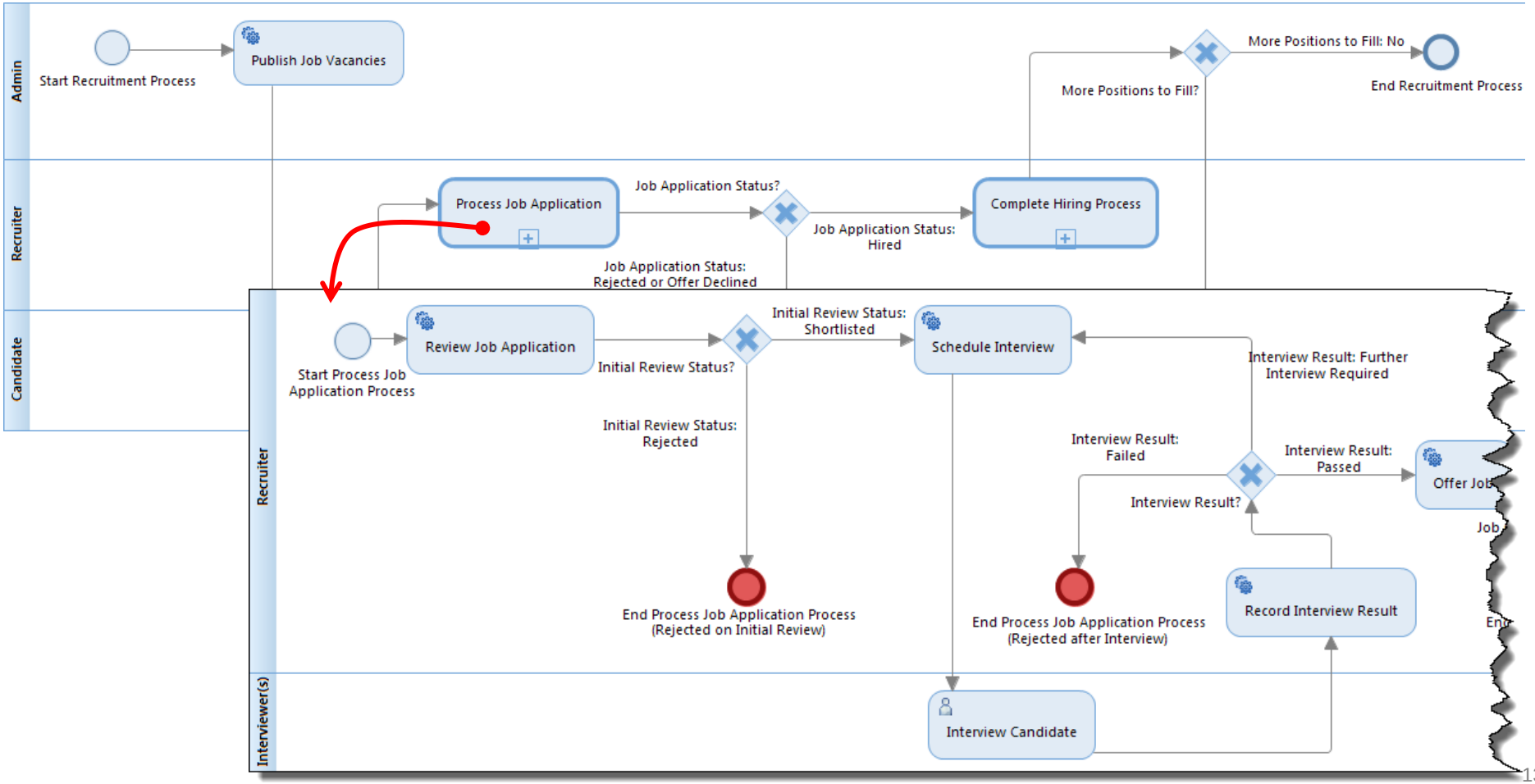
Preparation & Analysis
3. Capturing System Flows

- System Flows = Sequences of operations and/or activities performed by human users and/or external systems, and the system's various responses
  - Many types: textual or graphical, technical or business-oriented (e.g. workflows, business process diagrams, use case flows of events)
- Business Flows = business view of the system under test
  - Identify Business Actions (BAs) = elementary business units
  - Tests = sequences of BAs
  - Business flows represented as Business Processes
- Application Flows = technical details of the business flows
  - Identify Test Actions (TAs) = the "implementation" for the BAs
  - BAs are sequences of TAs

Preparation & Analysis

4. Business Processes

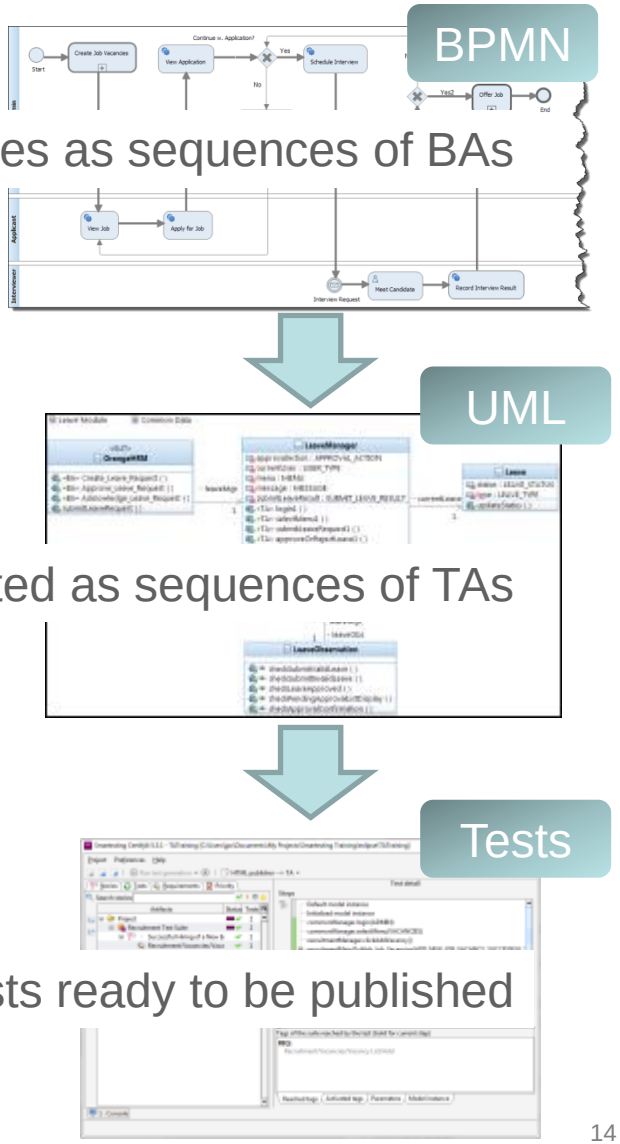
- CertifyIt supports Business Process Modeling using BPMN (Business Process Modeling Notation)



Preparation & Analysis

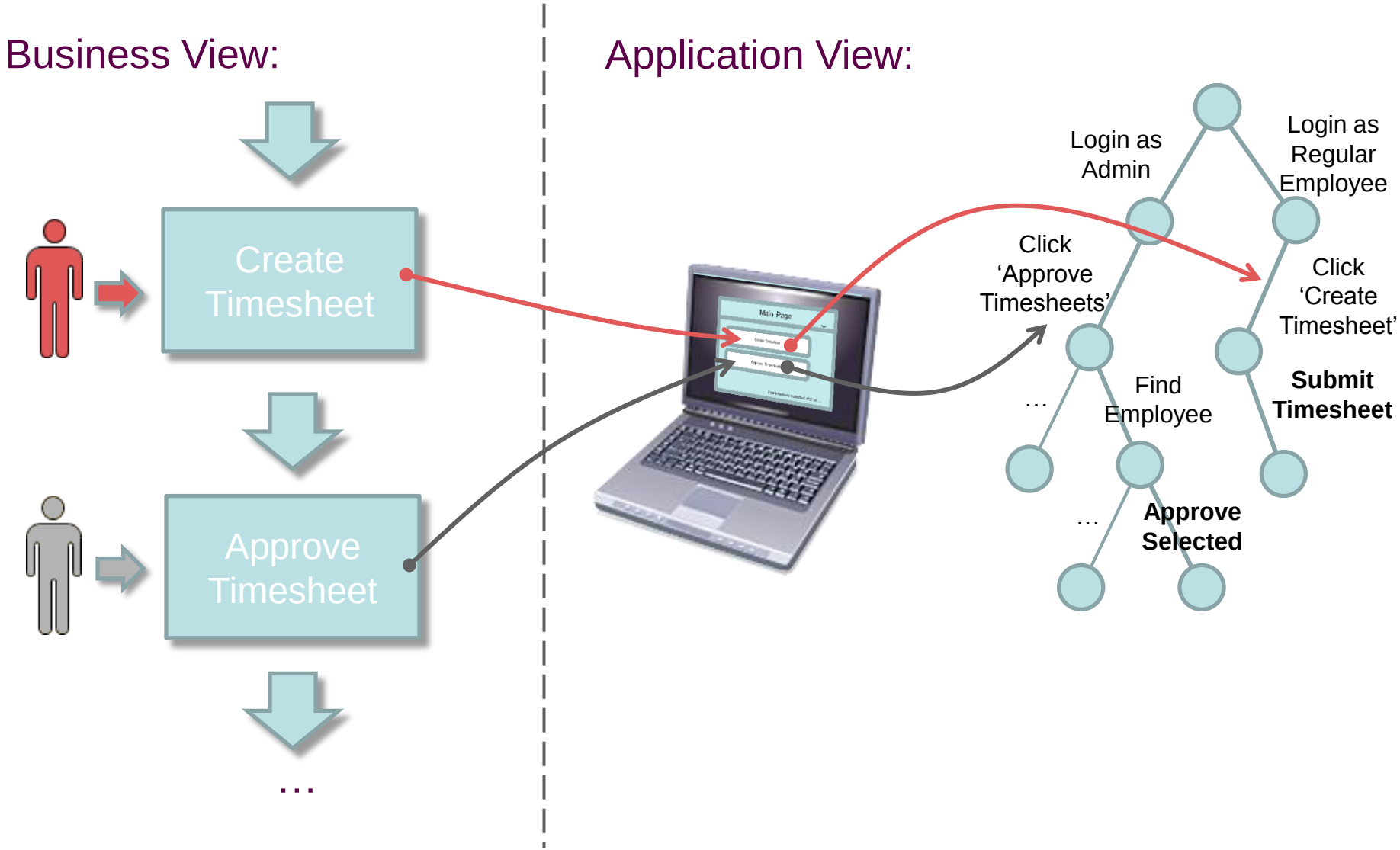
5. Strategy for the Test Generation Process

- Top-Down Strategy
  - Most natural approach:
    - Create the BPs (BAs are produced)
    - Implement the BAs (TAs are produced)
    - Generate the tests
- Meet-in-the-Middle Strategy
  - If you have trouble identifying the “right” BPs:
    - Identify the BAs first
    - Implement them
    - Generate test cases using “test-only” BPs



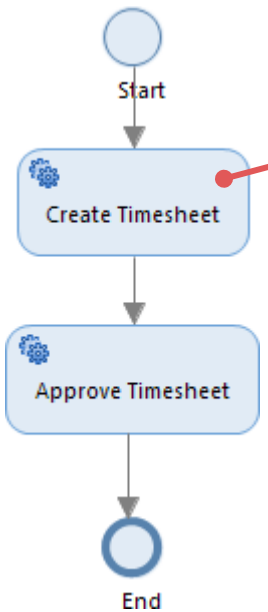
Building the Test Generation Models

1. Business and Application Scenarios



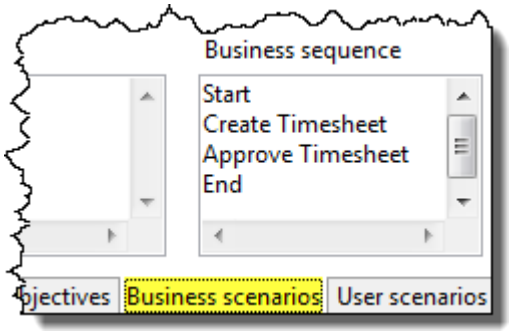
Building the Test Generation Models

2. Business and Application Scenarios (cont'd)

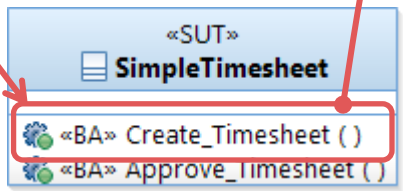


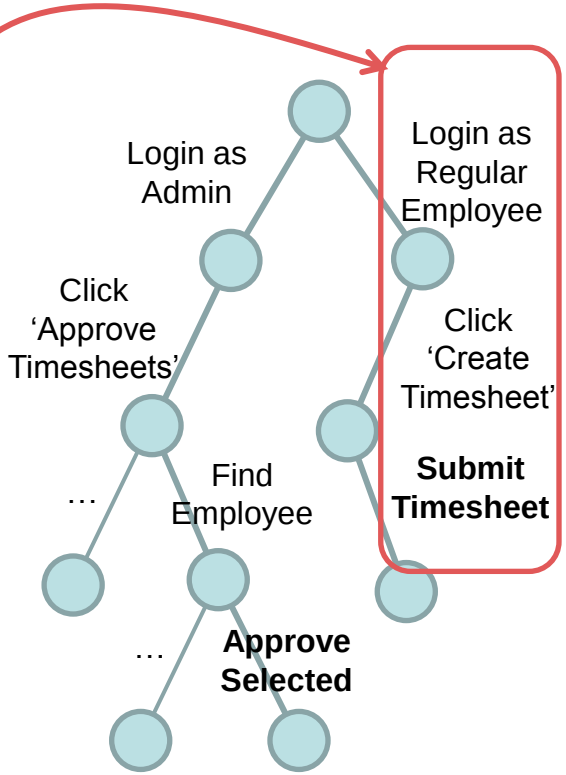
Business tasks are mapped to BA operations in the Test Generation model

Business scenarios are instances of business processes



BA operations are mapped to "application" scenarios, i.e. sequences of TAs





Business scenarios are higher-level constructs that exploit the fine-grained UML model

Building the Test Generation Models
3. Design Driven by BAs

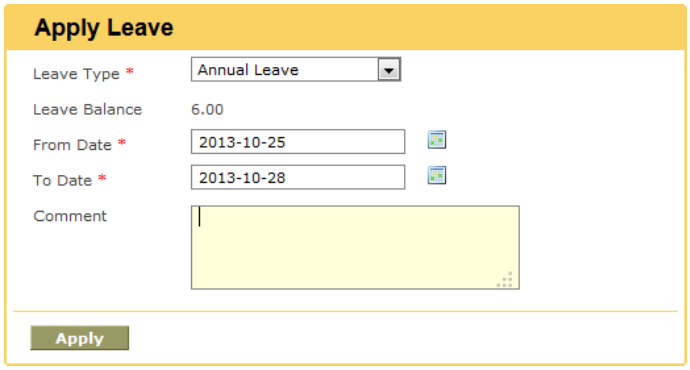
- Business Actions
  - The building blocks of your test projects
  - Every test is a sequence of BAs
  - Understanding the BAs is the key to a successful test project
    - This means first analyzing the BAs based on the test strategy and on the previous artifacts (TOC, Business Processes), and capturing the results in a BA specification
- The BA specification
  - Prerequisites to use the BA
  - Factors of variability: all the elements that impact the behavior of the BA (and that requires testing)
  - Usage context: all the valid configurations for use of the BA (corresponding to possible combinations of the factors of variability)
  - Application workflow: the actions that a user would need to take to achieve the desired outcome

White-box view of the BA

Building the Test Generation Models

4. Example of a BA Specification

- *Prerequisites:* To be connected (any user)
- *Factors of variability:*
  - Connected user: Regular, Admin, Manager
  - Leave type: annual leave, sick leave, family leave, etc.
  - User inputs:
    - Success: 1 day or less, 2 days or more
    - Error: mandatory field(s) missing, invalid date format, etc.



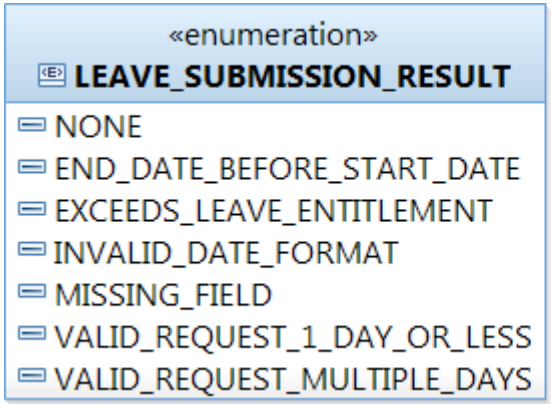
- *Usage context:*

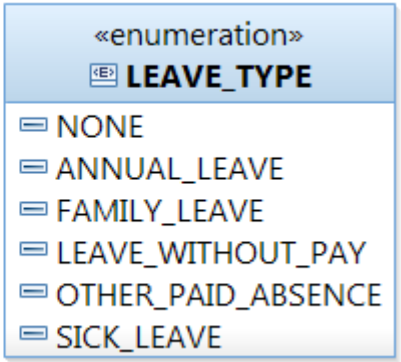
Connected user	Leave type	User inputs	
Regular	Each type	Success (2 days or more)	Nominal case
Regular	Indifferent	Success (1 day or less)	Nominal case
Regular	Indifferent	Each error case	Error cases
Other users	Indifferent	Success (any case)	

- *Application workflow:*
  1. Select the menu *Leave > Apply*
  2. Fill out the form (based on the cases being tested) and click ‘Apply’

Building the Test Generation Models
5. Equivalence Classes as Enumerations

- Equivalence Classes
  - Black-box testing technique
  - Divides all possible inputs (and outputs) into equivalence classes:
    - The test that results from the representative value for a class is said to be "equivalent" to the other values in the same class
  - Example: UNDER_AGE (less than 18), YOUTH_AGE (between 18 and 25), ADULT_AGE (over 25)
- Modeled as enumerations to represent the values the factors of variability
  - Each value documented in natural language, e.g.:





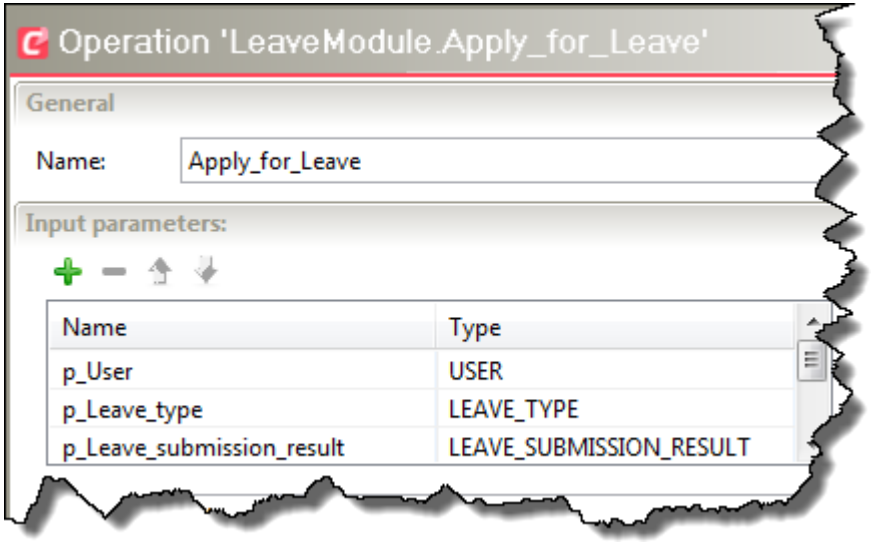
ANNUAL_LEAVE = "Annual Leave"

EXCEEDS_LEAVE_ENTITLEMENT = "Enter a duration that exceeds the number of days available"

Building the Test Generation Models

6. Modeling the BA

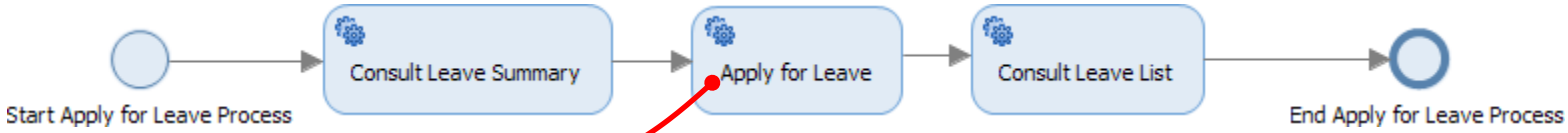
- BA = model operation
  - Prerequisite:
    - Precondition of the operation
  - Factors of variability:
    - Parameters of the operation based on enumerations



  - Usage context:
    - “Decision table” associated with the operation
  - Application workflow:
    - Several solutions: here using “structured descriptions” in the BA operation

Building the Test Generation Models

7. Usage Context as a Decision Table



Double-click the task in the business process to open the decision table

*Apply_for_Leave | 02. Apply for Leave Process.bpmx | *01.Vacation Process.bpmx

Decision Tables

Add table

Usage Context for Apply_for_Leave

	p_User: USER	p_Leave_type: LEAV...	p_Leave_submission_result: LEA...	CUSTOM
1	ESS	ANNUAL_LEAVE	VALID_REQUEST_MULTIPLE_DAYS	NOMINAL_CASE
2	ESS	FAMILY_LEAVE	VALID_REQUEST_MULTIPLE_DAYS	NOMINAL_CASE
3	ESS	LEAVE_WITHOUT_PAY	VALID_REQUEST_MULTIPLE_DAYS	NOMINAL_CASE
4	ESS	OTHER_PAID_ABSENCE	VALID_REQUEST_MULTIPLE_DAYS	NOMINAL_CASE
5	ESS	SICK_LEAVE	VALID_REQUEST_MULTIPLE_DAYS	NOMINAL_CASE
6	ESS		VALID_REQUEST_1_DAY_OR_LESS	NOMINAL_CASE
7	ESS		END_DATE_BEFORE_START_DATE	ERROR_CASE
8	ESS		EXCEEDS_LEAVE_ENTITLEMENT	ERROR_CASE
9	ESS		INVALID_DATE_FORMAT	ERROR_CASE
10	ESS		MISSING_FIELD	ERROR_CASE

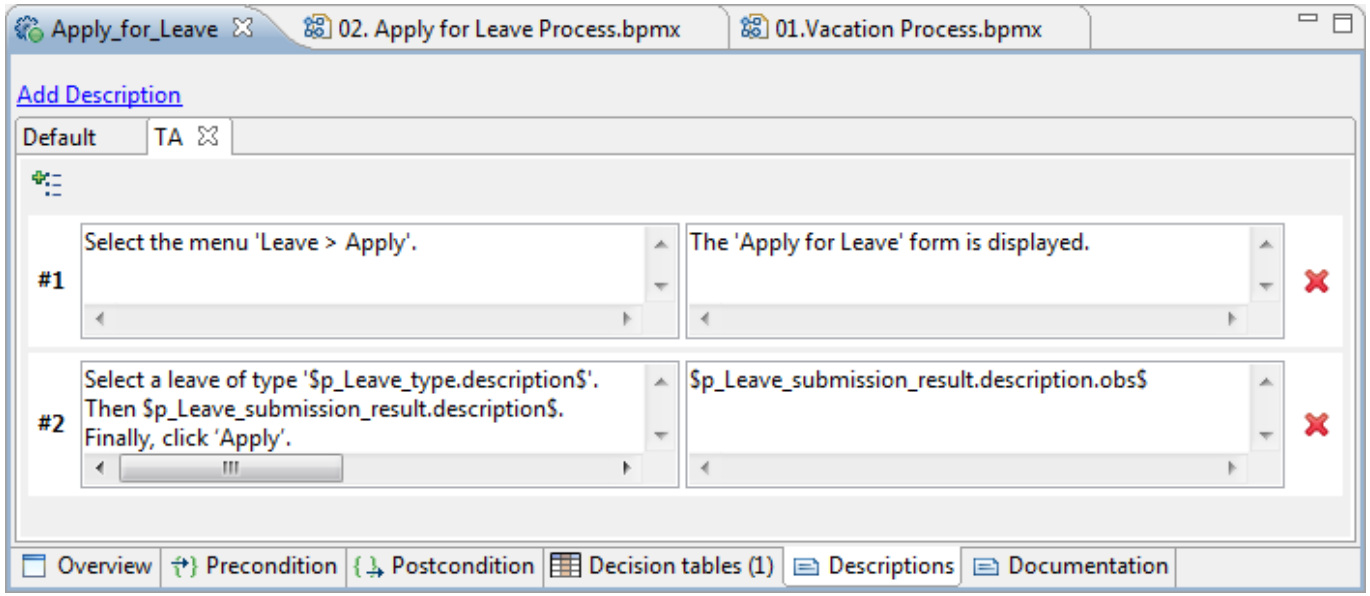
Overview | Precondition | Postcondition | Decision tables (1) | Descriptions | Documentation

We will see later on how to select criteria to generate tests based on one or more lines in the decision table

Building the Test Generation Models

8. Application Workflow as a Structured Description

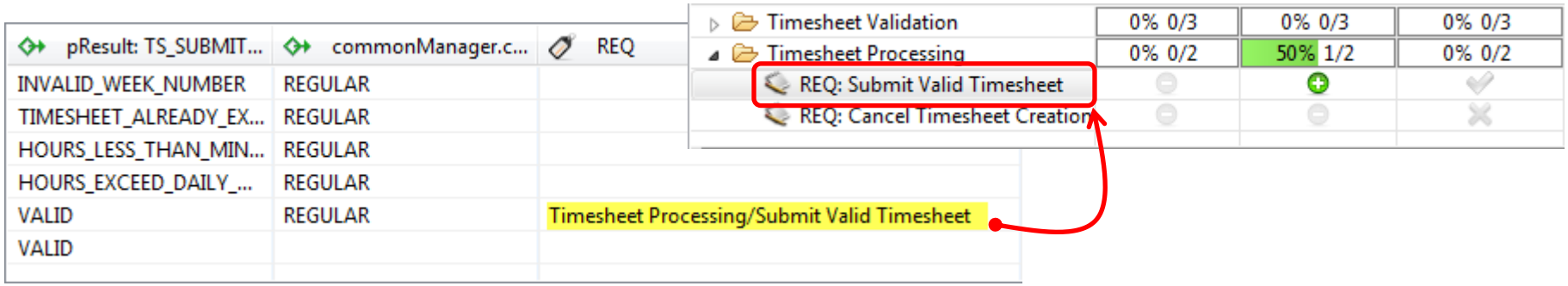
- CertifyIt provides the ability to use “multiple-step” description, which will eventually produce separate tests
  - The two columns correspond to the “design steps” and “expected results” in the future tests
  - On step #2, note the reference to the BA parameters and the use of “.description” to access their underlying descriptions in natural language



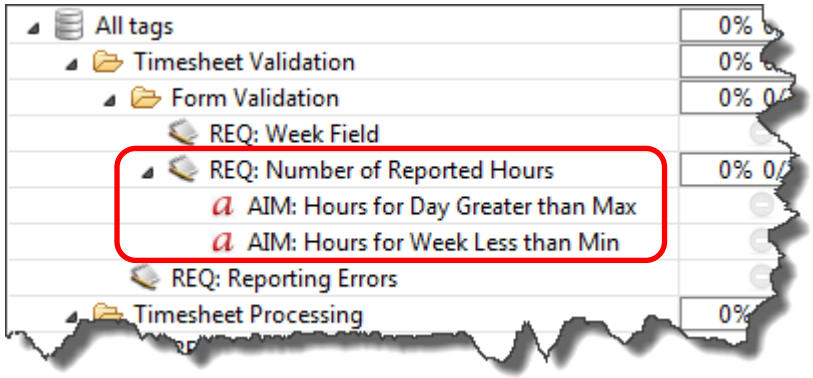
Building the Test Generation Models

9. Associate Test Requirements with the Model

- To associate a requirement with the model:
  - Drag-and-drop the requirement from the imported TOC (imported into the project) to the proper location in the model



- Using AIMs as Refinement of REQs
  - Requirements often too coarse-grained: “AIM” tags are used to provide additional information

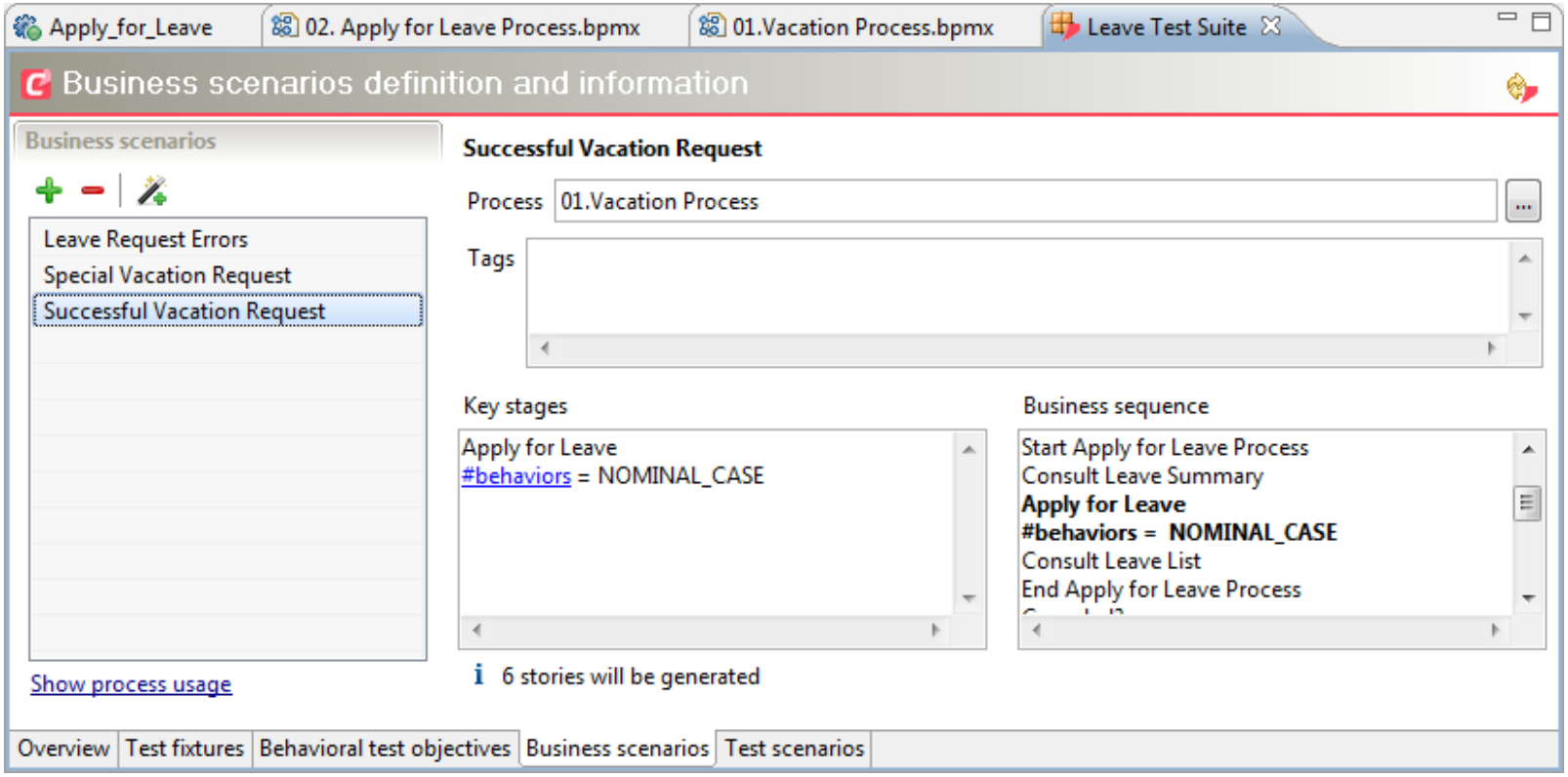


▲ The REQ is now divided into two AIMs. It will be completed when (and only when) the AIMs have been processed.

Building the Test Generation Models

10. Using Test Suites to Select Test Criteria

- Use business scenarios to create test objectives
  - Use dedicated keywords to target specific objectives (`#behaviors` below to select all NOMINAL_CASES specified in decision table)



Building the Test Generation Models

11. Generating Tests

Smartesting CertifyIt 6.0 - LeaveModule [C:\Users\jps\Documents\My Projects\Smartesting Training\eclipse1\LeaveModule]

Project Preferences Help

Run test generation | Generic Excel Publisher

Stories | Tests | Requirements | Priority | Custom

Search stories

Artifacts	Status	Tests
Project		24
Leave Regression Tests		13
Leave Test Suite		11
Leave Request Errors		1
My Scenario		1
Special Vacation Request		1
Special Va		1
Special Va		1
Special Va		1
Special Vacation Request		1
Successful Vacation Request		1
Leave/Apply/Leave Types		1
Leave/Apply/Submitting a Request		1
Annual Leave		1
Valid Request Multiple Days		1
Successful Vacation Request - Annual Leave, Valic		
Successful Vacation Request		1
Successful Vacation Request		1
Successful Vacation Request		
Successful Vacation Request		
Successful Vacation Request		

Test Objective

Generated test

1 : Console

Test detail

Steps

- Default model instance
- Initialized model instance
- leaveModule.ServiceTask1()
- leaveModule.Consult_Leave_Summary(ESS, ONE_PAGE_OR_LESS)
- leaveModule.Apply_for_Leave(ESS, ANNUAL_LEAVE, VALID_REQUEST_MULTIPLE
- leaveModule.Consult_Leave_List(ESS)
- leaveModule.Consult_Leave_Requests_to_Process(MANAGER_PREVIOUS_USER)
- leaveModule.TestNoDescription()
- leaveModule.Approve_or_Reject_Leave(MANAGER_PREVIOUS_USER, APPROVE)

Point of view

Tags activated by the test (bold for current step

REQ:
- **+ Leave/Apply/Leave Types**
- **+ Leave/Apply/Submitting a Request**

AIM:
- **+ Leave/Apply/Leave Types/Annual Leave**
- **+ Leave/Apply/Submitting a Request/Valid Request Multiple Days**
- Annual Leave Approved

Requirements covered by this test

Reached tags | Activated tags | Parameters | Model instance

Building the Test Generation Models

12. Publishing Tests

- Generated tests can be published to most standard test environments (HP ALM, IBM RQM, Microsoft Excel, etc.)

Successful Vacation Request - Annual Leave, Valid Request Multiple Days

Business Action	Logical Data	Values
Multiple sources (see comment)	Standard profile required for this test	ESS (any ESS user)
Multiple sources (see comment)	Special profile required for this test	ADMIN (Admin)

Business Action	Step #	Action	Expected Results	Tags
Consult_Leave_Summary	1	Login as any ESS user.	Your user name is displayed in the page header on the right ("Welcome user_name").	
	2	Select the menu 'Leave > Leave Summary'.	The Leave Summary page. For an ESS, it shows for each leave type, the leave entitlements (in days), the number of days scheduled and taken, and the balance.	
Apply_for_Leave	3	Select the menu 'Leave > Apply'.	The Apply for Leave form is displayed.	CUSTOM: NOMINAL_CASE REQ: Leave/Apply/Leave Types AIM: Annual Leave REQ: Leave/Apply/Submitting a Request AIM: Valid Request Multiple Days
	4	• Select a leave of type 'Congés annuels'. • Then enter valid values in all form fields and a duration of at least two days. • Finally, click 'Apply'.	The message 'Successfully Submitted' is displayed above the Leave Request form.	
Consult_Leave_List	5	Select the menu 'Leave > My Leave'.	The leave request you successfully submitted in the previous step is now visible in the Leave list. The date, leave type and number of days should match what you had entered. The status is set to 'Pending Approval' with the number of days appended to the label. The Select Action list contains only one possible action: 'Cancel'.	
Consult Leave Requests to Process	6	Log out.		
	7	Login as Admin.	Your user name is displayed in the page header on the right ("Welcome user_name"). You should see the complete menu bar: Admin, PIM, Leave, Time, Recruitment, Performance, Help.	
	8	Select the menu 'Leave > Leave List'.		
Approve_or_Reject_Leave	9	In the 'Actions' column for the request to process, select		

Agenda

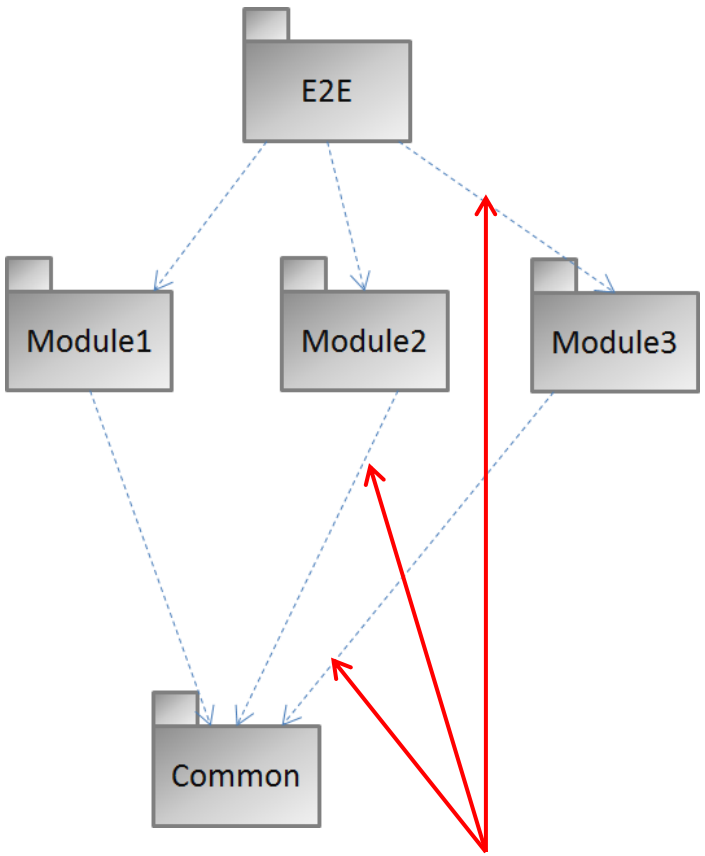
- MBT for Large IT Systems
  - Current challenges of testing large-scale applications
  - Levels of testing addressed by model-based testing
- Building the Test Generation models
  - Understanding and controlling your test requirements
  - Understanding and composing business process models
  - Designing the test generation models
- Reuse and Multi-Model Systems
  - Enabling reuse and collaborative work
  - Structuring models as a layered architecture

Multi-Model Systems and Reuse
1. Introduction to Multi-Model Systems

- Applicability
  - Large applications divided into modules
  - IT systems divided into separate applications
- Each separate module/application becomes of separate test project
  - See next slide for a representative architecture
- Purpose of multi-model systems
  - Need different levels of testing
    - Functional tests at the level of individual applications/modules
    - End-to-end tests involving two or more individual applications/modules
  - Enable collaborative work with minimal impact/cost
  - Enable reuse

Multi-Model Systems and Reuse
2. A Layered Architecture

- Layered Architecture
  - Involves a top-down, hierarchical structure of models
    - Models at one level use only models at lower levels, and
    - Are independent of client models
- Different Types of Projects
  - *E2E* project produces end-to-end tests based on high-level business processes
  - *Module-n* projects are projects that produce functional tests at the application/module level
  - *Common* projects capture enumerations and classes common to several other modules



Dependencies (source depends on targets but not the other way)

Multi-Model Systems and Reuse
3. Reuse

- *Reuse of UML elements*: Common elements captured in a *Common* model project (or more) offer a first level of reuse but it remains limited (relatively few truly reusable elements) (see notes)
- True reuse is found in the *reuse of business processes* and the *reuse of behavioral models* (case of the E2E project)
  - Makes it possible to reuse full model projects "as is" (… when well designed)
  - Imagine for instance systems built around SAP modules or any other ERP…

Multi-Model Systems and Reuse

4. Collaborative Work in a Graphical Environment

- Best Practice: NEVER allow a model to be modified by more than one user at a time
- Recommended to use an architecture with multiple projects
- In all cases, use a version control tool to control access to your separate units (models and other artifacts)
  - The tool must support locking a file before it is modified: only one person at a time can make changes to a given unit
  - Many version-control tools available: open-source (such as CVS and SVN), IBM Rational ClearCase, ...

smartesting®
Optimize your Test Center

Thank you
for your attention

twitter

twitter.com/smartesting

LinkedIn
MBT Group