



🕒 11 Avril 2017

9<sup>ème</sup> édition

# JOURNÉE FRANÇAISE DES TESTS LOGICIELS

**LE TEST LOGICIEL, MAÎTRISER L'ÉTAT  
DE L'ART DU TEST AVEC LE CFTL !**

**ISTQB**  
International Software  
Testing Qualifications Board

# RETOUR D'EXPÉRIENCE DU TEST DU MOTEUR DE COTATION DE LA BOURSE PAN –EUROPÉENNE EURONEXT ET DE SON SI EN MODEL-BASED TESTING





**Thierry Schartle– Practices QA Leader**  
**27 ans d'ancienneté chez Euronext (20 années d'expérience dans la Qualité)**



**Sandra Machon – IT Project Manager**  
**10 ans d'ancienneté chez Euronext (12 années d'expérience dans la Qualité)**

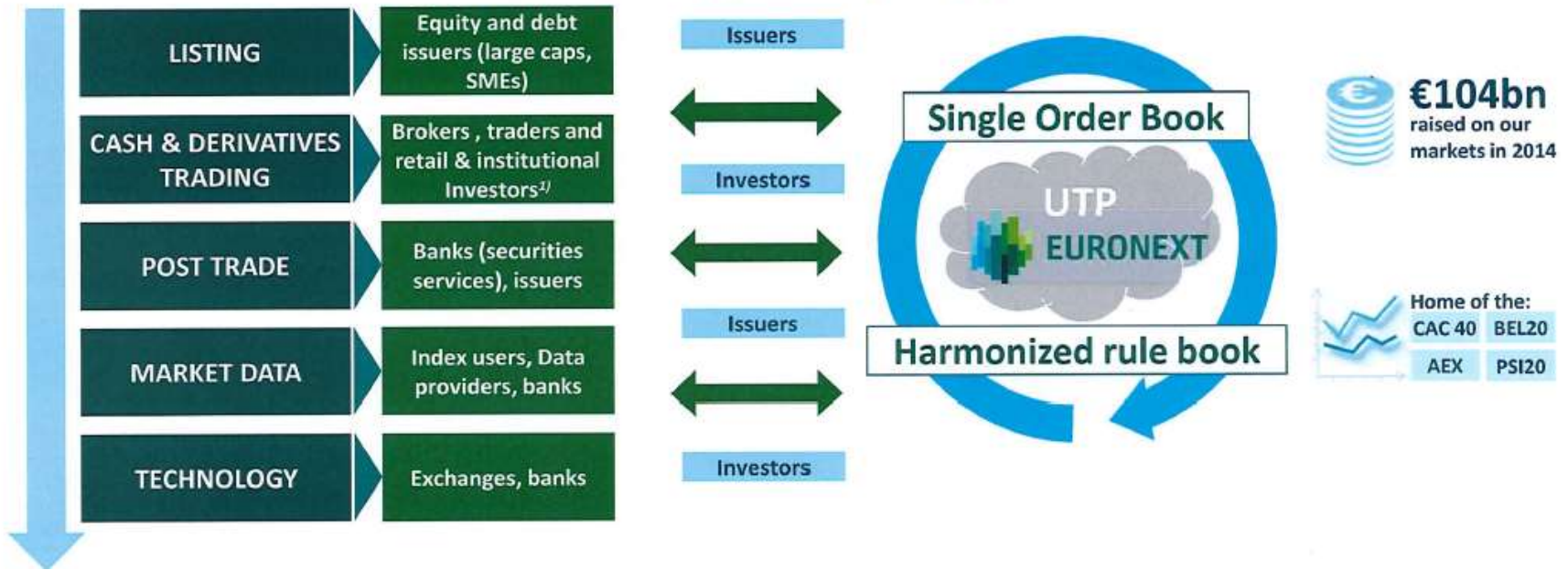
### **Euronext QA Group :**

- **2 Sites Paris and Porto**
- **2-3 personnes par équipes Agile**
- **50 Personnes**

# EURONEXT – THE ONLY PAN- EUROPEAN EXCHANGE FOR THE REAL ECONOMY



AMSTERDAM | BRUSSELS | LISBON | LONDON | PARIS  
 Belfast | Chicago | Hong Kong | Porto



...in multiple countries...

...on a single order book...

...and a common technology



# EURONEXT TECHNOLOGY – KEY FACTS

**113 $\mu$ s** average  
customer roundtrip time  
when trading on UTP  
(on regulated cash markets)

**2.2 billion**  
messages  
per day

**99,96%**  
availability on our  
Equities platform in  
2013 and **100%**  
on Derivatives

Internal matching engine  
roundtrip as low as

**23 $\mu$ s**

**526 member sites and  
700 member lines**  
connected to our systems

A leading-edge technology

1 $\mu$ s = one millionth (10<sup>-6</sup> or 1/1,000,000) of a second



## Contexte avant la mise du Model Based Testing

Phases principales du Cycle QA	Améliorations récentes	Faiblesses	Améliorations nécessaires
Planification, suivi et contrôle	Métriques sur la traçabilité entre les exigences et les cas de tests	La couverture n'est pas correctement mesurée, surtout en ce qui concerne la qualité de notre couverture.	Amélioration du suivi de notre couverture pour pouvoir améliorer la qualité
Analyse et conception des tests	La compréhension des exigences et des objectifs de test, a été améliorée par la mise en place d'Agile. Les cas de tests sont revus.	L'écriture des cas de tests est complètement manuelle et la maintenance en cas de changement d'exigence ou en phase de maintenance projet est très coûteuse et diminue la qualité du référentiel de test. La couverture est limitée par le nombre de cas de tests (et de pas de tests). Il est difficile d'évaluer la qualité de la couverture.	Automatisation de la conception des cas de tests pour maximiser la couverture et gagner en flexibilité quant à la stratégie de la couverture
Implémentation et exécution des tests	Le taux d'automatisation est amélioré en continue.	La première exécution/automatisation est manuelle.	Construire un processus entièrement automatisé
Evaluation des critères de sortie et clôture	Traçabilité	Traçabilité	Traçabilité



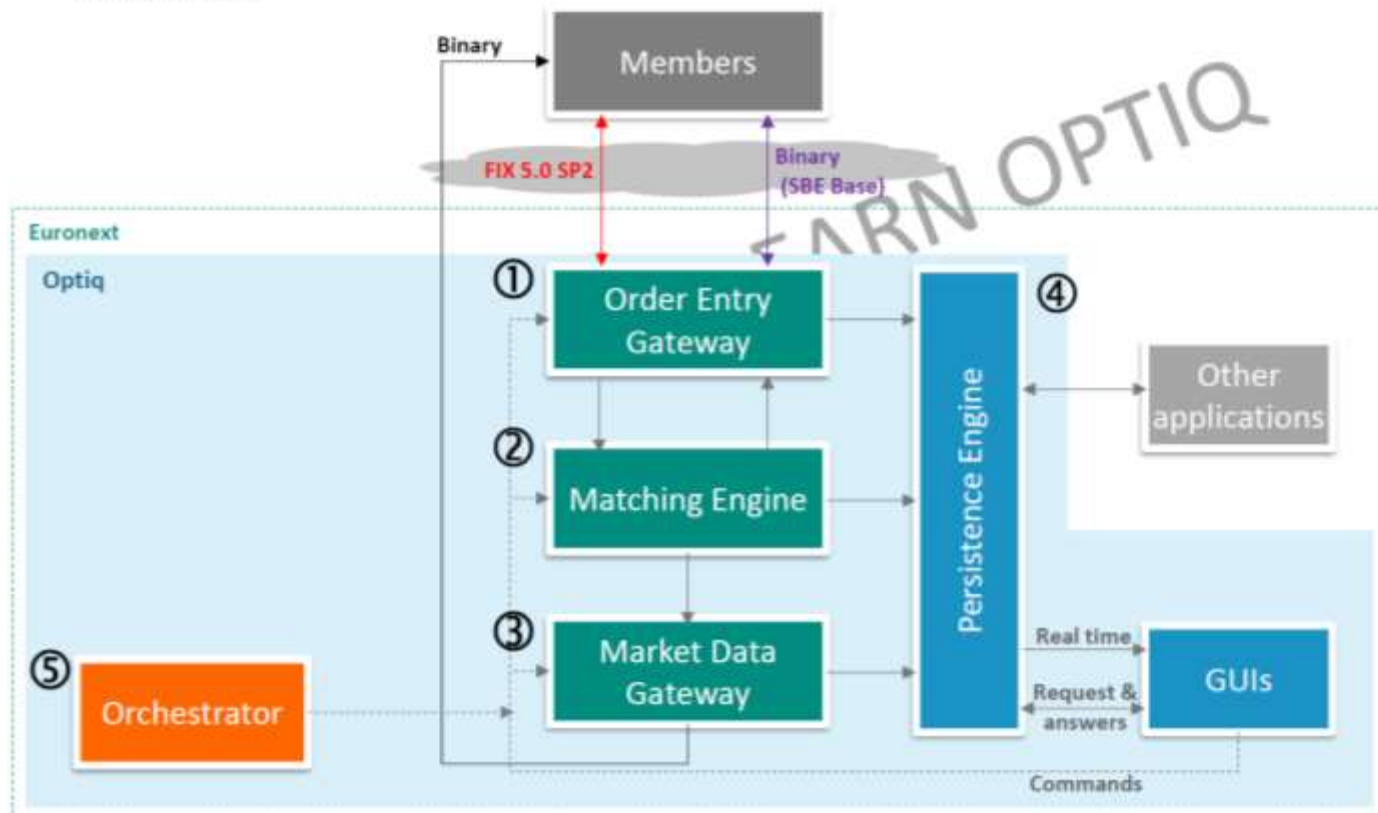
# Comment le Model Based Testing peut répondre aux besoins Business ?

Améliorations nécessaires [Quality + Time To Market]	Model Based Testing Approach	Objectif couvert
Automatisation de la conception des cas de tests pour maximiser la couverture et gagner en flexibilité quant à la stratégie de la couverture	En analysant les user Stories et les specifications, les testeurs élaborent des modèles qui génèrent automatiquement les suites de tests associées à une stratégie de tests	😊
Amélioration du suivi de notre couverture pour pouvoir améliorer la qualité	La traçabilité entre les exigences et les tests est définie au niveau du modèle, les cas de tests générés sont alors liés aux exigences. La qualité des tests est améliorée car les modèles sont revus, ce qui est plus efficace que de revoir x000 cas de tests, et surtout l'objectif de revue est devenue atteignable et fait maintenant partie du process (QA, dev, BA). Le choix de la stratégie de tests permet de prendre en compte la maturité du produit et les risques.	😊
Construire un process entièrement automatisé	Basé sur les modèles, les cas de tests sont générés automatiquement en fonction de la stratégie de tests, avec les scripts d'exécution associés aux résultats attendus, cela permet d'avoir une comparaison automatique des résultats observés et attendus dès la première exécution.	😊
Traçabilité	L'industrialisation de l'ensemble des phases permet de créer un process QA transparent.	😊



## OPTIQ TARGET ARCHITECTURE

**Optiq is composed of 5 core components:** ① the Order Entry Gateway (OEG), ② the Matching Engine (ME), ③ the Market Data Gateway (MDG), ④ the Persistence Engine/GUI and ⑤ the Orchestrator





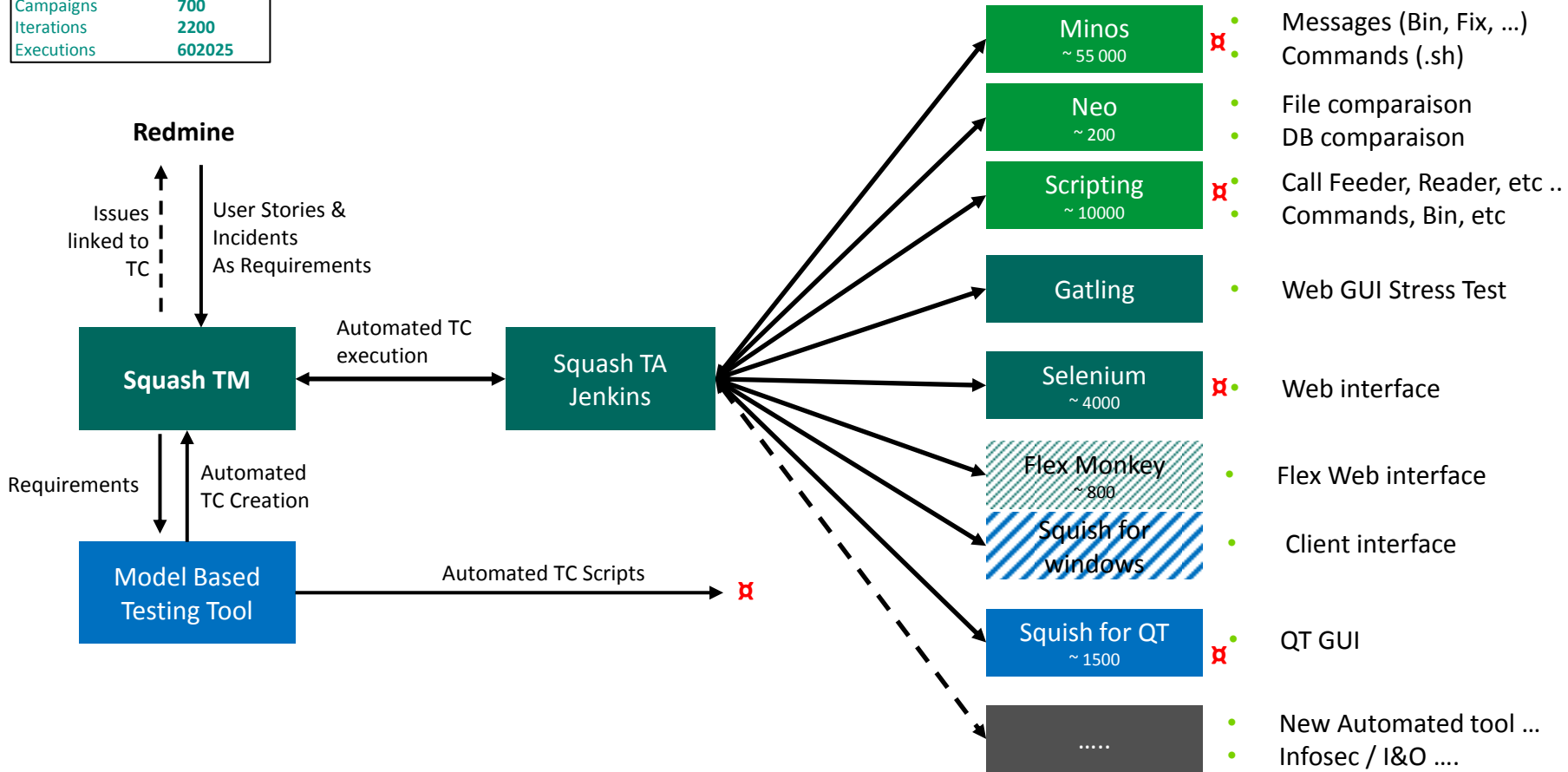
# LE PROGRAMME OPTIQ

	DDM - Data Dictionary Model	WS1 – OEG/ME Gateway messagerie privée Moteur de cotation	WS2- PE/GUI Moteur de persistance des data IHM	WS3 – MD Gateway messagerie publique
<b>Enjeu de la QA</b>	L'ensemble de notre système est basé sur l'échanges de messages, dont les structures sont définies dans le DDM. L'enjeu de la QA a été de mettre en œuvre une solution pour que tous les modèles « parlent » le même langage.	Couverture des algorithmes de trading, selon les différentes modalités d'ordres, les différents évènements, les règles fonctionnelles sous-jacentes.	Couverture sur l'affichage des propriétés selon les données d'entrées (flux ou fichiers), gestion des commandes, tests typiques d'interfaces (search, filters, tree, formulaire, export...). Gestion de la persistance des données.	Vérification du format des messages, avec test des combinaisons, et règles de gestion pour les flux entre les système d'information des traders et le moteur de cotation, gestion du real time et du snapshot.
<b>Organisation QA</b>	QA Manager + 1 QA Analyst de chaque projet	QA Manager + QA Delivery Manager + QA Analysts	QA Manager + QA Delivery Manager + QA Analysts	QA Manager + QA Delivery Manager + QA Analysts
<b>QA Tools</b>	MBT Tool Python scripts	Squash MBT Tool Minos	Squash MBT Tool Squish Producer (Data injector) Consumer (Data consumer) Python scripts	Squash MBT Tool Feeder (Data injector) Reader (Data consumer) Python scripts

# QA TOOLS

Squash TM – Jan 11 <sup>th</sup> 2017	
Projects	160
Users	166
Requirements	2263
Test cases	134074
Campaigns	700
Iterations	2200
Executions	602025

Squash TM – Jan 11 <sup>th</sup> 2017	
Valid TC	94 175
TRASH FOLDER	40880
Automated	64 %



# NOTRE IMPLÉMENTATION MBT

## LA MISE EN PLACE

- **Gestion de l'implémentation technique et des releases de l'outil MBT par notre équipe IT Tools**
- **Mise en place de deux Virtual Machines, chaque test analyst accède à l'une ou l'autre sur son propre workspace.**
- **Mise en place des formations avec l'éditeur MBT pour Paris et Belfast**
- **Accompagnement sur notre site par un expert outil MBT pendant 6 mois**
- **Lancement de l'utilisation de l'outil MBT sur le programme OPTiq (refonte de notre système, avec focus sur la chaîne de trading)**



# NOTRE IMPLÉMENTATION MBT - 1

## LE MODEL BASED TESTING POUR LE PROGRAMME OPTIQ

- **Quels modèles MBT en face de l'organisation du programme ?**
  - Choix : Un modèle pour une application
- **Et les tests d'intégration ?**
  - Choix : Intégration de chaque modèle MBT dans un modèle d'intégration
- **Comment gérer l'implication de plusieurs testeurs sur un même projet, sur un même modèle ?**
  - Choix : gestion des sujets isolés par chaque test analyst, puis merge des projets pour constituer un seul modèle qui portera l'ensemble du périmètre (avec gestion des versions sous GIT)

# NOTRE IMPLÉMENTATION MBT - 2

## ORGANISATION DU QA FRAMEWORK AVEC L'OUTIL MBT

- **L'organisation s'est faite par le biais de workshops hebdomadaires impliquant l'ensemble des utilisateurs. Les premiers risques identifiés portent sur notre capacité à modéliser et à maintenir la polyvalence de nos analystes de test (ie interventions de nos analystes de test sur l'ensemble du périmètre sous test).**
- **L'objectif principal est de synchroniser au maximum l'implémentation faite sur chaque projet par le biais :**
  - Des bonnes pratiques
  - D'une utilisation commune des structures et data
  - Du développement et du partage des compétences de modélisation
  - Du suivi par les QA Managers sur chaque projet

# NOTRE IMPLÉMENTATION MBT - 3

## DE L'ANALYSE DES EXIGENCES JUSQU'AU REPORTING DES CAMPAGNES DE TEST

- Exigences de chaque projet saisies dans Redmine (User Stories) avec import automatique dans Squash, puis import des exigences de Squash vers l'outil MBT.
- La modélisation porte le lien entre les transitions et les exigences, puis les cas de tests générés et importés vers Squash permettent la traçabilité des exigences et le suivi de la couverture dans Squash.
- L'outil MBT génère la description des cas de tests (importés dans Squash sous forme de HTML) ainsi que les scripts de tests (pour les stimulations et les résultats attendus) selon l'outil d'exécution/automatisation utilisé.

# REFERENTIAL MBT

## FROM XML TO DATA IN THE MBT TOOL

```

<enx:structure enx:numId="721" enx:name="MDBOrderUpdateData repl">
  <enx:description>Repeating sub-structure of Order Update Event.</enx:description>
  <enx:structItem enx:name="FirmID">
    <enx:fieldlink enx:fieldID="Firm ID"/>
  </enx:structItem>
  <enx:structItem enx:name="SymbolIndex">
    <enx:fieldlink enx:fieldID="Symbol Index"/>
  </enx:structItem>
  <enx:structItem enx:name="ActionType">
    <enx:fieldlink enx:fieldID="Market Data Action Type"/>
  </enx:structItem>
  <enx:structItem enx:name="OrderPriorityTime">
    <enx:fieldlink enx:fieldID="Order Priority"/>
  </enx:structItem>
  <enx:structItem enx:name="OriginalOrderPriority">
    <enx:fieldlink enx:fieldID="Original Order Priority"/>
  </enx:structItem>
  <enx:structItem enx:name="OrderType">
    <enx:fieldlink enx:fieldID="Order Type"/>
  </enx:structItem>
  <enx:structItem enx:name="OrderPx">
    <enx:fieldlink enx:fieldID="Order Price"/>
  </enx:structItem>
  <enx:structItem enx:name="OrderSide">
    <enx:fieldlink enx:fieldID="Order Side"/>
  </enx:structItem>
  <enx:structItem enx:name="Quantity">
    <enx:fieldlink enx:fieldID="Quantity"/>
  </enx:structItem>
</enx:structure>

<enx:structure enx:numId="722" enx:name="MDBOrderUpdateData">
  <enx:description>MDBOrderUpdateData</enx:description>
  <enx:structItem enx:name="">
    <enx:structlink enx:structID="MDBHeader"/>
  </enx:structItem>
  <enx:structItem enx:name="EventTime">
    <enx:fieldlink enx:fieldID="Event Time"/>
  </enx:structItem>
  <enx:structItem enx:maxOccurs="255" enx:name="Orders" enx:occurrenceCounter="Yes">
    <enx:structlink enx:structID="MDBOrderUpdateData repl"/>
  </enx:structItem>
</enx:structure>

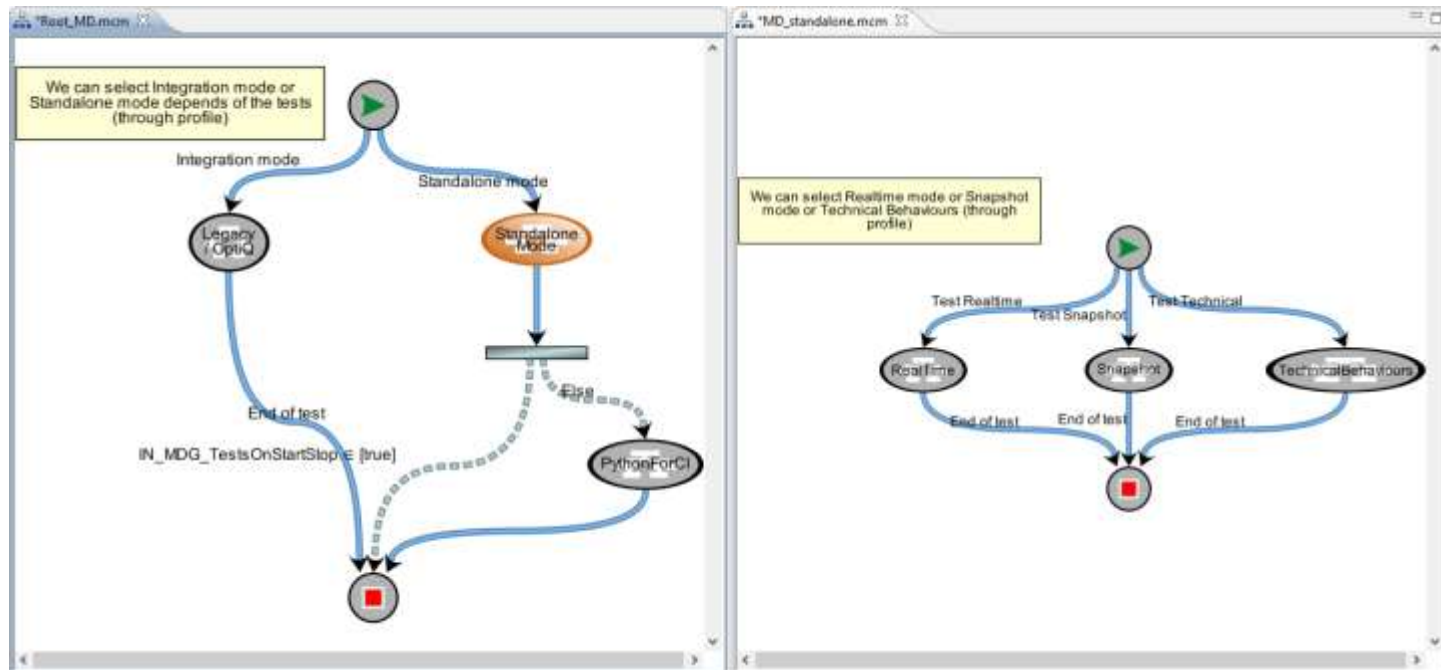
```



Structure	MDPO...
TYPE_INTERNAL_EVENTS_MDBOrderUpdateData	TYPE_INTER...
name	name
MDBIdentifier	MDBI...
SnapshotIndicator	Snaps...
EMM	Euronext Mar...
CentralOrderBook(COB)	1
NAVTradingFacility	2
PrimaryMarket	3
Wholesales	4
OffBook(TCS)	5
OTC(TCS)	6
OffBook(AtomX)	7
SocieteGeneraleSystematicInternaliser(SI)	50
NotApplicable(ForindexesandNAV)	254
EventId	EventId
BookOUTTime	Book...
MDBINTime	MDBI...
MDBOUTTime	MDBO...
EventTime	Event...
Orders	Orders
FirmID	FirmID
SymbolIndex	Symb...
ActionType	Identifies if th...
NewOrder	1
DeletionOfIdentifiedOrder	2
DeletionOfAllOrdersBySide	3
ModificationOfExistingOrder	4
RetransmissionOfAllOrders	5
OrderPriorityTime	Order...
OriginalOrderPriority	Origin...
OrderType	Type of Order.
Market	1
Limit	2
Pegged	5
Markettolimit/Marketonopening	6
OrderPx	OrderPx
OrderSide	Indicates the ...
Buy	1
Sell	2
Cross	3
Quantity	Quan...

# MARKET DATA

- 2 modes de fonctionnement : Integration, Standalone. Zoom sur Standalone >> Real Time



Modélisation

Génération

Export Squash

Génération script

Export et exécution  
Banc de Test



# MARKET DATA

- Zoom sur la chaîne « RealTime » gérant les messages valides en se basant sur les classes d'équivalence

Sélection du type de message en se basant sur les classes d'équivalence

The screenshot shows the 'Distribution' window for 'IN\_MDG\_MSGTYPE'. The 'Properties' section shows Name: IN\_MDG\_MSGTYPE, Type: Input, Data Type: Enumerated. The 'Distribution' section shows Distribution kind: Rectangular. Below is a table of distribution ranges:

Name	Value	Frequency	Probability
EC0	OrderUpdate	Normal	0.334
EC1	MarketUpdate	Normal	0.333
EC2	Trade	Never	0
EC3	PriceUpdate	Normal	0.333
EC4	MarketStatusChange	Never	0

Validation des valeurs possibles basée sur les classes d'équivalence

The screenshot shows the 'Distribution' window for 'ExecutionID'. The 'Properties' section shows Name: ExecutionID, Type: Input, Data Type: Numerical, Unit: 429487295. The 'Distribution' section shows Distribution kind: Rectangular. Below is a table of distribution ranges:

Name	Include	Lower Bound	Upper Bound	Include	Frequency	Probability
EC0	[	0 (MPO)	100	]	Normal	0.5
	[	100	100	]	Never	0
	[	100	429487295 (MAO)	]	Normal	0.5

Modélisation

Génération

Export Squash

Génération script

Export et exécution  
Banc de Test

# MARKET DATA

- Zoom sur la chaîne « RealTime » gérant les messages valides

## Paramétrage des opérations de test associées sur les transitions

[-] RepeatedGroup					
[-] MDG_Python_MessageTyperep1-ForFeederForInternalPE-part2	MDG_Python_M...				
➔ FirmID		Transition Input	012 IN_INTERNA...	Value	
➔ ActionType		Transition Input	<E> IN_INTERNA...	Item Value	
➔ OrderPriorityTime		Transition Input	012 IN_INTERNA...	Value	
➔ OrderType		Transition Input	<E> IN_INTERNA...	Item Value	
➔ OrderPrice		Transition Input	012 IN_INTERNA...	Value	
➔ OrderSide		Transition Input	<E> IN_INTERNA...	Item Value	
➔ Quantity		Transition Input	012 IN_INTERNA...	Value	

## Ci-dessous le script décrit dans l'outil MBT

```
if(messageType == "OrderUpdate" and (tool == "Feeder" or tool == "PEIn")):  
    part2 = "{FirmID=${FirmID},SymbolIndex=${SymbolIndex},ActionType=${ActionType},OrderPriorityTime=${OrderPriorityTime},OriginalOrderPriority=${OriginalOrderPriority},OrderType=${OrderType},Or  
    fichier.write(part2)  
elif(messageType == "MarketUpdate" and (tool == "Feeder" or tool == "PEIn")):  
    part2 = "{UpdateType=${UpdateType},SymbolIndex=${SymbolIndex},Price=${OrderPrice},Quantity=${Quantity} }"  
    fichier.write(part2)  
elif(messageType == "Trade" and (tool == "Feeder" or tool == "PEIn")):  
    part2 = "{TradeTime=${TradeTime},ExecutionID=${ExecutionID},FirmID=${FirmID},MemberID2=${MemberID2},TradeType=${TradeType},TradeQualifier=${TradeQualifier},Price=${OrderPrice},Quantity=${Quantit  
    fichier.write(part2)  
elif(messageType == "PriceUpdate" and (tool == "Feeder" or tool == "PEIn")):  
    part2 = "{PriceType=${PriceType},SymbolIndex=${SymbolIndex},Price=${OrderPrice},Quantity=${Quantity},InbalanceQty=${InbalanceQty},InbalanceQtySide=${InbalanceQtySide} }"  
    fichier.write(part2)  
elif(messageType == "MarketStatusChange" and (tool == "Feeder" or tool == "PEIn")):  
    part2 = "{ChangeType=${ChangeType},SymbolIndex=${SymbolIndex},EventTime=${EventTime},PhaseType=${PhaseType},StatusReason=${StatusReason},TradingMode=${TradingMode},OrderEntryQualifier=${Order  
    fichier.write(part2)
```

Modélisation

Génération

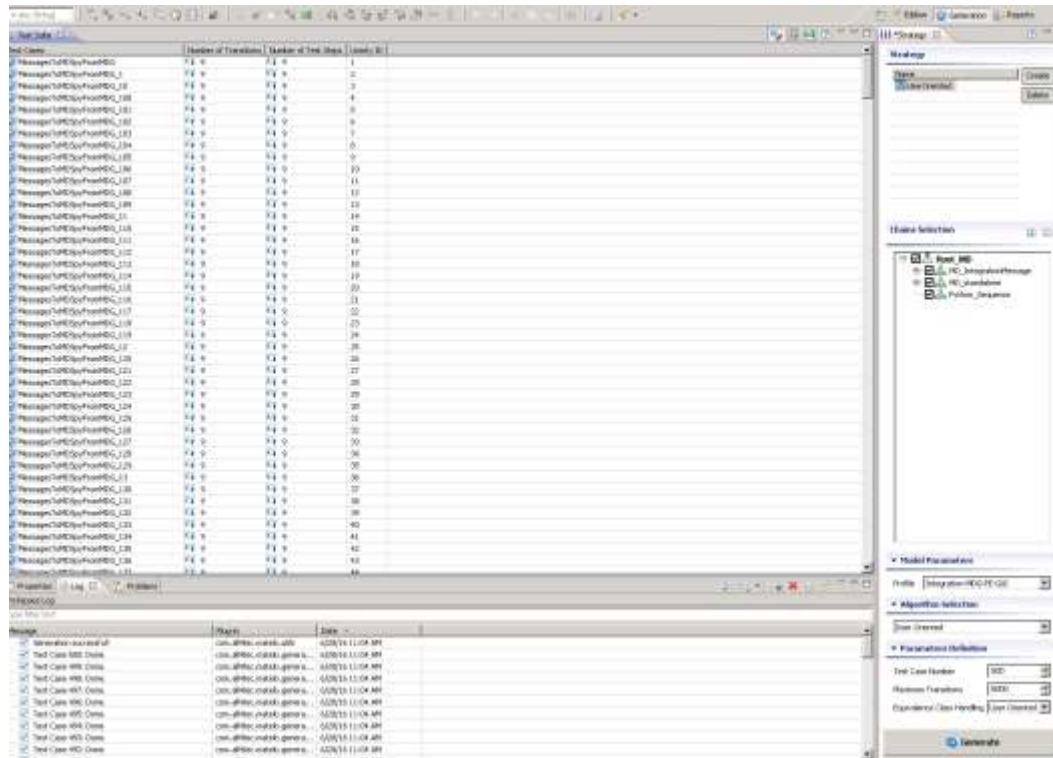
Export Squash

Génération script

Export et exécution  
Banc de Test

# MARKET DATA

- Génération de cas de test



Modélisation

Génération

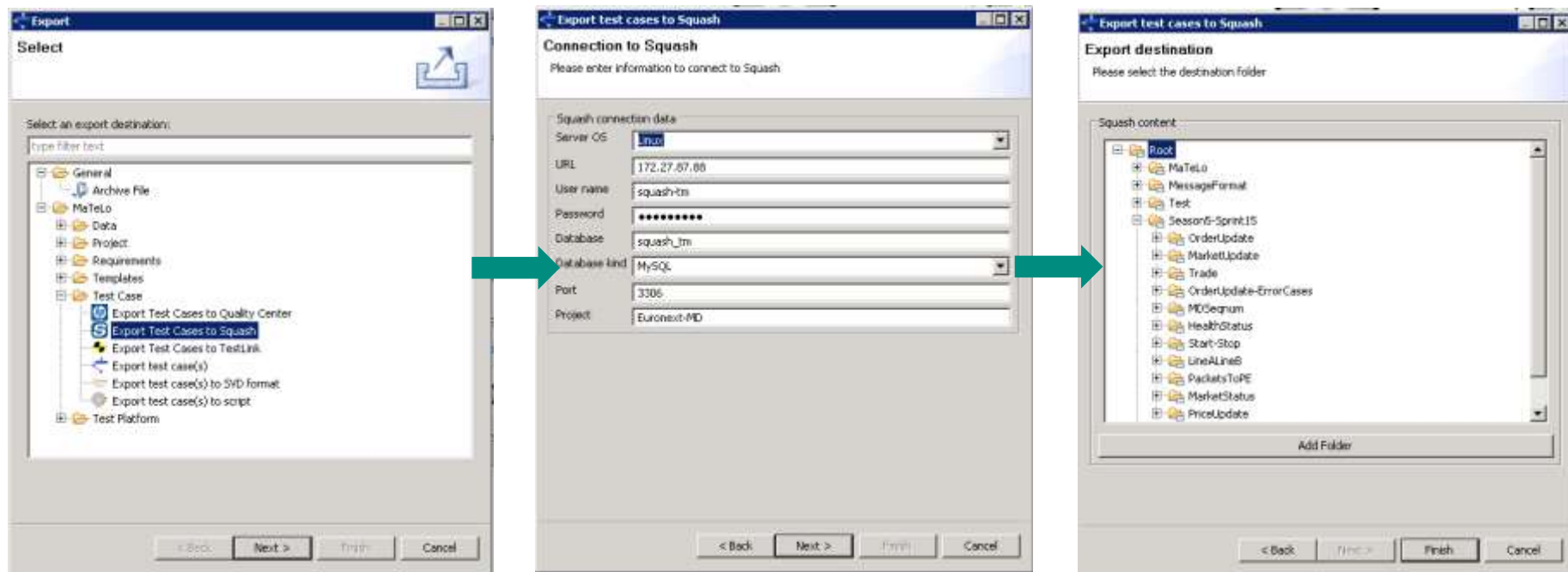
Export Squash

Génération script

Export et exécution  
Banc de Test

# MARKET DATA

- Export des cas de test vers Squash afin de faire le lien entre les cas de test générés et les exigences



Modélisation

Génération

Export Squash

Génération script

Export et exécution  
Banc de Test

# MARKET DATA

- Visualisation des exigences liées aux cas de test

The screenshot shows the 'Espace Exigences' application. The main window displays a requirement titled 'Exigence : fr-15\_20000179-optiq-ws3-md-gtw#40914 - Start - Simple and clean start up'. The requirement is created on 24/03/2016 20:47 and is currently in the '1-En cours de rédaction' status. The description states it was created automatically from a Radmine ticket. Below the requirement, a table lists test cases that verify this requirement:

#	Projet	Reference	Cas de test	Type
1	Euronext-MD		WhenAllMessagesAreAck	manual
2	Euronext-MD		WhenAllMessagesAreNotAck	manual
3	Euronext-MD		WhenNoMessageOccured	manual

Modélisation

Génération

Export Squash

Génération script

Export et exécution  
Banc de Test

# MARKET DATA

- Pour chaque cas de test de l'outil MBT exporté au format « script », un fichier « python » est généré et exécuté sur l'environnement de QA.

```
Stop -> mdreader
Stop -> mdspy(t)
Run PE consumer with -> (release) for feeder /app/mqm/scripts_qa/tc_from_PE/peFeeder_OrderUpdateK0_BadUpdateCount.txt
Run PE consumer with -> (release) for reader /app/mqm/scripts_qa/tc_from_PE/peReader_OrderUpdateK0_BadUpdateCount.txt
Run PE consumer with -> (release) for spy /app/mqm/scripts_qa/tc_from_PE/peSpyA_OrderUpdateK0_BadUpdateCount.txt
Stop -> consumers Feeder / Reader / Spy
Check Reader with -> (release) /app/mqm/scripts_qa/tc_from_MaTeLo_for_Reader/Reader_OrderUpdateK0_BadUpdateCount.txt
[ OK ] for Reader
Check Spy with -> (release) /app/mqm/scripts_qa/tc_from_MaTeLo_for_Spy/SpyA_OrderUpdateK0_BadUpdateCount.txt
[ OK ] for Spy line A
Check PE with -> (release) Reader /app/mqm/scripts_qa/tc_from_MaTeLo_for_PE_Reader/peReader_OrderUpdateK0_BadUpdateCount.txt (line R)
[ OK ] for PE from Spy line A
Check PE with -> (release) Feeder /app/mqm/scripts_qa/tc_from_MaTeLo_for_PE_Feeder/peFeeder_OrderUpdateK0_BadUpdateCount.txt (line F)
[ OK ] for PE from Feeder
Check PE with -> (release) Spy /app/mqm/scripts_qa/tc_from_MaTeLo_for_Spy/peSpyA_OrderUpdateK0_BadUpdateCount.txt (line A)
[ OK ] for PE from Spy line A
Check MDSeqNum with -> (release) /app/mqm/scripts_qa/tc_from_Reader/FromReader_OrderUpdateK0_BadUpdateCount.txt_se on Reader (line R)
[ OK ] for SeqNum on Reader
Check MDSeqNum with -> (release) /app/mqm/scripts_qa/tc_from_Spy_LineA/FromSpy_OrderUpdateK0_BadUpdateCount.txt_se on Spy (line A)
[ OK ] for SeqNum on Spy on line A

=====
Test case #5 -> OrderUpdateK0_UpdateCountNull_20180028_061135415_test_case.py
wateToTestCodeId -> (release) wB5x2BzEaaCN5P*HX0A
Run Spy line A with -> (release) /app/mqm/scripts_qa/tc_from_Spy_LineA/FromSpy_OrderUpdateK0_UpdateCountNull.txt_se
Run Reader without Packet with -> (release) /app/mqm/scripts_qa/tc_from_Reader/FromReader_OrderUpdateK0_UpdateCountNull.txt_se
Run Feeder with -> (release) /app/mqm/scripts_qa/tc_from_MaTeLo_for_Feeder/Feeder_OrderUpdateK0_UpdateCountNull.txt
Stop -> main-ct
Stop -> mdreader
Stop -> mdspy(t)
Run PE consumer with -> (release) for feeder /app/mqm/scripts_qa/tc_from_PE/peFeeder_OrderUpdateK0_UpdateCountNull.txt
Run PE consumer with -> (release) for reader /app/mqm/scripts_qa/tc_from_PE/peReader_OrderUpdateK0_UpdateCountNull.txt
Run PE consumer with -> (release) for spy /app/mqm/scripts_qa/tc_from_PE/peSpyA_OrderUpdateK0_UpdateCountNull.txt
Stop -> consumers Feeder / Reader / Spy
Check Reader with -> (release) /app/mqm/scripts_qa/tc_from_MaTeLo_for_Reader/Reader_OrderUpdateK0_UpdateCountNull.txt
[ OK ] for Reader
Check Spy with -> (release) /app/mqm/scripts_qa/tc_from_MaTeLo_for_Spy/SpyA_OrderUpdateK0_UpdateCountNull.txt
[ OK ] for Spy line A
Check PE with -> (release) Reader /app/mqm/scripts_qa/tc_from_MaTeLo_for_PE_Reader/peReader_OrderUpdateK0_UpdateCountNull.txt (line R)
[ OK ] for PE from Spy line A
Check PE with -> (release) Feeder /app/mqm/scripts_qa/tc_from_MaTeLo_for_PE_Feeder/peFeeder_OrderUpdateK0_UpdateCountNull.txt (line F)
[ OK ] for PE from Feeder
Check PE with -> (release) Spy /app/mqm/scripts_qa/tc_from_MaTeLo_for_Spy/peSpyA_OrderUpdateK0_UpdateCountNull.txt (line A)
[ OK ] for PE from Spy line A
Check MDSeqNum with -> (release) /app/mqm/scripts_qa/tc_from_Reader/FromReader_OrderUpdateK0_UpdateCountNull.txt_se on Reader (line R)
[ OK ] for SeqNum on Reader
Check MDSeqNum with -> (release) /app/mqm/scripts_qa/tc_from_Spy_LineA/FromSpy_OrderUpdateK0_UpdateCountNull.txt_se on Spy (line A)
[ OK ] for SeqNum on Spy on line A
```

Modélisation

Génération

Export Squash

Génération script

Export et exécution  
Banc de Test

# MARKET DATA

- Exécution des scripts dans un environnement de test virtualisé
- Mise à jour des résultats de l'exécution des test dans Squash

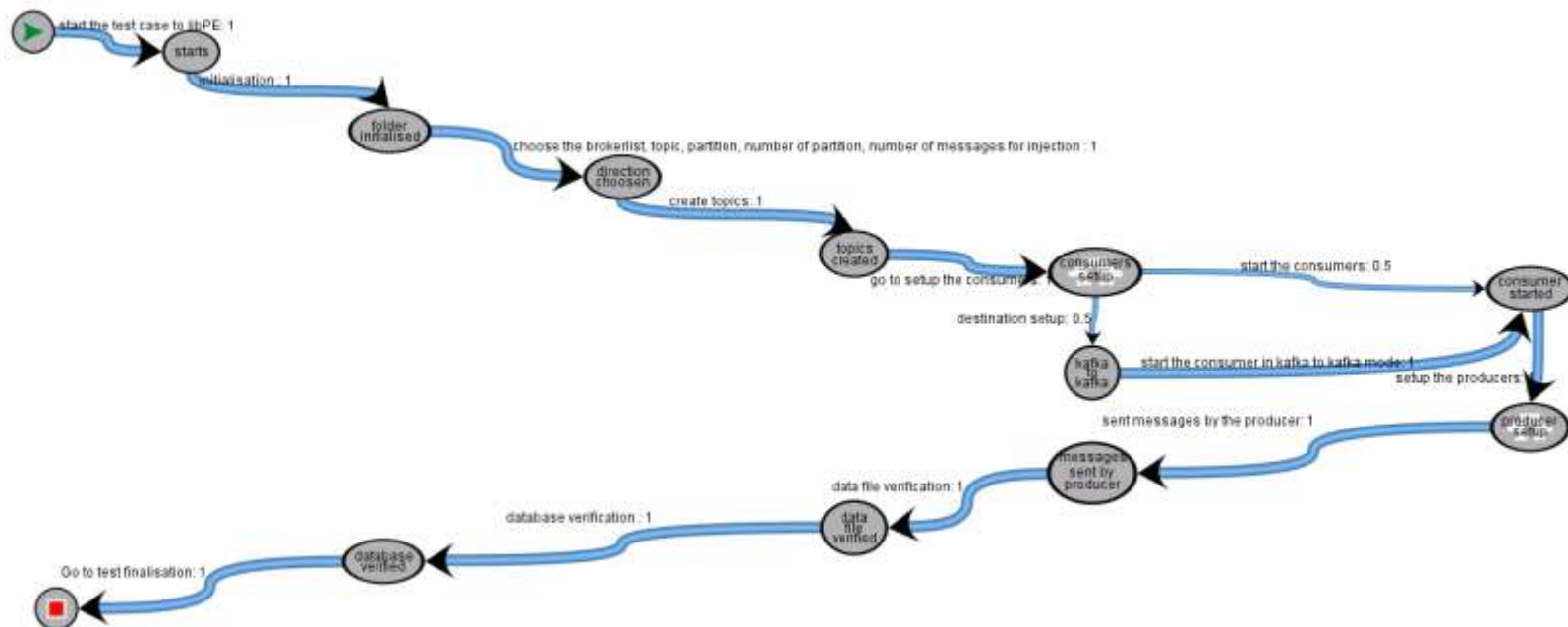
The screenshot displays the Squash CI web interface for an iteration named 'OrderUpdate'. The interface includes a sidebar with a project tree, a main header with navigation buttons, and a table of test results. The table columns are: #, Projet, Référence, Cas de test, Importance, Jeux de données, Suite de tests, Statut, Utilisateur, and Date d'exécution. All 12 test cases listed are in a 'succès' (success) status, executed by 'CRichardot' on 27/06/2016 at 09:41.

#	Projet	Référence	Cas de test	Importance	Jeux de données	Suite de tests	Statut	Utilisateur	Date d'exécution
1	Euronext-MD	_F-vkoCP0EeaI_7-hzPLODw	OrderUpdateOK_ValidField	Faible	-	FeederToPE	succès	CRichardot	27/06/2016 09:41
2	Euronext-MD	_F-vkoP0EeaI_7-hzPLODw	OrderUpdateOK_ValidField_1	Faible	-	FeederToPE	succès	CRichardot	27/06/2016 09:41
3	Euronext-MD	_F-vktCP0EeaI_7-hzPLODw	OrderUpdateOK_ValidField_10	Faible	-	FeederToPE	succès	CRichardot	27/06/2016 09:41
4	Euronext-MD	_F-vkbP0EeaI_7-hzPLODw	OrderUpdateOK_ValidField_11	Faible	-	FeederToPE	succès	CRichardot	27/06/2016 09:41
5	Euronext-MD	_F-vkuCP0EeaI_7-hzPLODw	OrderUpdateOK_ValidField_12	Faible	-	FeederToPE	succès	CRichardot	27/06/2016 09:41
6	Euronext-MD	_F-vkuP0EeaI_7-hzPLODw	OrderUpdateOK_ValidField_13	Faible	-	FeederToPE	succès	CRichardot	27/06/2016 09:41
7	Euronext-MD	_F-vkvCP0EeaI_7-hzPLODw	OrderUpdateOK_ValidField_14	Faible	-	FeederToPE	succès	CRichardot	27/06/2016 09:41
8	Euronext-MD	_F-vkvP0EeaI_7-hzPLODw	OrderUpdateOK_ValidField_15	Faible	-	FeederToPE	succès	CRichardot	27/06/2016 09:41
9	Euronext-MD	_F-vkwCP0EeaI_7-hzPLODw	OrderUpdateOK_ValidField_16	Faible	-	FeederToPE	succès	CRichardot	27/06/2016 09:41
10	Euronext-MD	_F-vkwP0EeaI_7-hzPLODw	OrderUpdateOK_ValidField_17	Faible	-	FeederToPE	succès	CRichardot	27/06/2016 09:41
11	Euronext-MD	_F-vkxCP0EeaI_7-hzPLODw	OrderUpdateOK_ValidField_18	Faible	-	FeederToPE	succès	CRichardot	27/06/2016 09:41
12	Euronext-MD	_F-vkxP0EeaI_7-hzPLODw	OrderUpdateOK_ValidField_19	Faible	-	FeederToPE	succès	CRichardot	27/06/2016 09:41



# PERSISTENCE ENGINE

- Objectifs : test de réplication, test de non corruption de données, test des plugins et services



Modélisation

Génération

Export Squash

Génération script

Export et exécution  
Banc de Test



# PERSISTENCE ENGINE

- Exemple de transition

Description	Profiling	Attributes	Test Paths	Requirements	Treatment Functions	Test Platforms	
<b>Test operation scheduling</b>							
Name	Calling	Value Category	Value From	Value	Description		
PEScripting							
open_consumer_on_kafka	open_consumer...						
brokerlist		Value	User	A			
topic		Value	User	A			
partition		Value	User	A			
filefolder		Value	User	A			
loglevel		Value	User	A			
plugin		External Input	PE_inp_cons...	Value			
service		External Input	PE_inp_cons...	Value			
stringforservice		External Input	PE_inp_cons...	Value			
messagetype		External Input	PE_inp_cons...	Value			

*#open\_consumer\_on\_kafka (brokerlist, topic, partition, filefolder, loglevel, mokorkafka, nbmessages, plugin, service, stringforservice, messagetype)*

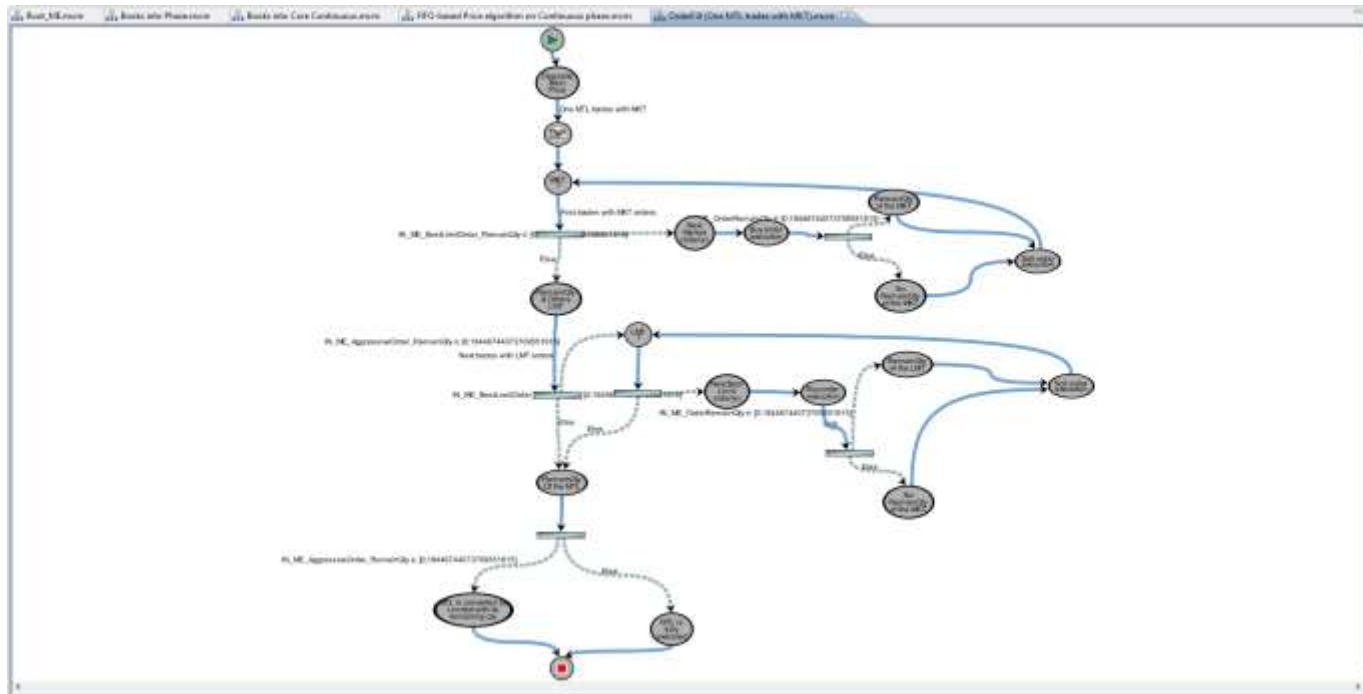
*PE\_python\_script.open\_consumer\_on\_kafka (brokerlist, topic, partition, filefolder, loglevel, "\$plugin", "\$service", "\$stringforservice", \$messagetype)*

➔ A la fin, nous obtenons des scripts qui seraient **facilement compréhensibles fonctionnellement**



# MATCHING ENGINE

- Execution d'un ordre « Market To Limit » dans un carnet ayant des ordres « Market » et/ou « Limited » >> modélisation déterministe par classe d'équivalences



Modélisation

Génération

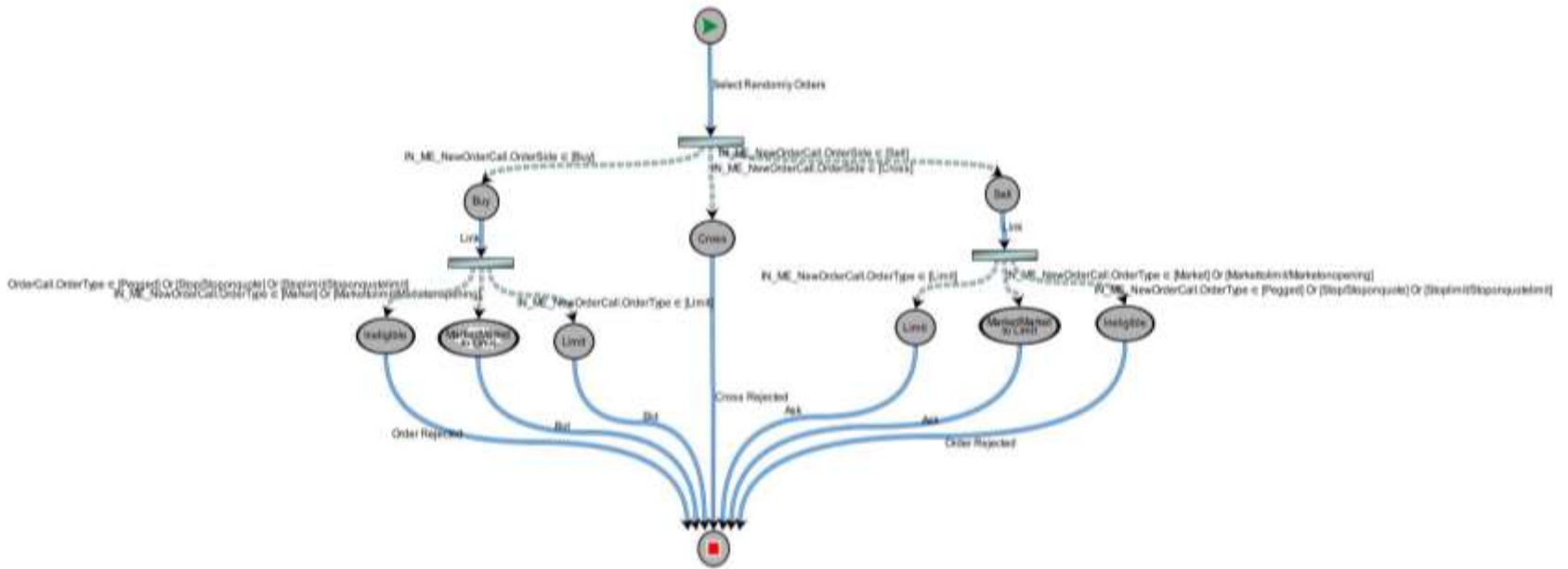
Export Squash

Génération script

Export et exécution  
Banc de Test

# MATCHING ENGINE

- Exemple de modélisation d'un algorithme >> modélisation du comportement du système avec des stimulations aléatoires et la gestion des règles déterminant les résultats attendus



Modélisation

Génération

Export Squash

Génération script

Export et exécution  
Banc de Test

# MATCHING ENGINE

- Exemple d'utilisation d'une opération de test

The screenshot displays a software development tool interface. The top-left pane shows a complex test model with various nodes and connections. The top-right pane shows a script editor with XML code for a test case. The bottom pane shows a table of test operations.

```
<?xml version="1.0" encoding="UTF-8" ?>
<DOCUMENT>
  <MINOS>
    <!-- $@Step:Webster -->
    <CYCLERS>
      <project_name>@ProjectName/@project_name>
      <case>@CaseName/@case>
    </CYCLERS>
  </MINOS>
  <TEST>
    <headers>
      <tc_name>@TestCaseName/@tc_name>
      <dir>@DirName/@dir>
      <description>@TestCaseDescription/@description>
    </headers>
    <body>
```

Name	Calling	Value Category	Value From	Value	Description
Minos					
MinosInformation_FIFO1	MinosInformation_FIFO1:Test Me...				
ProjectName		Value	User	ik Test Me/fo...	
UseName		Value	User	ik roch...	
DirName		Value	User	ik FIFO-based...	
TestCaseName		Test Case	Generated Test...	Name	
TestSequenceLogon_FIFO		Chain	FIFO-based Fir...	Name	
HeaderMinosEncoding_FIFO1	HeaderMinosEncoding_FIFO1:CT...				
ABE_header58C_MH	ABE_header58C_ENCST_MINOS_LE...				
DEG_Logon_FIFO	DEG_Logon_FIFO:CT_MINOS_NO...				

Fichier « script » correspondant à l'opération de test « MinosInformation\_FIFO1 »

Association de l'opération de test avec remplissage des paramètres

Modélisation

Génération

Export Squash

Génération script

Export et exécution  
Banc de Test

# MATCHING ENGINE

The screenshot displays the Matcha v4.4.2 software interface. At the top, there is a menu bar and a toolbar. The main workspace is divided into several sections:

- Test Case Configuration:** A table with columns for Name, ID, and Description. The first row shows 'Send & Received on OSG by ECHEFFO4' with ID '181'. Below this, there are two horizontal bars representing 'Simulation' (pink) and 'Verification' (blue).
- State Transition Diagram:** A central diagram showing a complex network of nodes and edges, representing the state space of the system being modeled.
- Test Case Details:** A table with columns for Name, ID, and Description. The first row shows 'New ML' with ID '181'. The second row shows 'MUT?' with ID '182'. The third row shows 'Switch' with ID '181'. The fourth row shows 'New Model in MUT?' with ID '181'. Each row has a 'Simulation' bar (pink) and a 'Verification' bar (blue).
- Test Case Parameters:** A table with columns for Name, ID, and Description. The first row shows 'RUB\_OrderPB' with ID '10000000'. The second row shows 'OUT\_ME\_OrderPB' with ID '10000000'. The third row shows 'LastTransBy' with ID '5'. The fourth row shows 'LastState' with ID '5'. The fifth row shows 'LastOrderBy' with ID '5'. The sixth row shows 'OrderDate' with ID '5'. The seventh row shows 'LiquidityOrder' with ID '5'. The eighth row shows 'LiquidityOrder' with ID '5'. The ninth row shows 'LiquidityOrder' with ID '5'. The tenth row shows 'LiquidityOrder' with ID '5'.

Modélisation

Génération

Export Squash

Génération script

Export et exécution  
Banc de Test

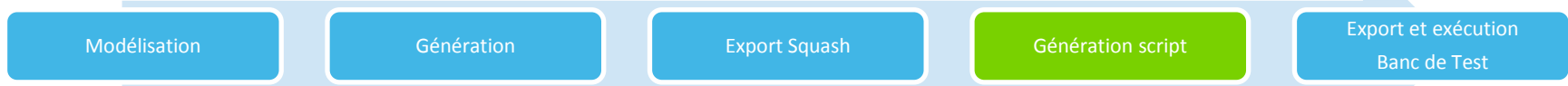
# MATCHING ENGINE

Génération de script par l'outil MBT et import dans notre injecteur de test

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <documents>
3 <document>
4 <GATELO>
5 <!-- 35 - Empty Book -->
6 <FOLDERS>
7 <project_name>Test MaTelo Project</project_name>
8 <user>web</user>
9 </FOLDERS>
10 <TC>
11 <header>
12 <to_name>HE_FIP01_BuyMFL_OppositeMKT_Qty_RQ_0</to_name>
13 <dir>FIFO-based Price algorithms on Continuous phase\LimitOrder</dir>
14 <description>FIFO-based Price algorithms on Continuous phase</description>
15 </header>
16 <body>
17 <!-- 35 - Empty Book -->
18 <message>
19 <header>
20 <conn>OEG</conn>
21 <seq>5</seq>
22 <firm>
23 <type>NAME</type>
24 <value>0</value>
25 </firm>
26 <waitingTime>0</waitingTime>
27 </header>
28 <body>
29 <!-- 39 - Empty Book -->
30 <messageLength>
31 <option>$LENGTH_MESSAGE</option>
32 <value>0</value>
33 </messageLength>
34 <blockLength>
35 <option>$LENGTH_MANDATORY</option>
36 <value>0</value>
37 </blockLength>
38 <templateId>
39 <option>MINUS_OPTION</option>
40 <value>25876</value>
41 </templateId>
42 <schemaId>
43 <option>MINUS_OPTION</option>
44 <value>0</value>
45
```



NO	TEST	RESULT	STATUS
1	Test 1	OK	Pass
2	Test 2	OK	Pass
3	Test 3	OK	Pass
4	Test 4	OK	Pass
5	Test 5	OK	Pass
6	Test 6	OK	Pass
7	Test 7	OK	Pass
8	Test 8	OK	Pass
9	Test 9	OK	Pass
10	Test 10	OK	Pass
11	Test 11	OK	Pass
12	Test 12	OK	Pass
13	Test 13	OK	Pass
14	Test 14	OK	Pass
15	Test 15	OK	Pass
16	Test 16	OK	Pass
17	Test 17	OK	Pass
18	Test 18	OK	Pass
19	Test 19	OK	Pass
20	Test 20	OK	Pass
21	Test 21	OK	Pass
22	Test 22	OK	Pass
23	Test 23	OK	Pass
24	Test 24	OK	Pass
25	Test 25	OK	Pass
26	Test 26	OK	Pass
27	Test 27	OK	Pass
28	Test 28	OK	Pass
29	Test 29	OK	Pass
30	Test 30	OK	Pass
31	Test 31	OK	Pass
32	Test 32	OK	Pass
33	Test 33	OK	Pass
34	Test 34	OK	Pass
35	Test 35	OK	Pass
36	Test 36	OK	Pass
37	Test 37	OK	Pass
38	Test 38	OK	Pass
39	Test 39	OK	Pass
40	Test 40	OK	Pass
41	Test 41	OK	Pass
42	Test 42	OK	Pass
43	Test 43	OK	Pass
44	Test 44	OK	Pass
45	Test 45	OK	Pass



# MATCHING ENGINE

Exécution dans Minos avec comparaison entre les résultats obtenus et les résultats attendus générés par l'outil MBT

The screenshot displays the Euronext Minos Web interface. At the top, there are navigation buttons for 'CONFIGURATION' and 'INJECTOR', along with login fields for 'User' and 'Password', and a 'LOG IN' button. Below the navigation bar, there are tabs for 'TEST CASE', 'MESSAGES', 'EXPECTED RESULT', and 'RESULTS'. The 'RESULTS' tab is active, showing a table of test results. The table has columns for 'Recv Order', 'Send Seq', 'Tag', 'Expected value', 'Cmp', and 'Test value'. The results are grouped into two sections, each starting with a 'Recv Order' of 1. The first section shows a 'PASSED' status for the first two rows. The second section shows a 'PASSED' status for the first two rows, followed by a 'Logical Connection ID (149)' and a 'Last Message Sequence Number (741)'. The table also includes a 'HIDE TREE' section on the left with 'Squash' and 'MaTeLi' buttons, and a 'Last Execution' summary at the top of the results area.

Recv Order	Send Seq	Tag	Expected value	Cmp	Test value
1	1	storeID (2)	0	ANY	0
1	1	storeID (2)	0	ANY	0
1	1	storeID (2)	25077	=	25077
1	1	storeID (2)	0	ANY	0
1	1	storeID (2)	0	ANY	0
1	1	Logical Connection ID (149)	2	=	2
1	1	Last Message Sequence Number (741)	0	=	0
2	2	storeID (2)	0	ANY	129
2	2	storeID (2)	0	ANY	110
2	2	storeID (2)	25053	=	25053
2	2	storeID (2)	0	ANY	0
2	2	storeID (2)	0	ANY	0

Modélisation

Génération

Export Squash

Génération script

Export et exécution  
Banc de Test

# CONCLUSION - SITUATION COURANTE

- 1 Projet concerné – refonte du SI – ~100 FTE
- 20 utilisateurs Test Analystes
- 4 Modèles majeurs en cours
- ~15000 TC Générés dont 95% automatisés
  - Commandes (Script Shell)
  - Python to Squish primitives – GUI
  - Export XML pour injecteur messages
- Partage des modèles pour ‘validation’
  - Avec les Business Analyst
  - Intégration des TC dans le process de Continuous intégration

## Next Steps

- Développeurs intéressés – Test Unitaires ...
- Autres projets d'évolutions du SI ‘legacy’ ...





# CONCLUSION - ELEMENTS DE REUSSITES

- **Argumentaires Gestion du changement**
  - R&D, innovation
  - Intégration dans un écosystem QA Tools riche et varié

=> Value to QA
- **Processus de décision sur l'utilisation de l'outil MBT**
  - Partage du PoC avec équipes Devs et PM
  - Validation tour de table Steering Comittee
- **New Skills**
  - IT ! Python, scripts, ....
- **Gestion du risque**
  - Training et accompagnement sur site
  - Réfèrent interne ultra motivé !
  - Suivi hebdo



# CONCLUSION – POINTS A AMÉLIORER

- **Fusion des Modèles**
  - **Difficultés pour fusionner les modèles a plus de 3 utilisateurs**
  - **Amélioration proposée: utilisation d'une data base**
- **Prise en main de l'Outil**
  - **Prévoir entre 1 et 2 mois d'utilisation pour une bonne prise en main**
  - **Bien choisir ces ressources**



# ROI ATTENDU

Le ROI a été défini en prenant en compte l'implémentation d'Optiq, puis la réutilisation pour les nouveaux projets et BAU, Crs.

Optiq QA Estimations : ~6000 md	Pratiques avant Matelo	Optiq avec MaTeLo – 18 mois	Post Optiq – BAU & CRs
Analyse des Spec	10 %	10 %	10 %
Création des Tests	30 %		
Modélisation avec Matelo		30 %	10 % updates
Exécution des tests	30 %	30 %	20 %
Automatisation des Tests	20 %	10 %	10 %
Régression en Continue	10 %	10 %	10 %
<b>Total</b>	<b>100 %</b>	<b>90 %</b>	<b>60 %</b>
<b>ROI attendu</b>	<b>Rien pour les prochains 18 mois – implémentation d'Optiq</b>		<b>30 % to 40 % en 2018</b>

