

Testing Tools inside Eclipse

Michael G Norman, PhD
CEO Scapa Technologies

TPTP Testing Tools Project Lead

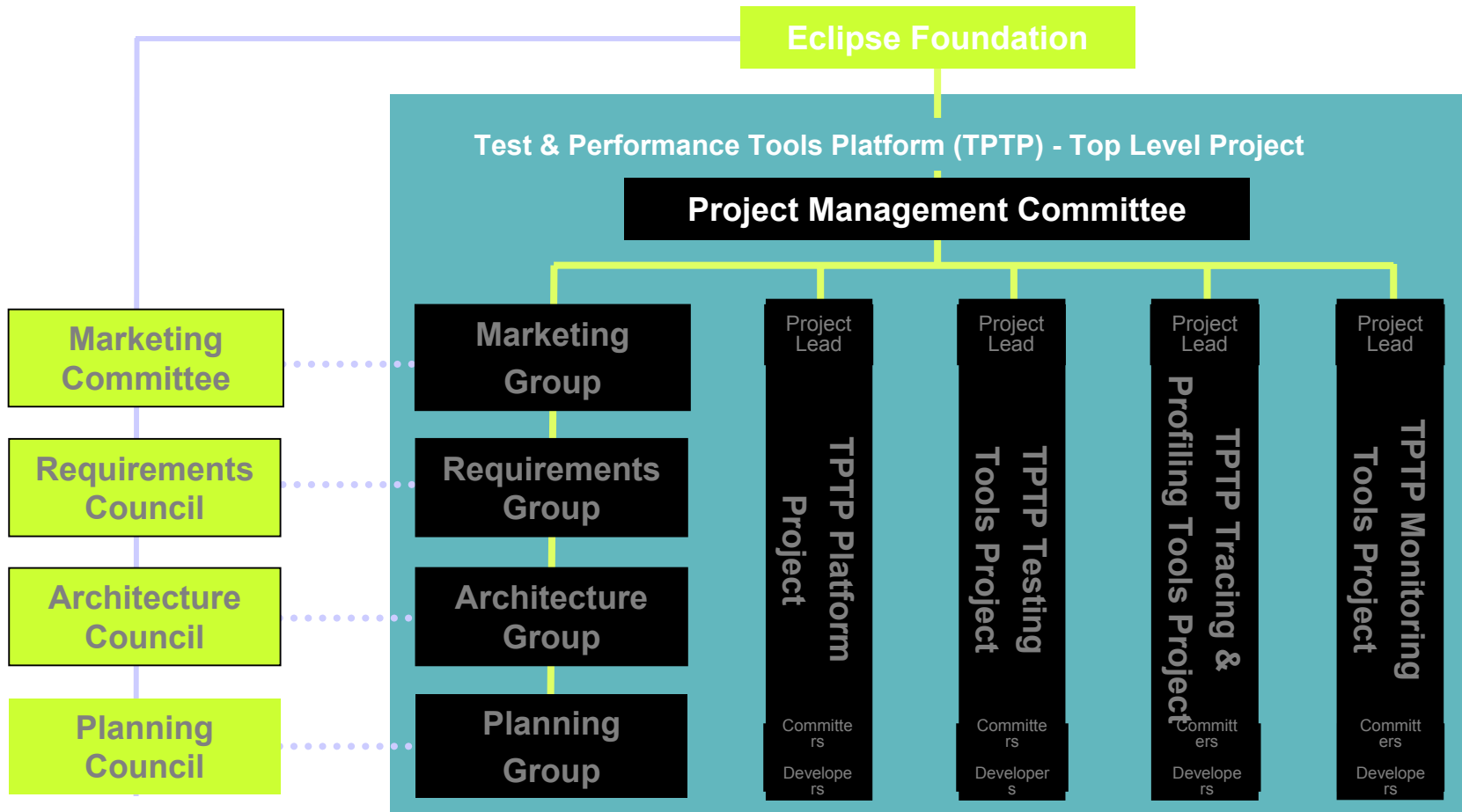
TPTP – Formerly Hyades

- Launched 2002, 1.0 GA alongside Eclipse 2.1, 3.0/3.0, 4.0/3.1
- Multi-vendor contribution/adoption strategy
 - IBM*, Scapa*, Intel*, CA*, SAP†, FOKUS, OC Systems, Compuware
 - Layered and flexible integration/adoption model
- Test, trace, profile, monitor
- Cross-lifecycle
 - May operate in source-code free environments
- Initial motivation was consumption by other tools
 - Sample tools are relatively low function
 - Some overall useability issues
 - A project “theme” to improve exemplary tools

* Strategic Developer

† Strategic Consumer

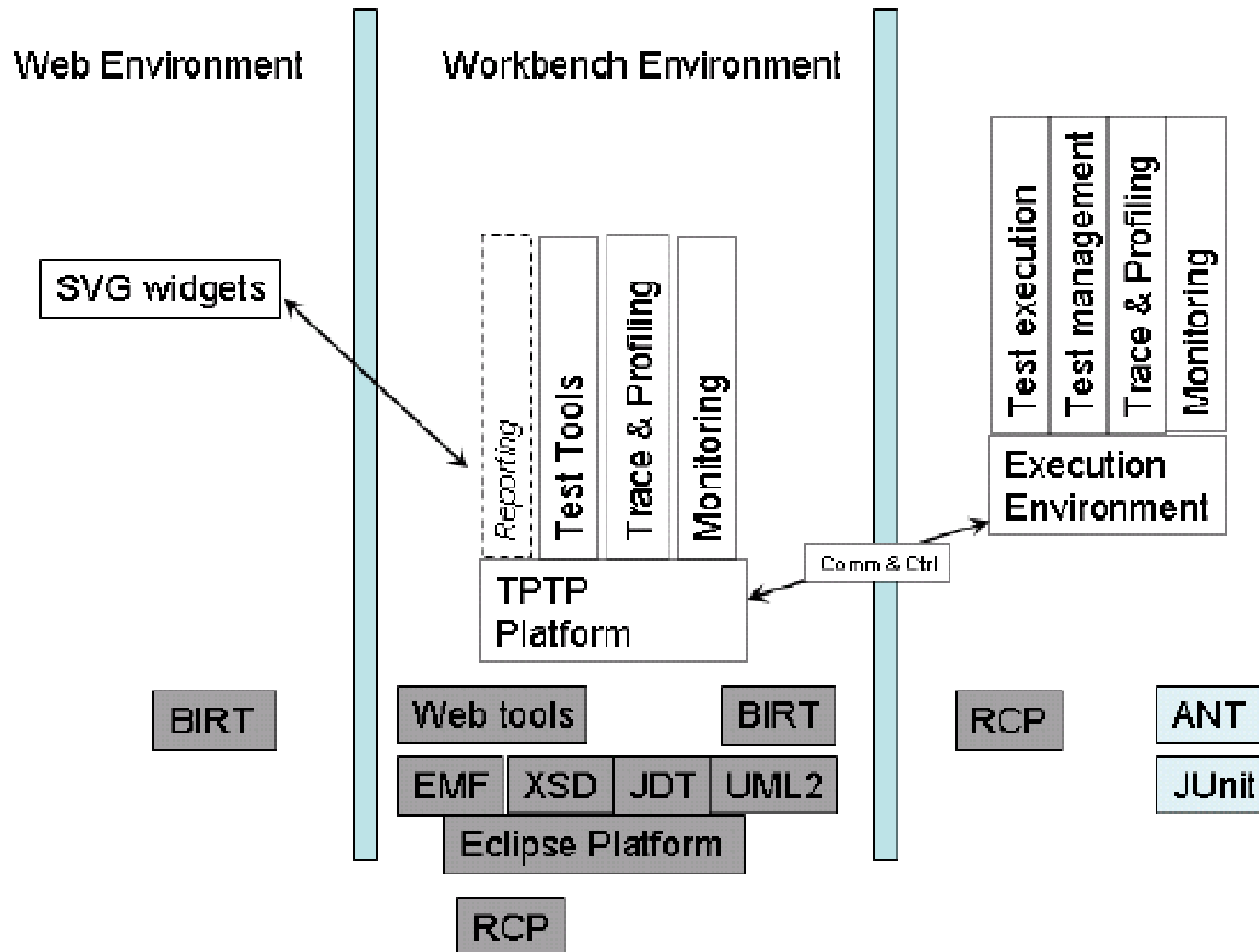
TPTP Project Structure



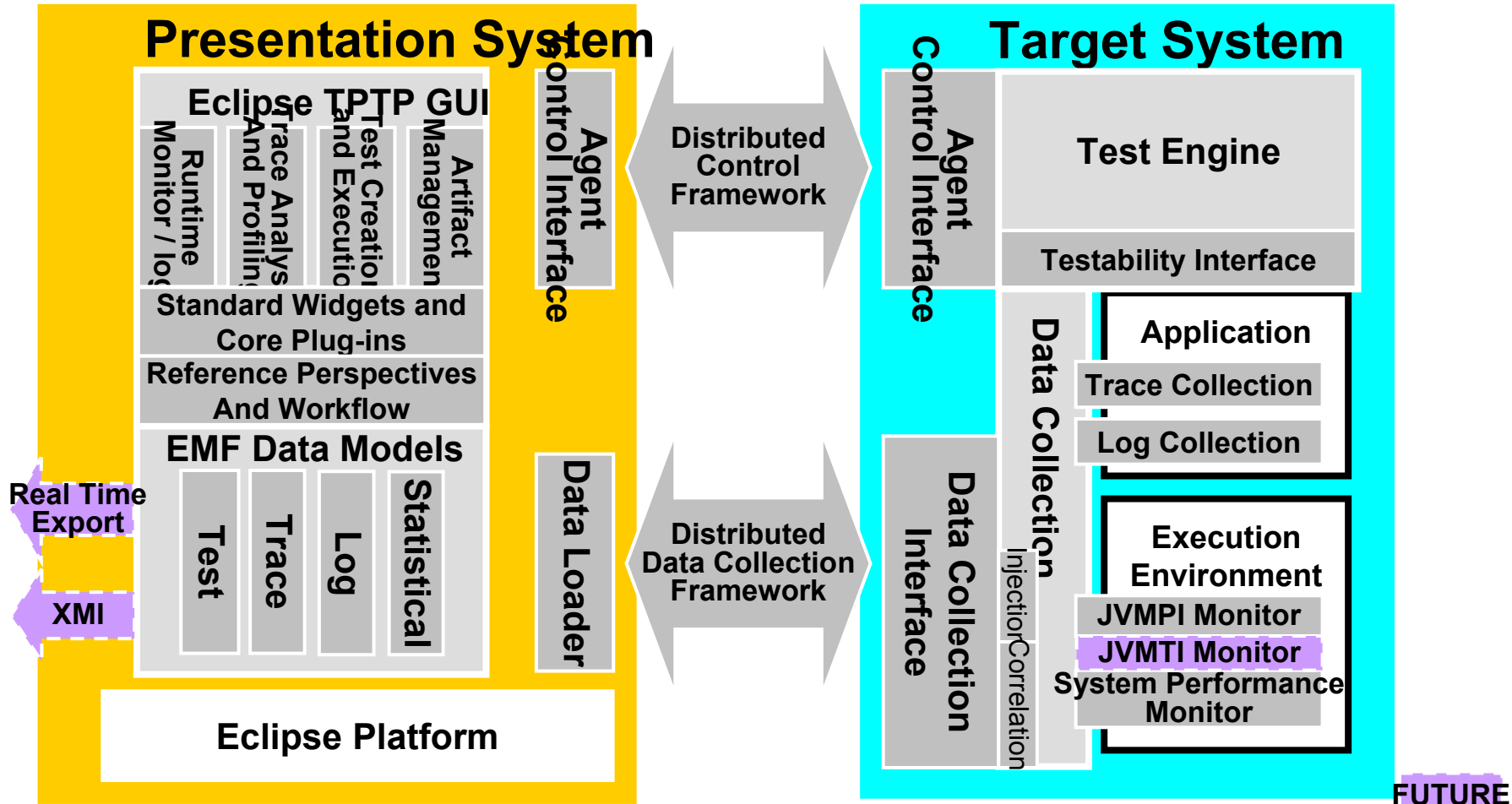
Some issues

- Genuine diversity of experience & interest
 - Different Development Processes
 - Different objectives
 - Collaborators sometimes compete in the marketplace
- Concerns about auto-commoditization
 - Vendors need to understand how TPTP contributions impact their own markets – “value point” migration & timing thereof
 - We have to pay our staff
- Layered adoption model
 - Product re-engineering costs money & can often be very difficult
 - Migration is disruptive to installed base
 - Market pressures drive towards interoperability & “ever closer union”

The Official TPTP Architecture Diagram



The Unofficial Hyades Architecture Diagram



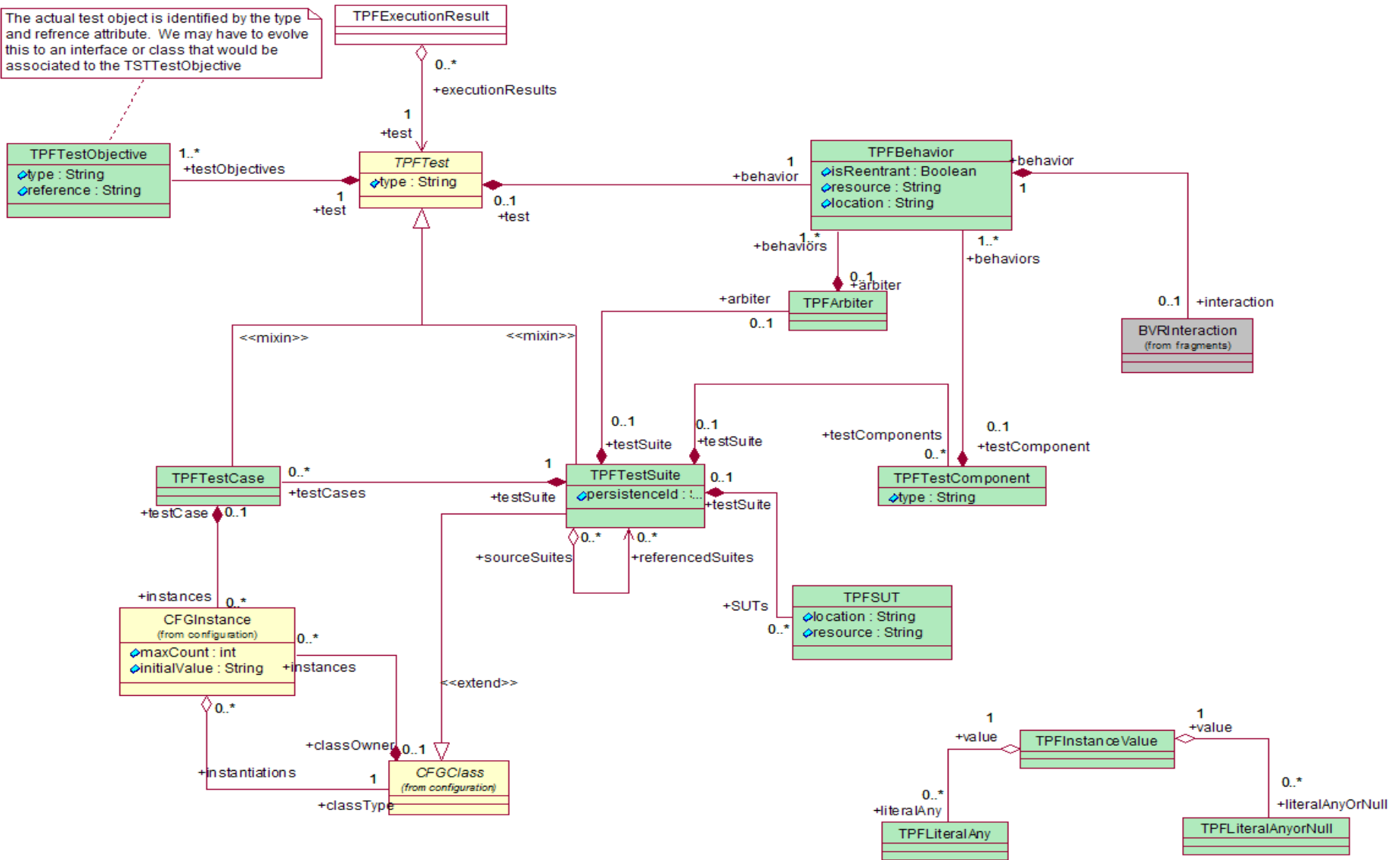
What's an EMF data Model

- The problem
 - how do I store/retrieve my data?
 - How do I share data with other tools?
- EMF
 - A model is a UML/MOF description of the “things” the tool deals with
 - Auto-generated API within the workbench to persist/retrieve
 - Bound into the Eclipse workspace model
 - Serialization/deserialization etc handled “for free” to XMI
 - Tools interoperate if they use the APIs and respect the semantics in the model

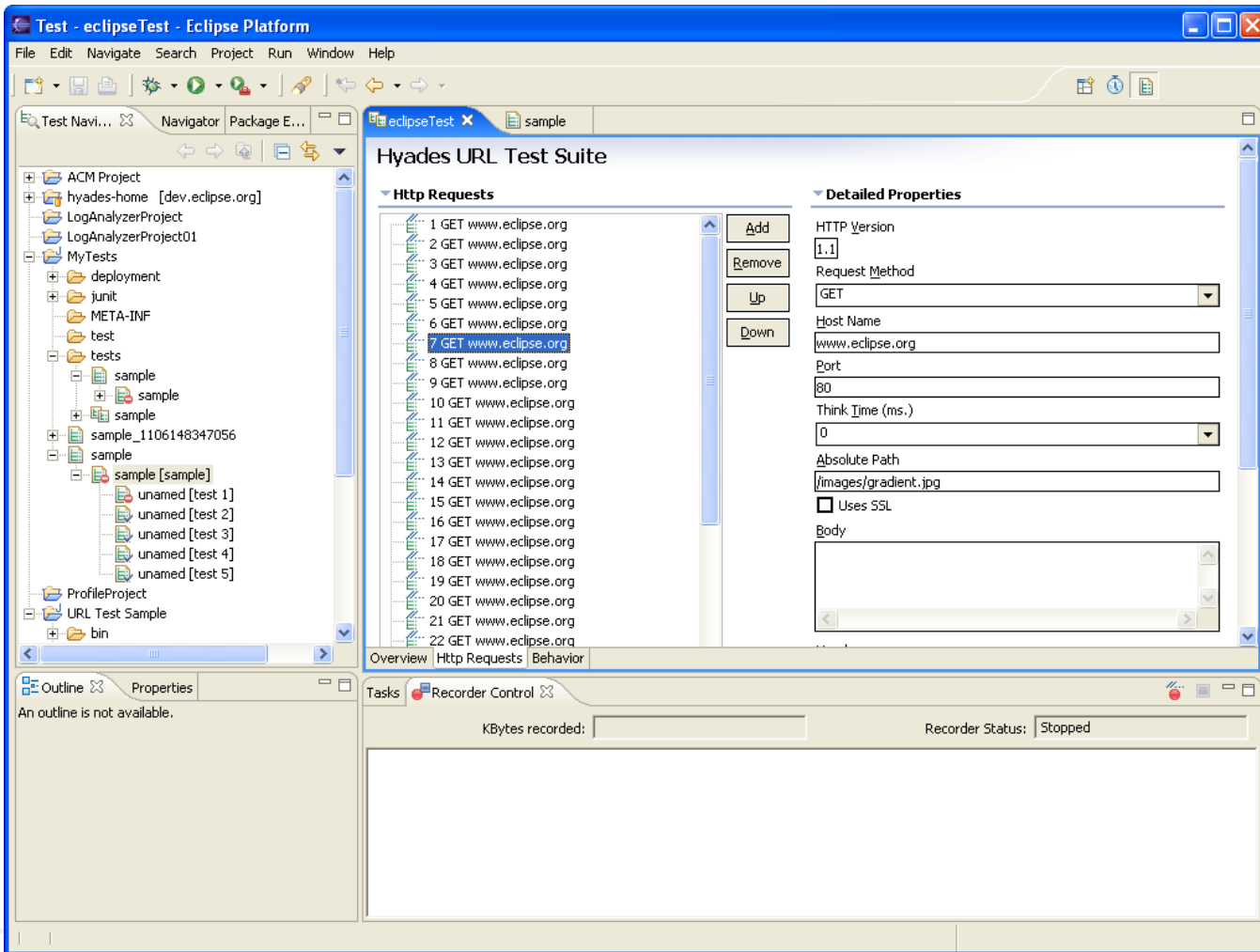
The 4 models

- **Test data model** supports test cases, input data, results and execution history. It consists of three models
 - A data model for creation, definition and management of test artifacts. This implements UML 2 Testing Profile meta model
 - A data model for test case behaviors. It implements UML 2 Interaction meta model.
 - A data model for test execution history. It supports execution traces and results from disparate test types.
- **Trace data model** supports traces of local and distributed execution stacks and heaps
- **Log data model** supports sequence of CBEs and other logged messages that are transformable into CBE.
- **Statistical data model** supports snapshots of arbitrary data over time

The actual test object is identified by the type and reference attribute. We may have to evolve this to an interface or class that would be associated to the TSTTestObjective



User interface isn't (usually) UML



Data Models (Test Model)

Behavioral model

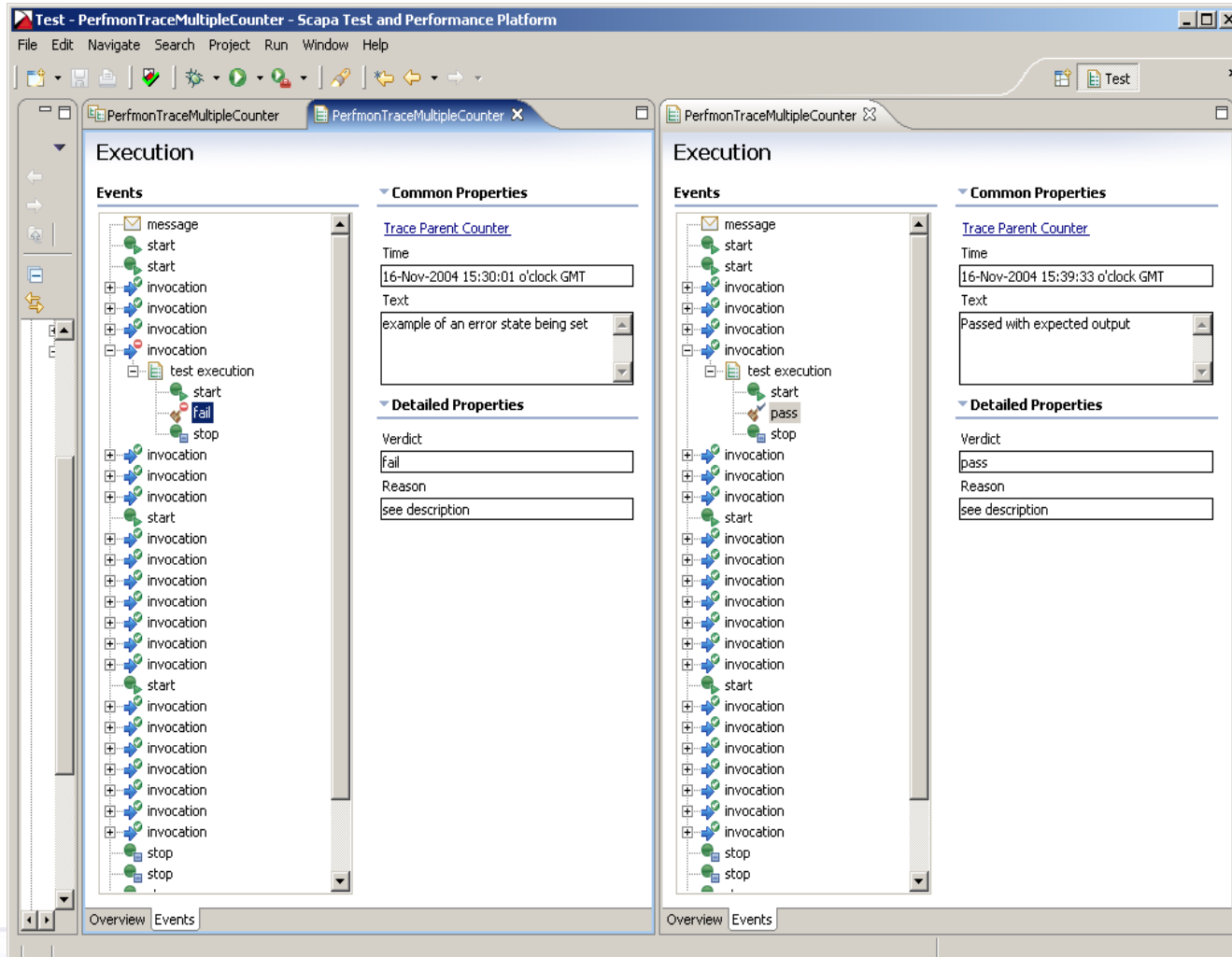
- The behavioral portion of the Test Model is an implementation of UML2 Interactions.
- Behavioral model is optional
 - An extension point to glue in pre-existing tools engines as external behaviours
 - Possible to mix e.g. wrap External test tools inside TPTP behaviour
- Interface is via a façade
 - To provide a more semantically familiar programming model (loops and decisions, not combined interaction fragments with operators.)
 - To facilitate interop among tests from different test tool vendors (multiple ways of representing the same transaction)
 - To (potentially) allow TPTP based products to access behavior that is not modeled (via another implementation of the same façade interface.)

Data Models (Test Model)

Execution Histories

- Stores the results of test executions
 - References to test, trace, deployment info
 - Verdicts
 - Test Log messages and console out from test
 - Provides an extensibility mechanism for custom typed attributes

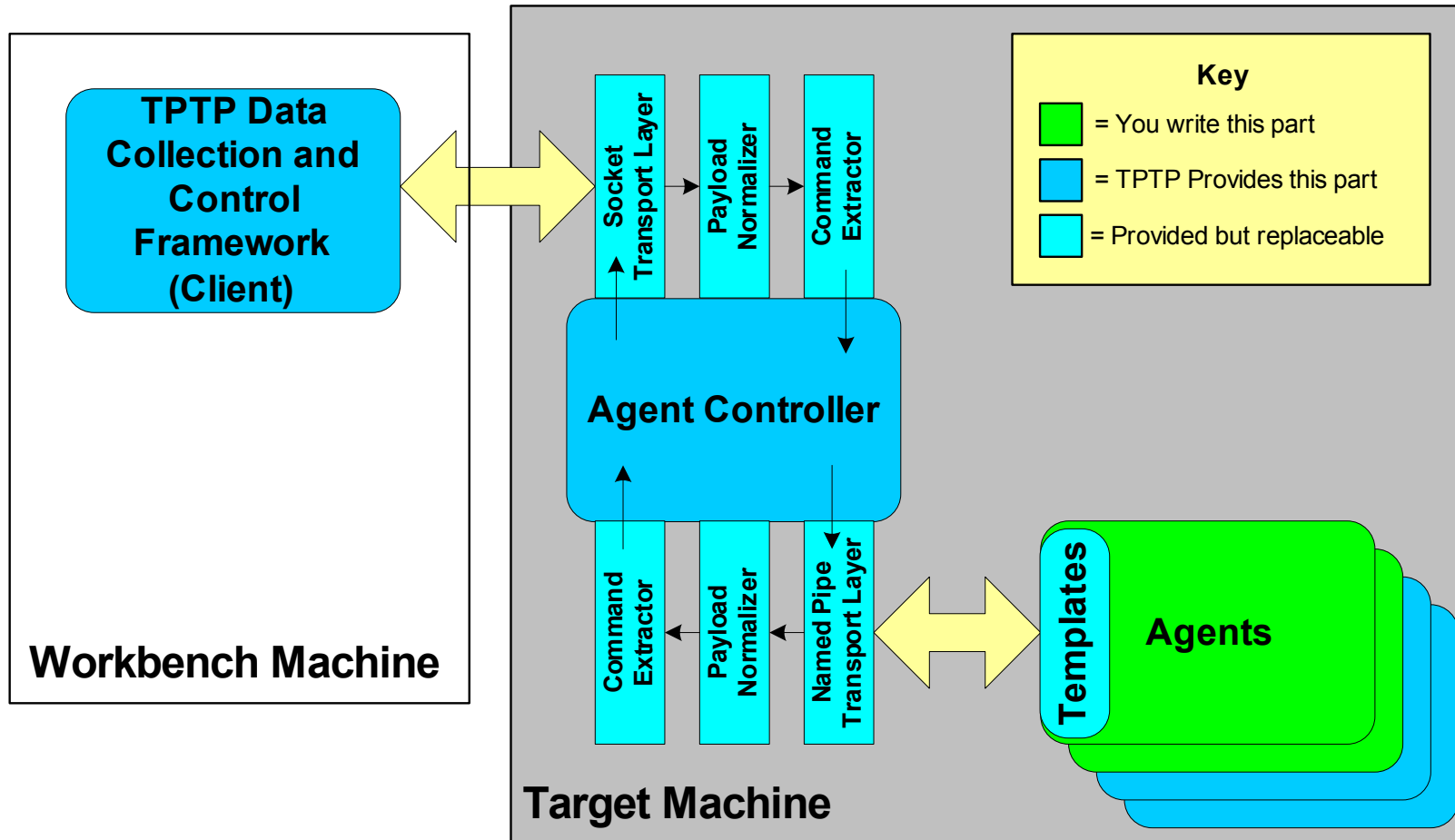
Test Execution Histories



Execution Capabilities

- TPTP Provides
 - extension points for external execution
 - It's own extensible runtime/code generation for sample test tools based around façade & native behaviours (more later)
 - Combinations of native and external behaviours (e.g. junit)
 - A generic non-test-specific external behaviour is also emerging through leveraging BPEL/WSDL/XSD/XPATH
 - Finally provides us with a testability interface – WSDL
 - Also allows all TPTP agents to be controlled via WSDL
 - Will leverage WTP

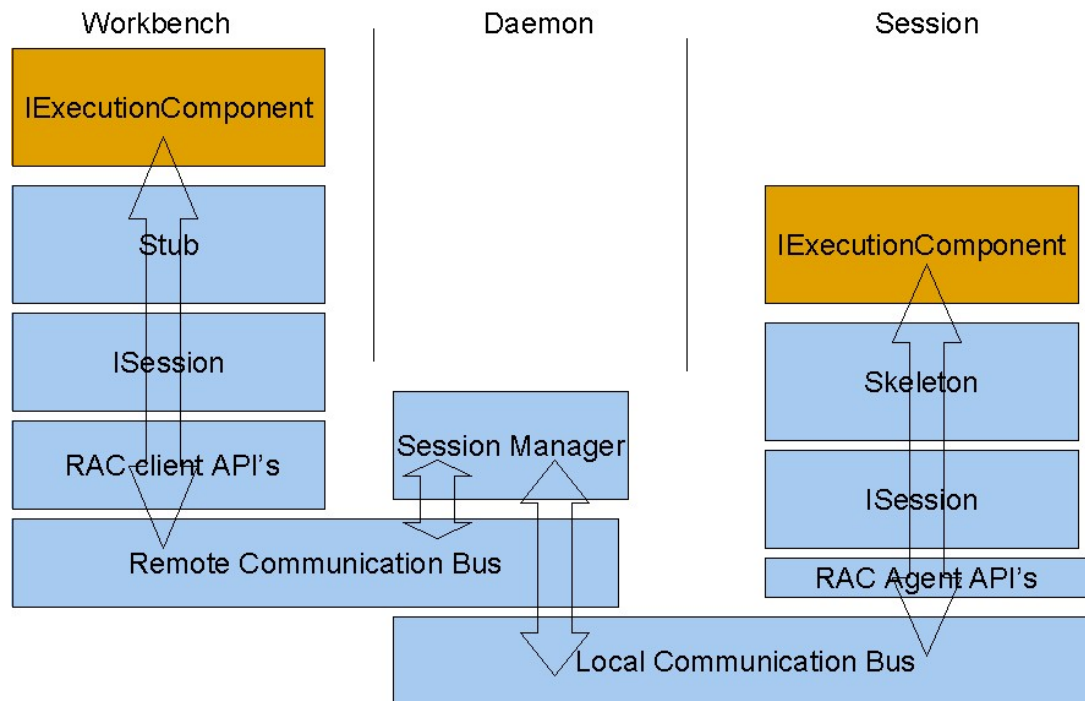
TPTP Agent Controller



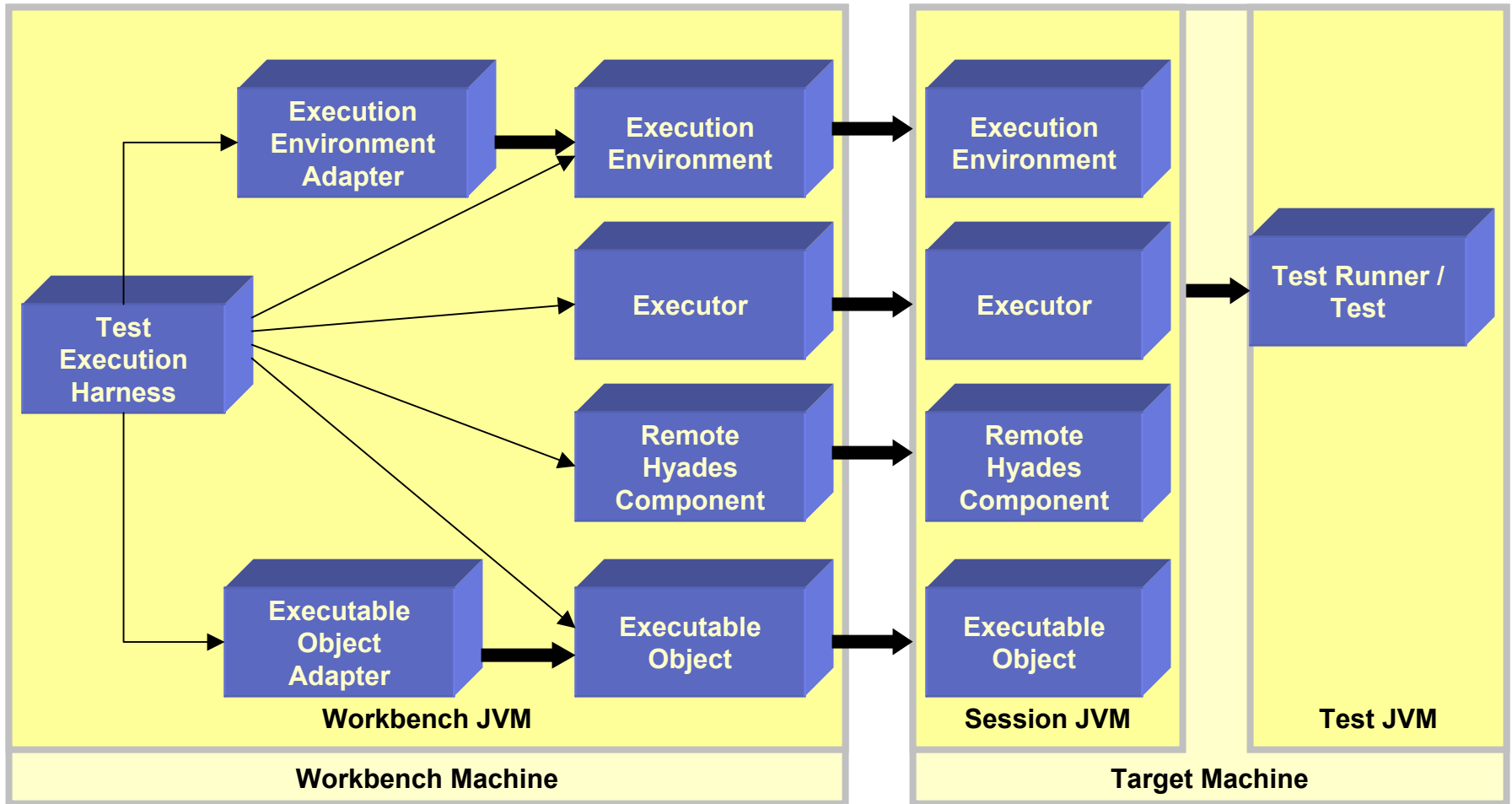
Note: Workbench machine and target machine can be the same machine.

Test Execution explained

- The Execution Components are remoted via the TPTP Platform Execution Environment
- The Test Execution Harness invokes methods of the Execution Components locally in the Workbench process, and the agent controller marshals and invokes the methods on the target execution environment.



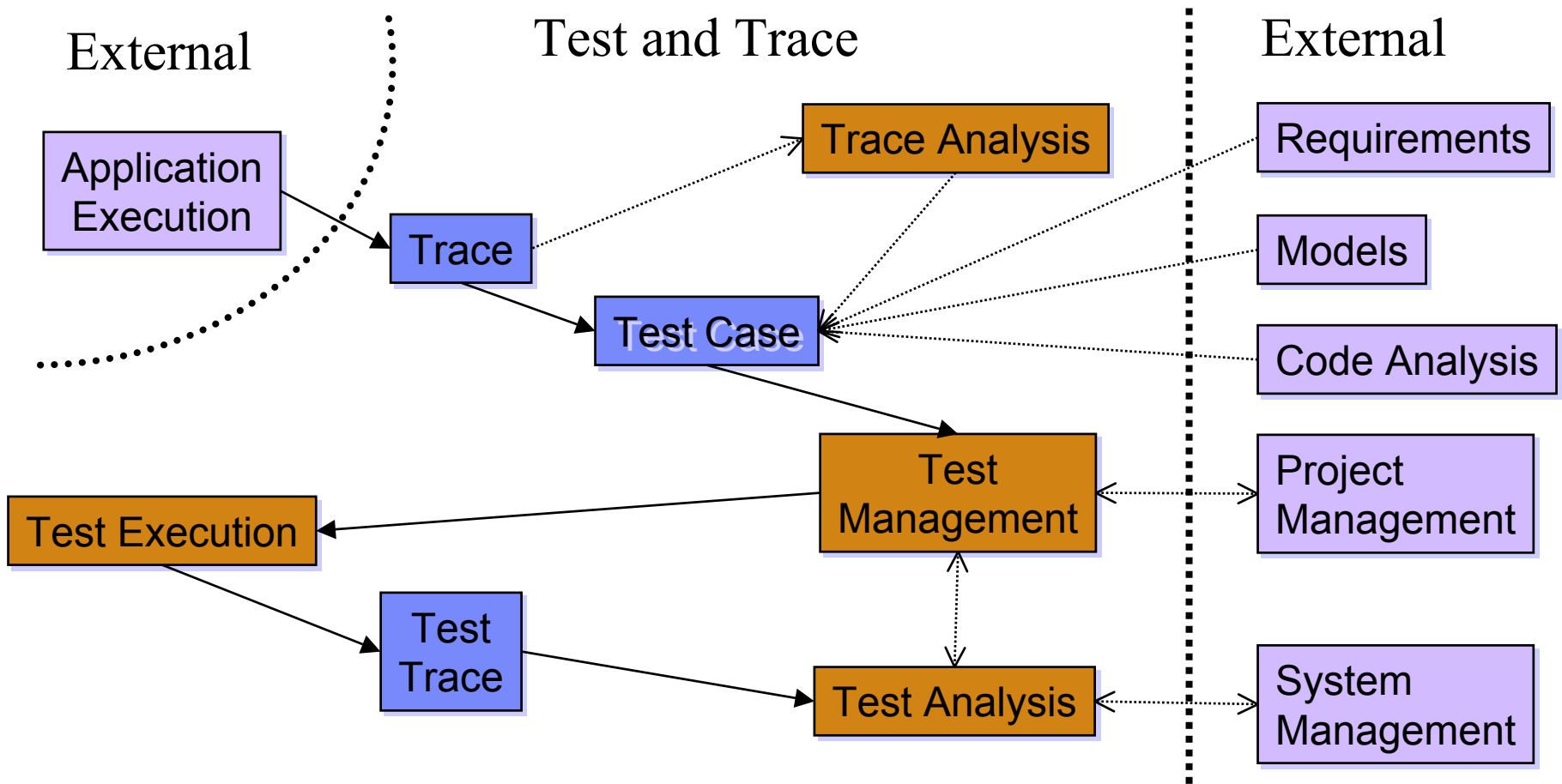
Test Execution explained



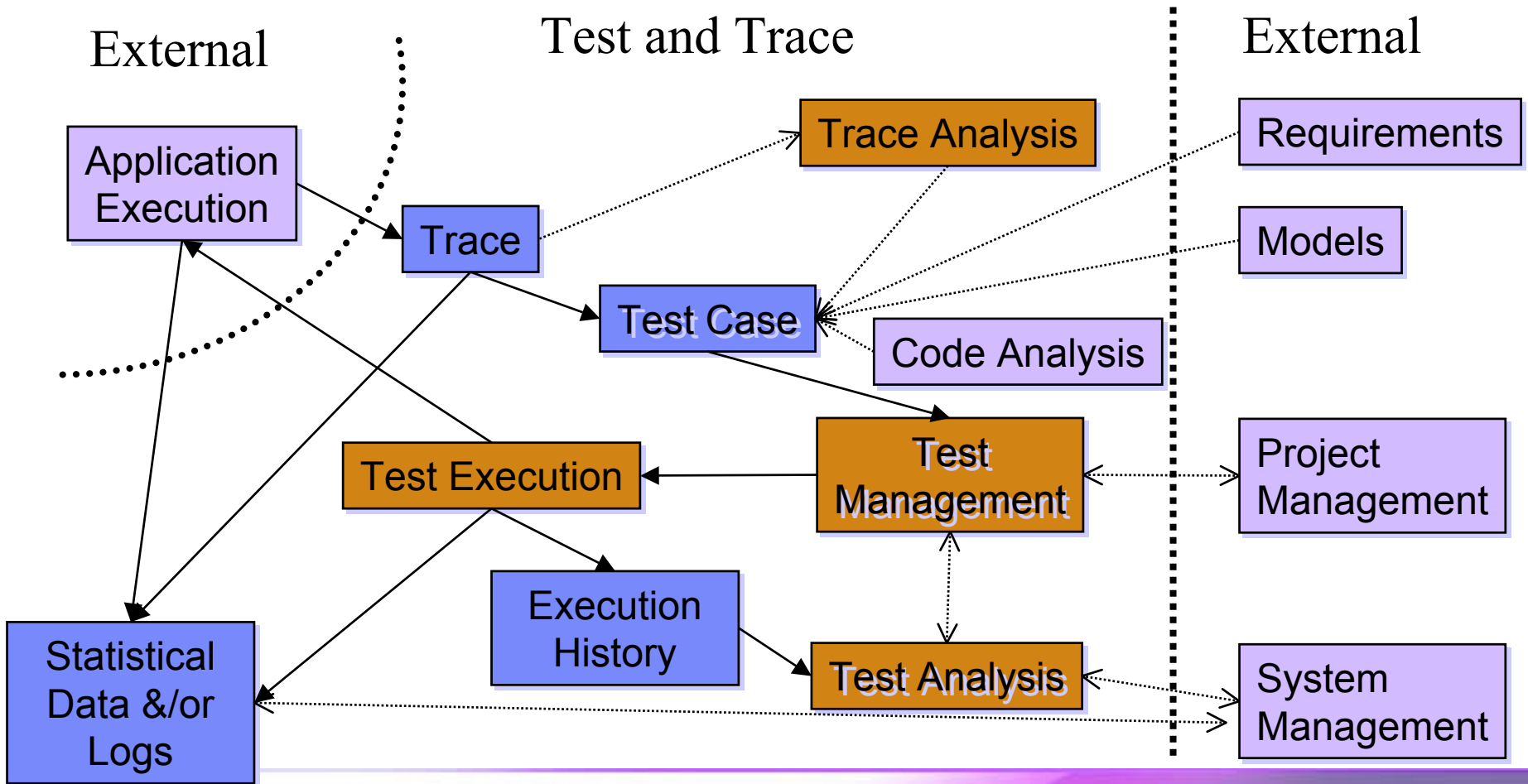
Sample Tools

- URL Test - Performance Testing of Web Applications
- Junit (within TPTP framework)
- Manual test (assisted)
 - Test Creation / Recording (Web)
 - Editing
 - Generating Java code
 - Deploying & Running test
 - Analyzing test results

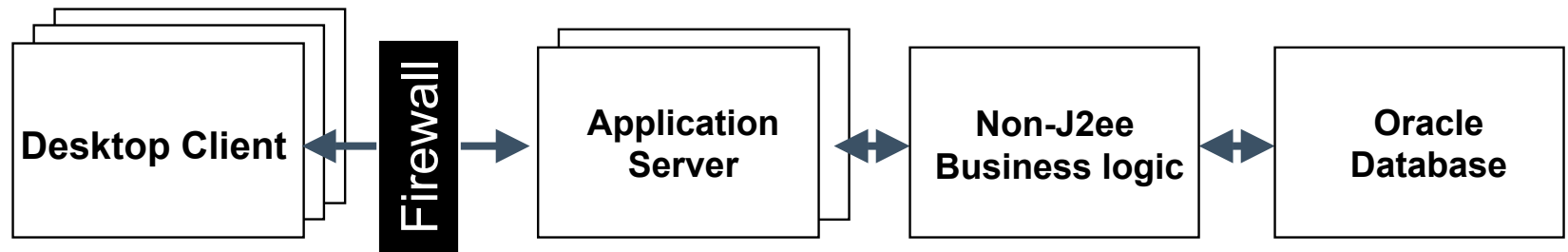
What about the other models? (Prehistoric slide)



What about the other models? (Updated slide)

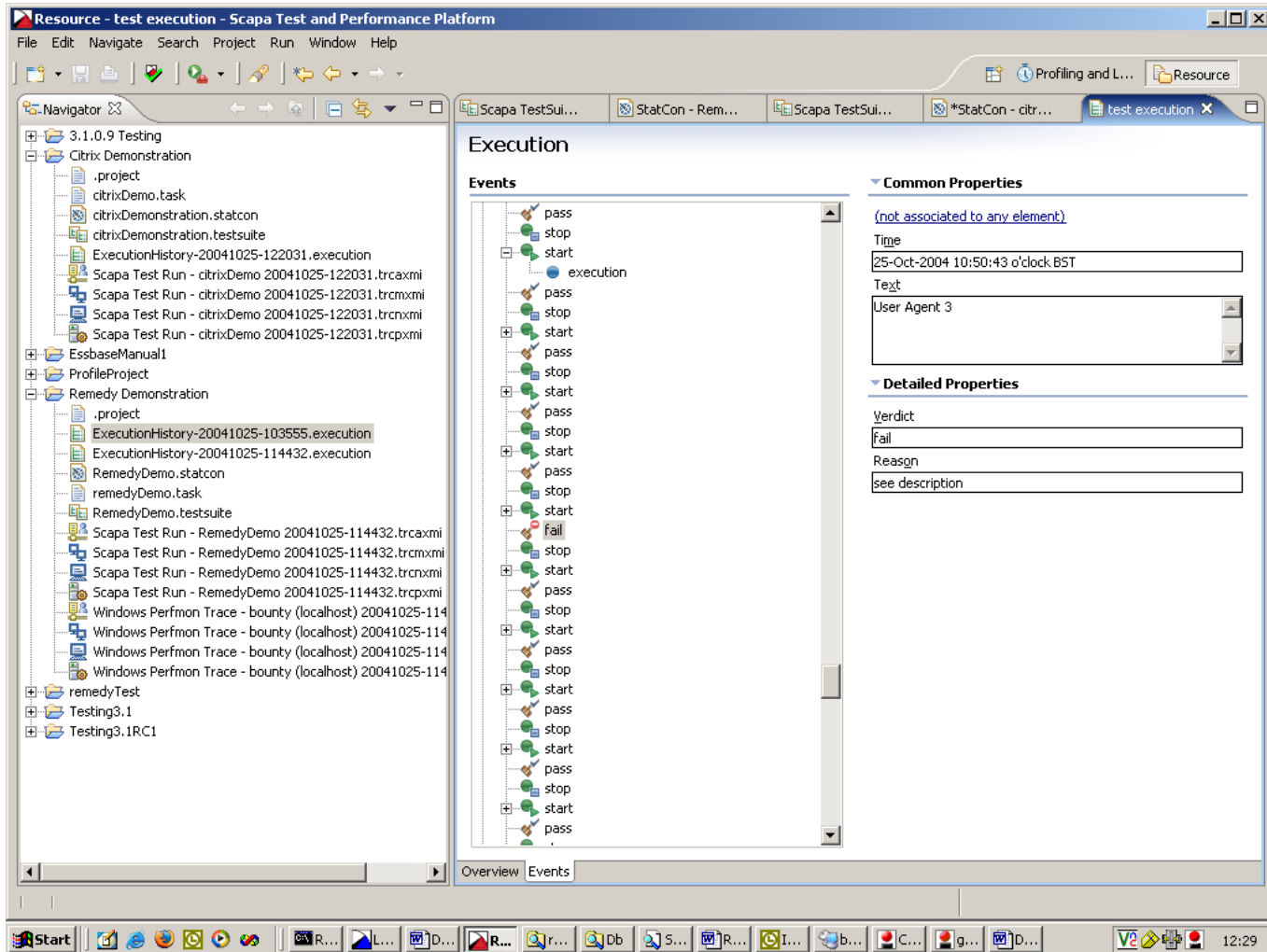


TPTP in action – It's all one thing!



- It's not *My* fault, honest
- Loosely Fictionalized Scenario –
 - real customer, finance sector, Europe
 - hosting company, ISV, Integrator
- Based on Scapa 3.1 but almost equivalent scenario could be performed with out-of-the-box TPTP for some system technologies

Test Execution History



Correlate Logs

The screenshot displays the Eclipse IDE interface for the 'Scapa Test and Performance Platform'. The main window is titled 'UML2 Sequence Diagram' and shows a sequence diagram titled 'Sequence Diagram: Log interactions <Scapa Time Correlation (Large Tolerance Level)>'. The diagram features two lifelines: 'Scapa Test Trace - cit...' and 'Remedy Error Log D:'. The 'Scapa Test Trace' lifeline has several activation bars, and the 'Remedy Error Log D:' lifeline has one. A message arrow points from the Scapa Test Trace lifeline to the Remedy Error Log D: lifeline, with the text: '390600 : This version of the Action Request System(R) is ready for use or evaluation...'. The left sidebar shows a 'Log Navigator' with a tree view of logs and correlations. The logs include entries from 'localhost : Scapa Test Trace - citrixDemo 2004' and 'bounty : Remedy Error Log D:\remedy\Arserver\'. The bottom of the screen shows the Windows taskbar with various application icons and the system clock at 18:26.

Find Logged Error Message

The screenshot shows the Eclipse IDE interface with the 'Profiling and Logging' window open. The window title is 'Profiling and Logging - StatCon - citrixDemonstration.statcon - Scapa Test and Performance Platform'. The 'Log Navigator' on the left shows a tree view of logs, with the selected log being 'localhost : Scapa Test Trace - ThinSmoke1 20041019'. The 'Log View' on the right displays a list of log records, with the selected record being '390600 : Failure during SQL operation to the database (ARERR 552)'. The 'Property' table on the right shows the details of the selected log record.

Property	Value
analyzed	false
creationTime	2004-10-25 16:55:43.000000-00:00
elapsedTime	0
extensionName	CBCommonBaseEvent
globalInstanceId	FDB4910F1E1828BA01C2BBF326AC11D9
localInstanceId	
msg	390600 : Failure during SQL operation to the datab
priority	0
repeatCount	0
sequenceNumber	68
severity	60
version	1.0.1

Futures

- Significant UI enhancements to sample tools
- BIRT-based reporting
- Trace->test conversion
- Some sample tooling based on SOA runtimes

- ANYONE IN THE WORLD CAN SUBMIT AN ENHANCEMENT REQUEST TO TPTP

- TPTP has a broad contribution base. We would love to see it getting even broader.