Test Automation at the useR interface level

Test*

Tanja E. J. Vos

Software Testing and Quality Group

Research center for Software Production Methods (PROS)

Universidad Politecnica de Valencia

Spain

SATToSE 2014, L'Aquila 2014

Contents

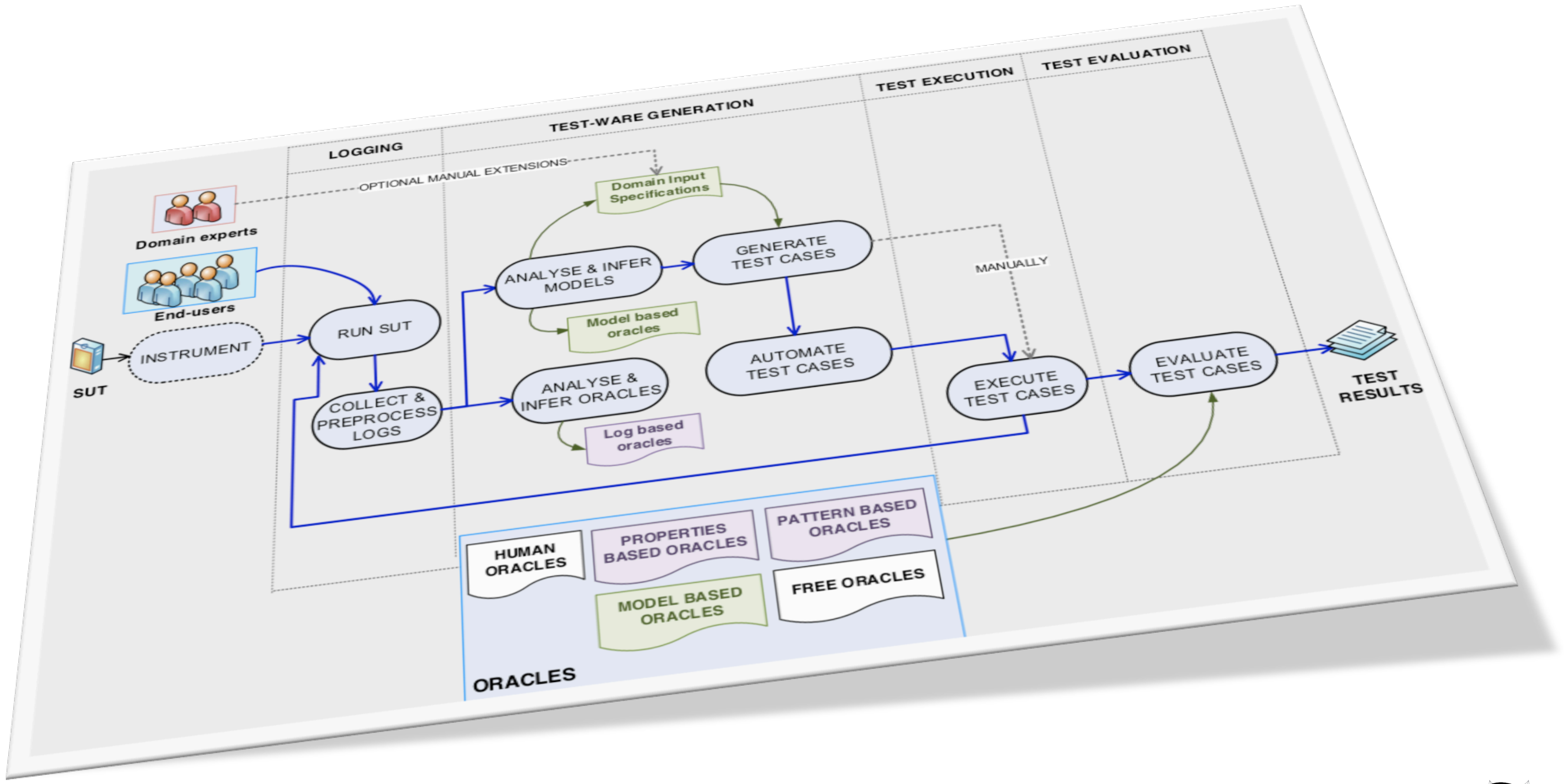
FITTEST

- Future Internet Testing
- September 2010 – February 2014
- Total costs: 5.845.000 euros
- Partners:
  - Universidad Politecnica de Valencia (Spain)
  - University College London (United Kingdom)
  - Berner & Mattner (Germany)
  - IBM (Israel)
  - Fondazione Bruno Kessler (Italy)
  - Universiteit Utrecht (The Netherlands)
  - Softteam (France)
- http://www.pros.upv.es/fittest/

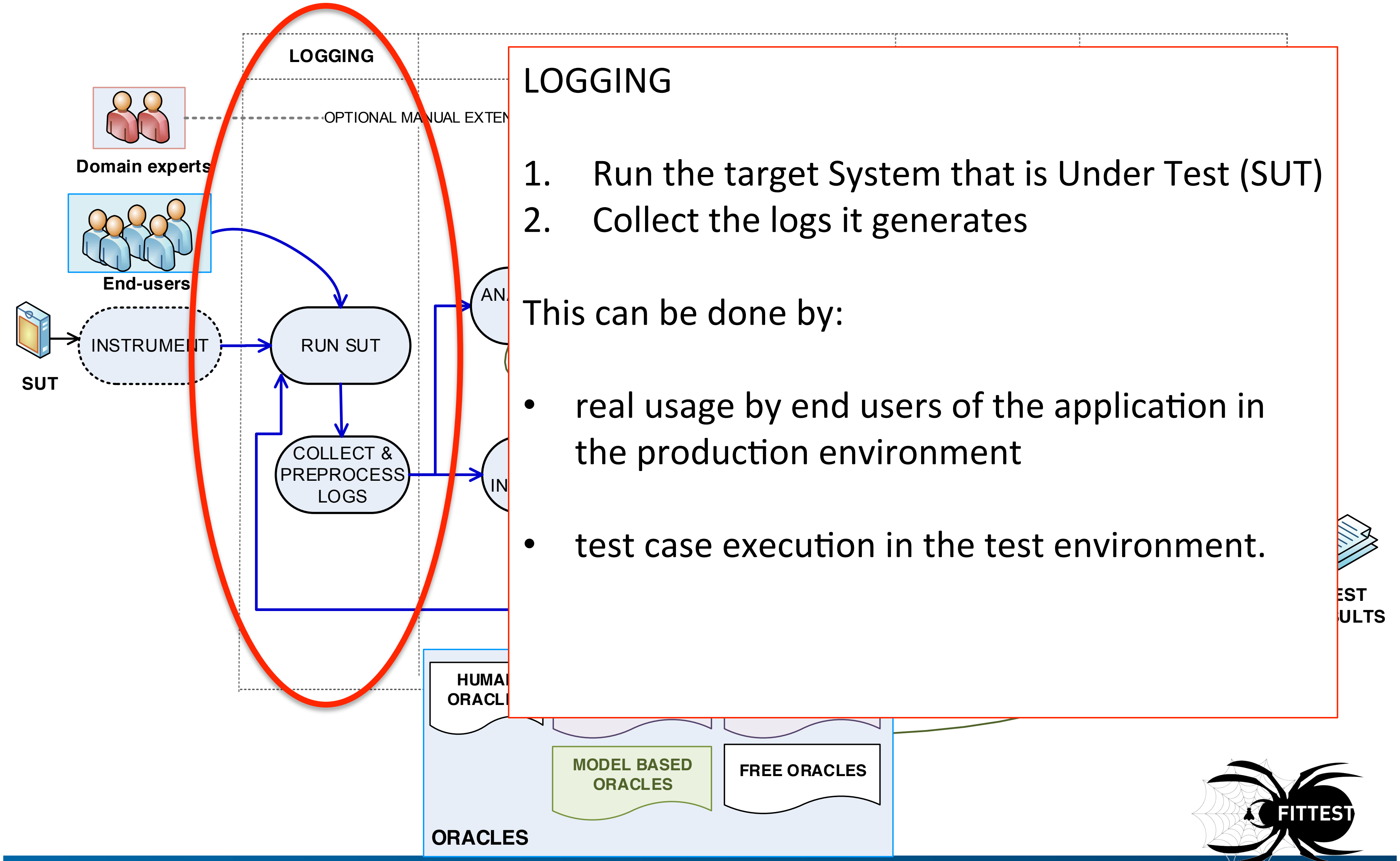
FITTEST objectives/results

- **Future Internet Applications**
  - Characterized by an extreme high level of dynamism
  - Adaptation to usage context (context awareness)
  - Dynamic discovery and composition of services
  - Limited observability (3rd party black-box components)
  - Etc..
- **Testing of these applications gets extremely important**
  - Society depends more and more on them
  - Critical activities such as social services, learning, finance, business.
- **Traditional testing is not enough**
  - Testwares are fixed
- **Continuous testing is needed**
  - Testwares that automatically adapt to the dynamic behavior of the Future Internet application
  - This is the objective of FITTEST

The FITTEST tools for Continuous Testing



FITTEST continuous testing system



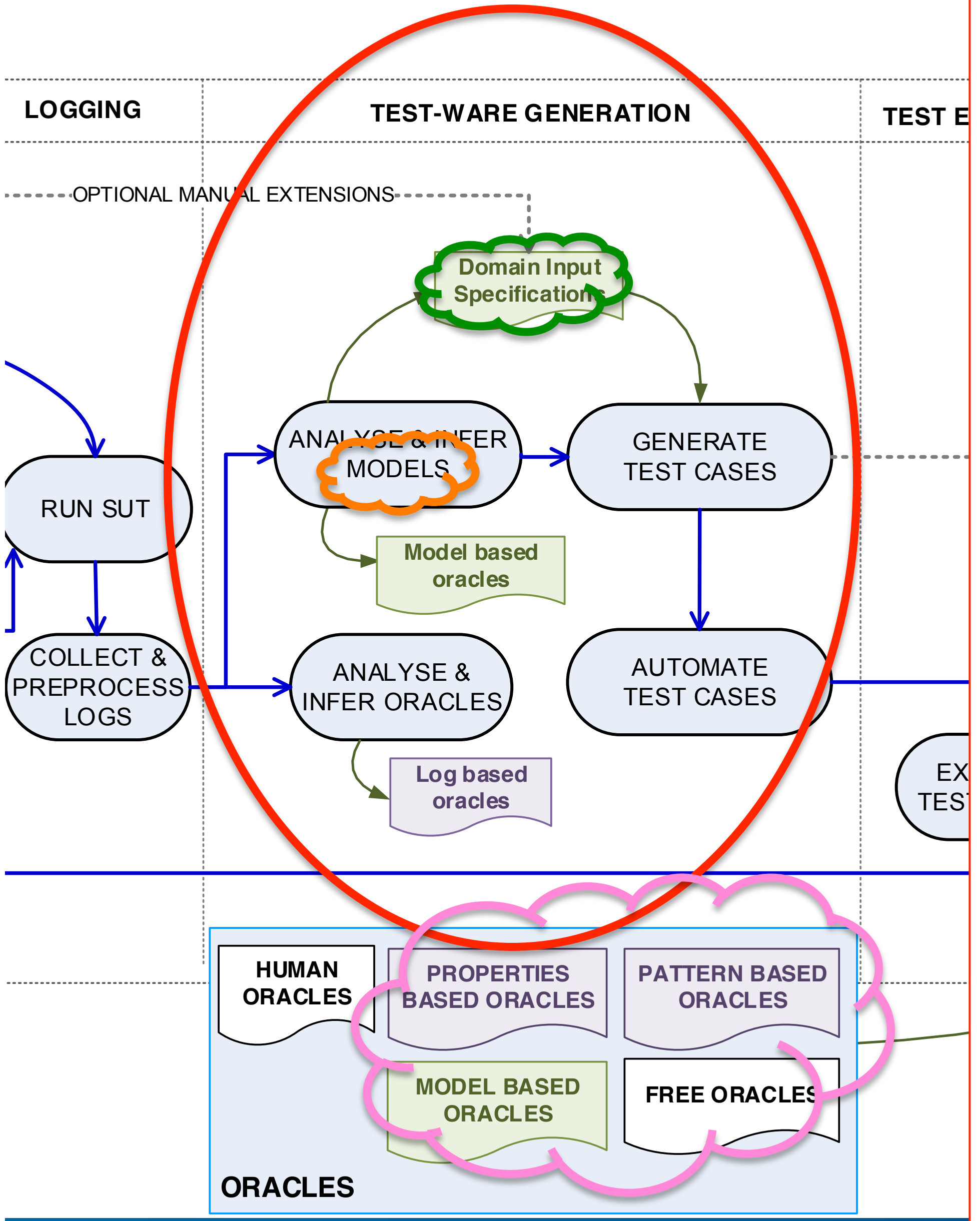
LOGGING

1. Run the target System that is Under Test (SUT)
2. Collect the logs it generates

This can be done by:

- real usage by end users of the application in the production environment
- test case execution in the test environment.

How does it work?



GENERATION

1. Analyse the logs

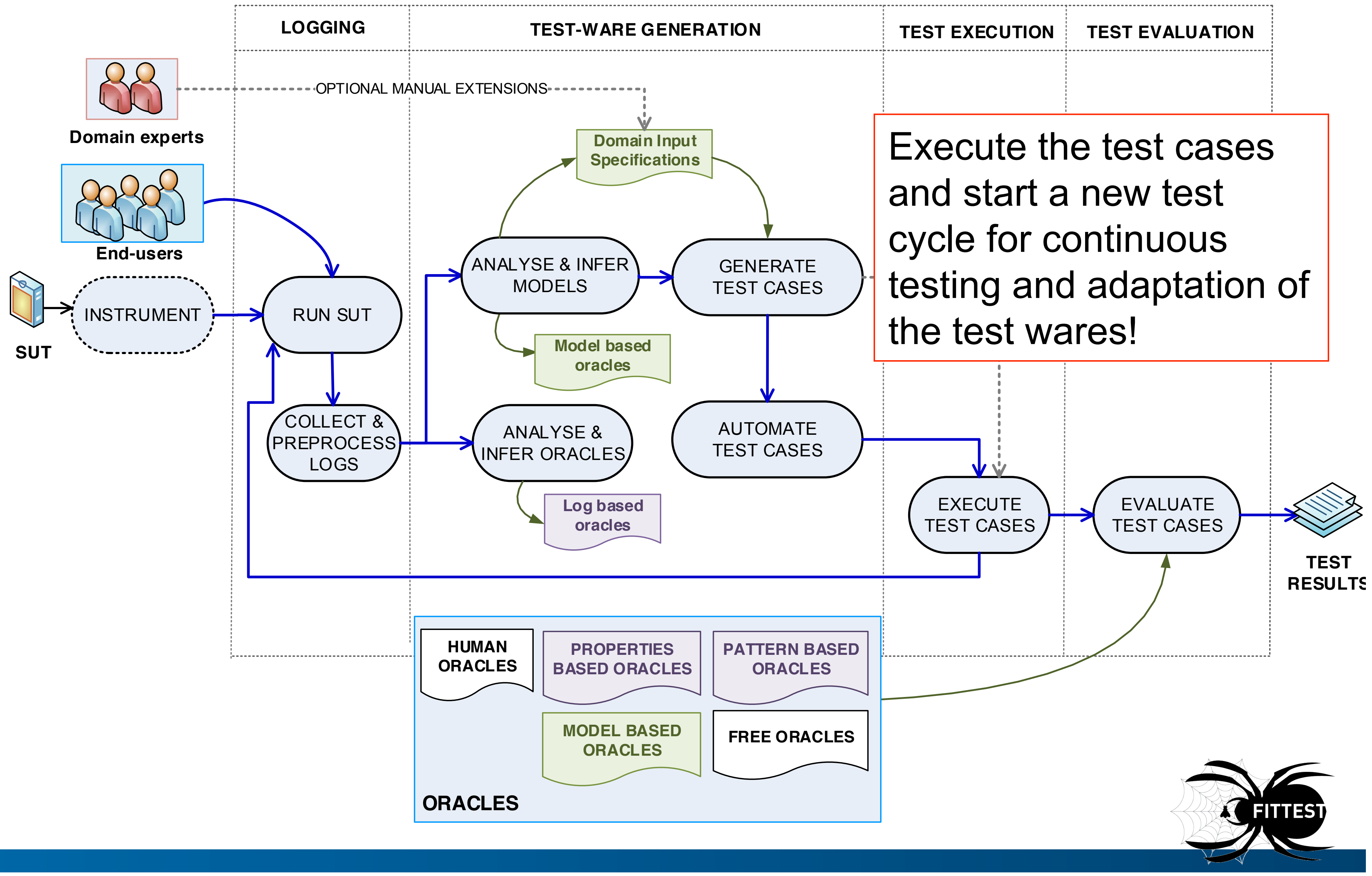
2. Generate different testwares:

   - Models
   - Domain Input Specification
   - Oracles

3. Use these to generate and automate a test suite consisting off:

   - Abstract test cases
   - Concrete test cases
   - Pass/Fail Evaluation criteria

How does it work?



Execute the test cases and start a new test cycle for continuous testing and adaptation of the test wares!

And it does work, but…….

- We cannot always get the logs…
- The logs do not always contain the info we need to derive a good model/oracle
- Instrumentation is not always an option (3rd party components)

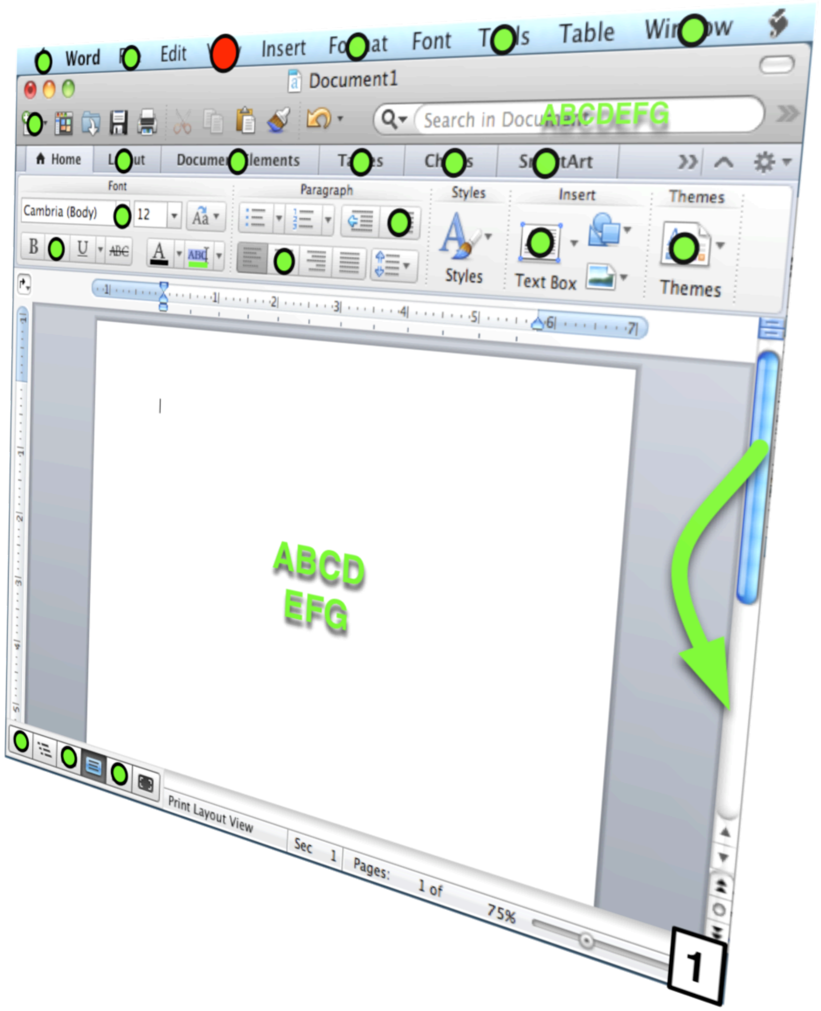
Do you want to know more…

- Vos, Tanja E.J., Lakhotia, Kiran, Bauersfeld, Sebastian (Eds.) **Future Internet Testing**, LNCS 8432, 2014

- Paolo Tonella youtube lecture: https://www.youtube.com/watch?v=TnuiEGS6iyc

- Cu D. Nguyen, Bilha Mendelson, Daniel Citron, Onn Shehory, Tanja E.J. Vos, and Nelly Condori-Fernandez. **Evaluating the fittest automated testing tools: An industrial case study**. In Proceedings ESEM 2013, pp 332–339.

If we cannot rely on the logs, why not rely on what we can see…. the UI



Testing at the UI Level

- UI is where all functionality comes together
  - Integration / System Testing

- Most applications have UIs
  - Computers, tables, smartphones….
- Faults that arise at UI level are important
  - These are what your client finds -> test from their perspective!
- No need for source code
  - But if we have it even better ;-)

State of the art in UI testing

- **Capture Replay**
  - The tool captures user interaction with the UI and records a script that can be automatically replayed during regression testing
  - UI change (at development time & at run time)
  - Automated regression tests break
  - Huge maintenance problem
- Visual Testing
- Model-based Testing

State of the art in UI testing

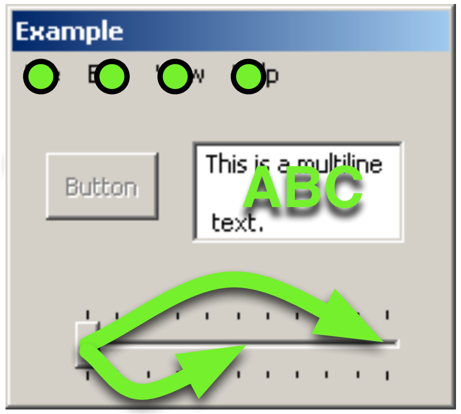
- Capture Replay

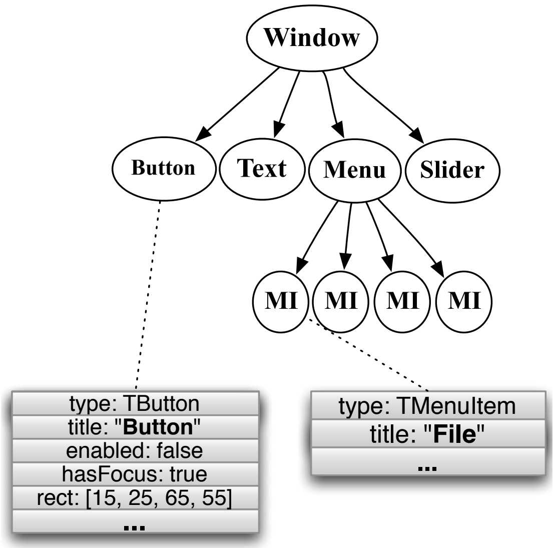
- **Visual testing**
  - Based on image recognition
  - Easy to understand, no programming skills needed
  - Solves most of maintenance problem
  - Introduces additional problems:
    - Performance of image processing
    - False positives and false negatives
      - the ambiguity associated with image locators
      - imprecision of image recognition feeds into oracles

- Model-based Testing

State of the art in UI testing

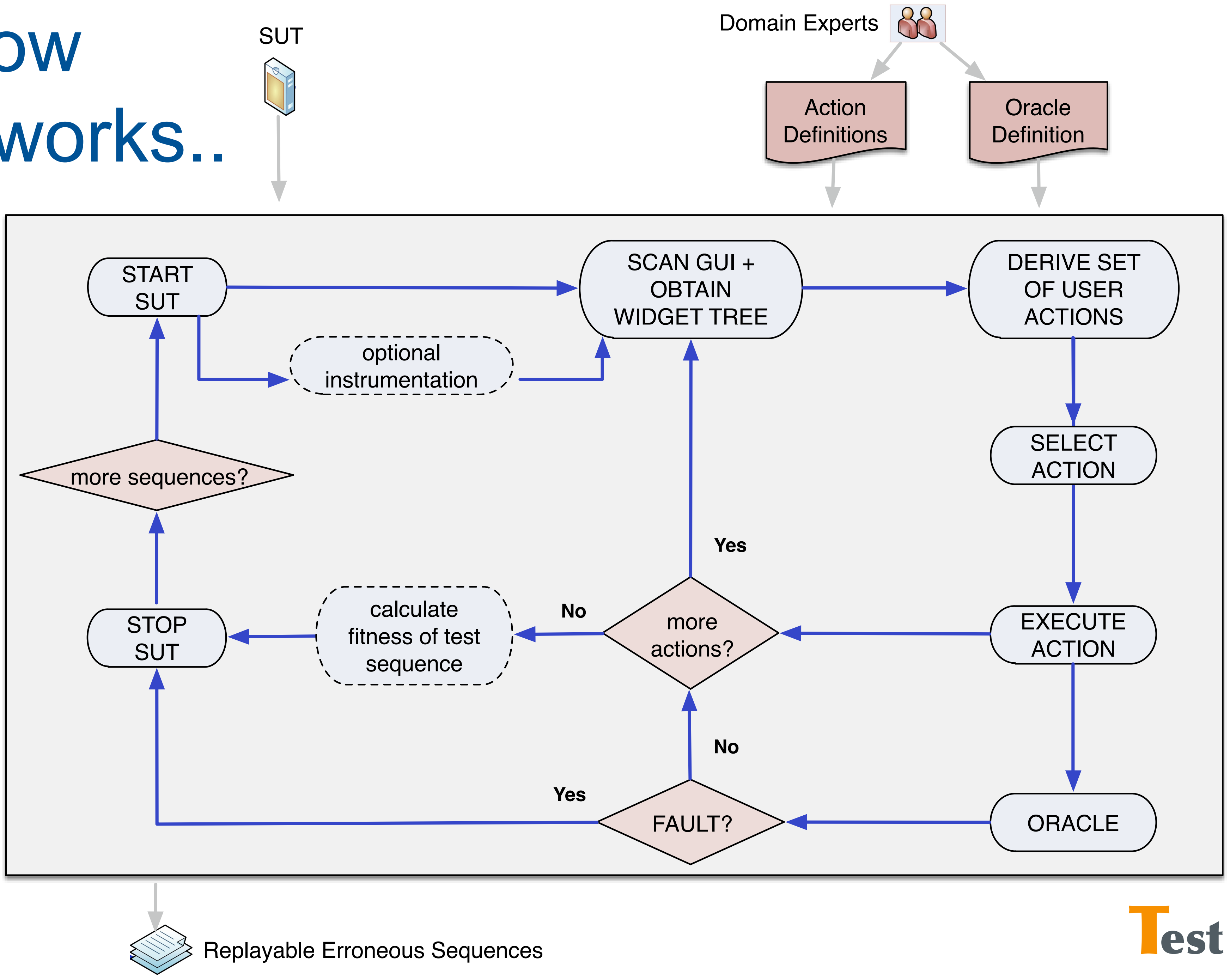
- Capture Replay
- Visual testing



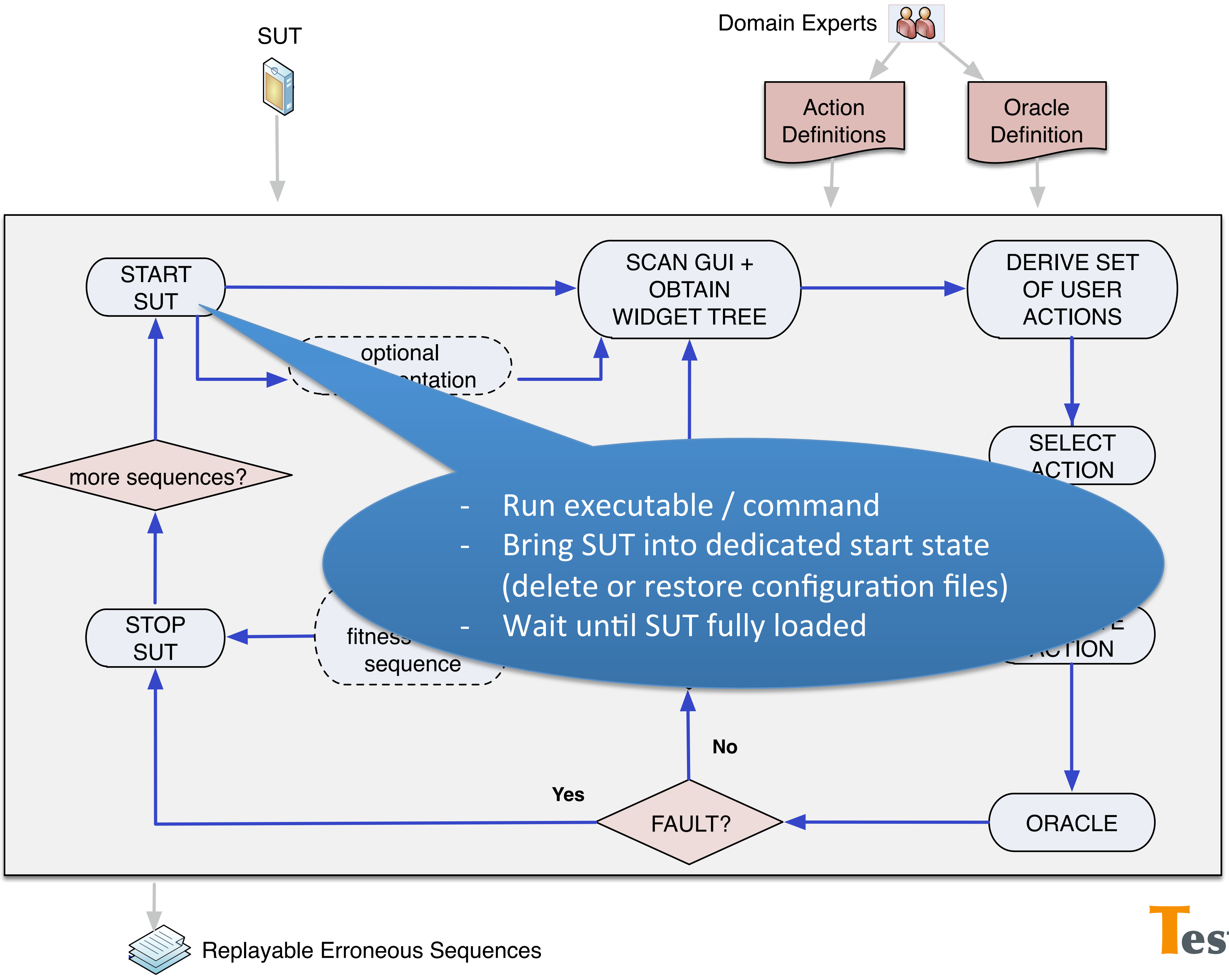


- **(ui) Model-based testing -- TESTAR**
  - Based on automatically inferred tree model of the UI
  - Tests sequences are derived automatically from the model
  - Executed sequences can be replayed
  - If UI changes so does the model/tests -> no maintenance of the tests
  - Programming skills are needed to define powerful oracles
    - It needs to be investigated more if this is really a problem….
    - Do we want testers to have programming skills?

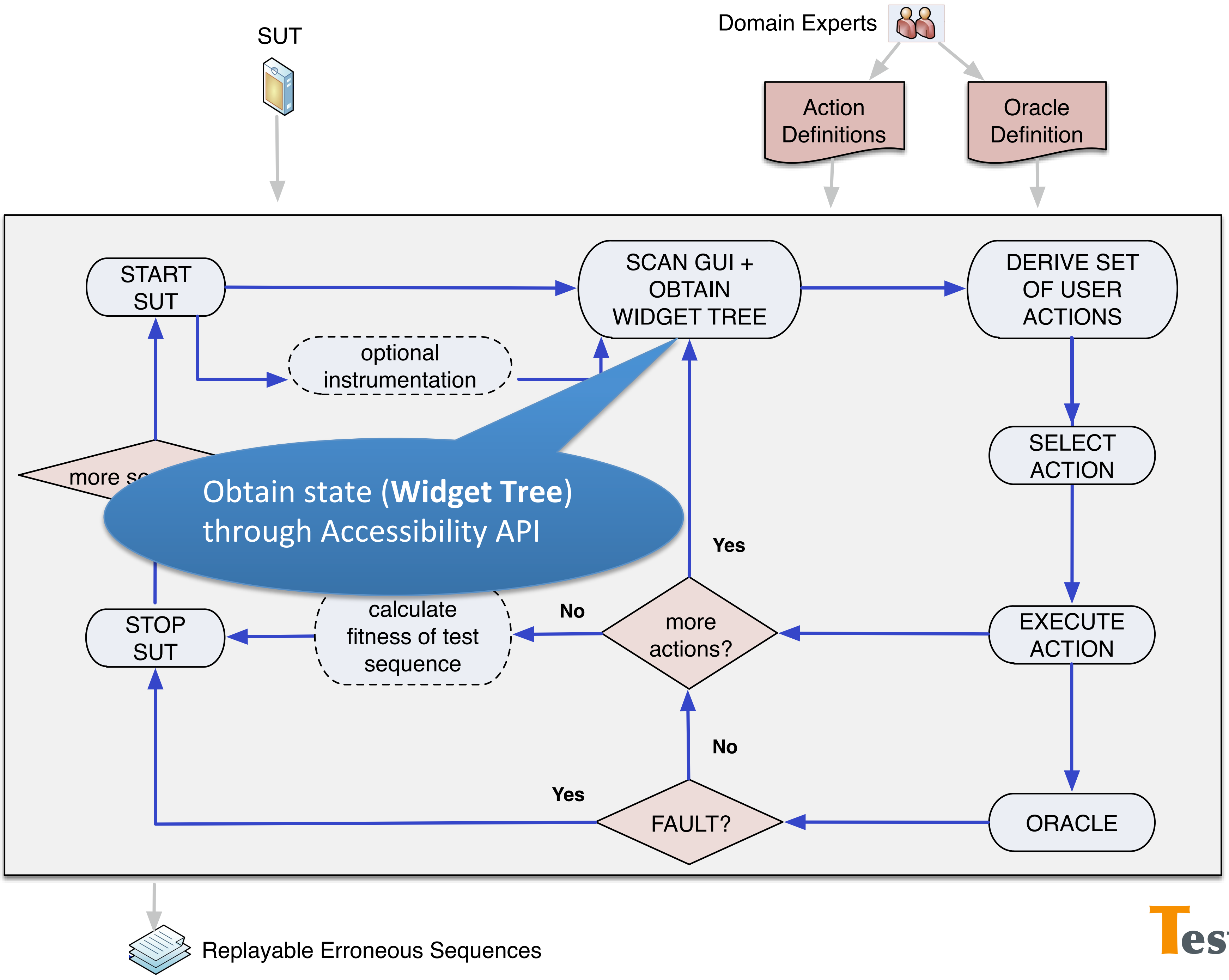
How it works..



Test*

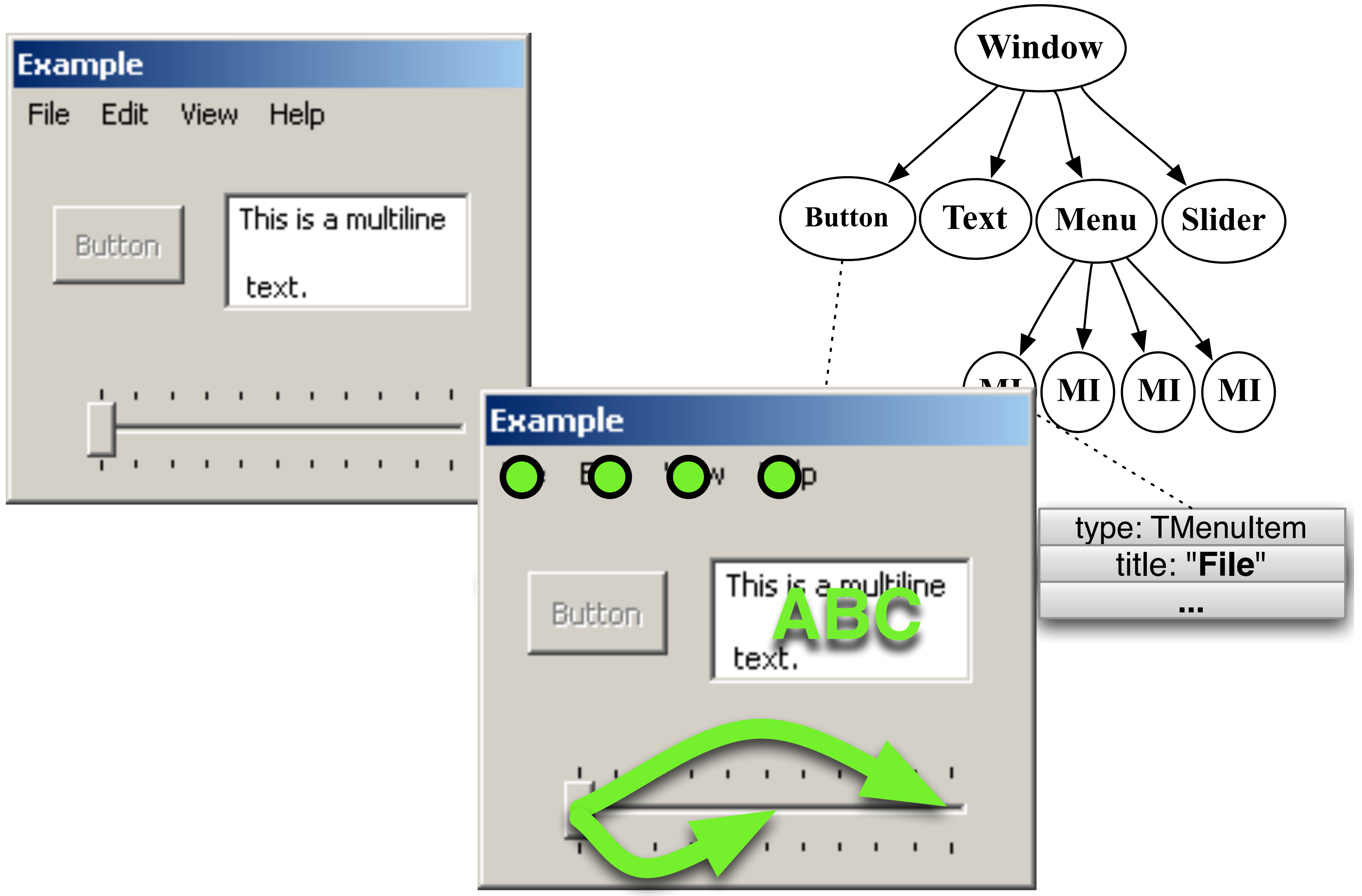


- Run executable / command
- Bring SUT into dedicated start state (delete or restore configuration files)
- Wait until SUT fully loaded



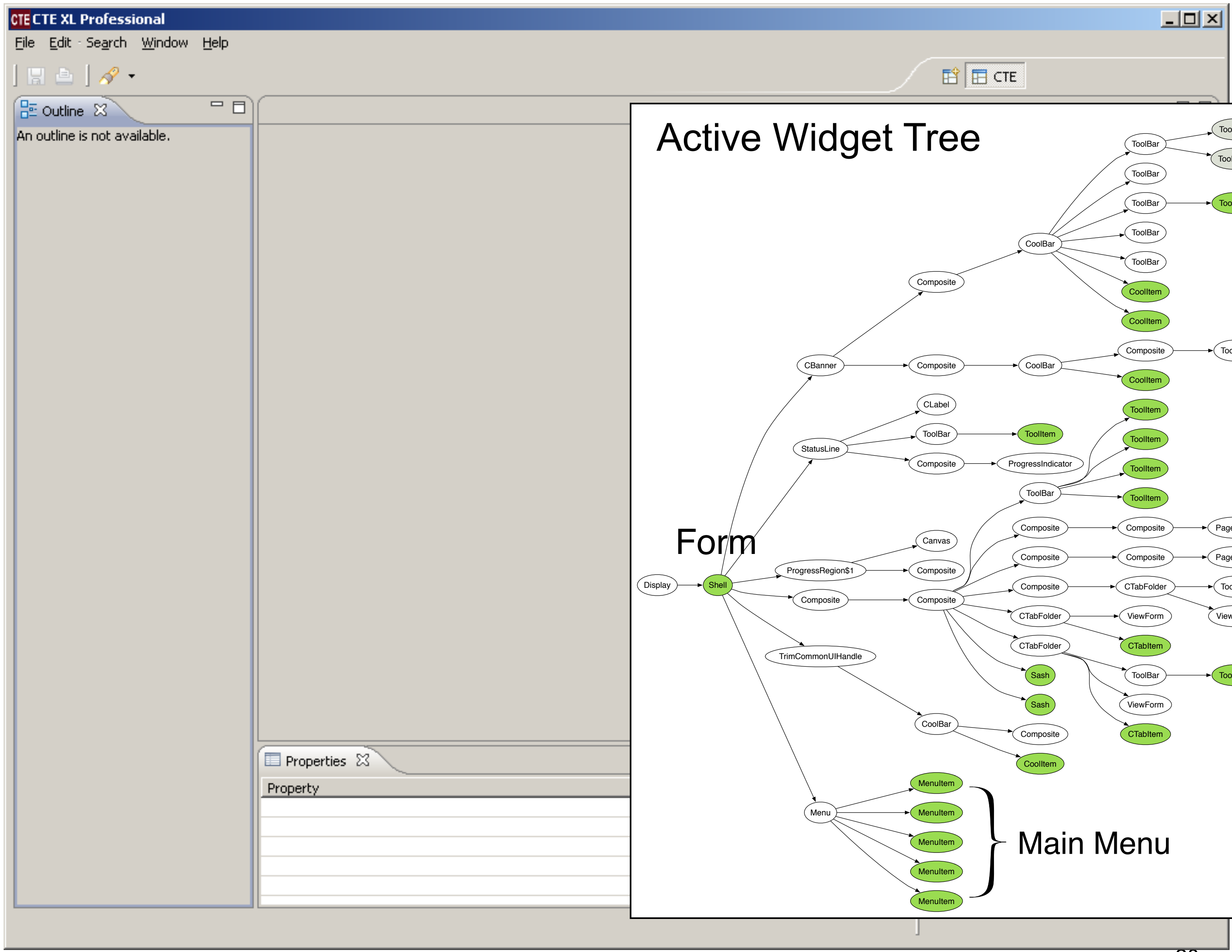
SUT
Domain Experts
Action Definitions
Oracle Definition
START SUT
SCAN GUI + OBTAIN WIDGET TREE
DERIVE SET OF USER ACTIONS
optional instrumentation
SELECT ACTION
more se
Obtain state (**Widget Tree**) through Accessibility API
Yes
No
calculate fitness of test sequence
STOP SUT
more actions?
EXECUTE ACTION
No
Yes
FAULT?
ORACLE
Replayable Erroneous Sequences

Widget Trees

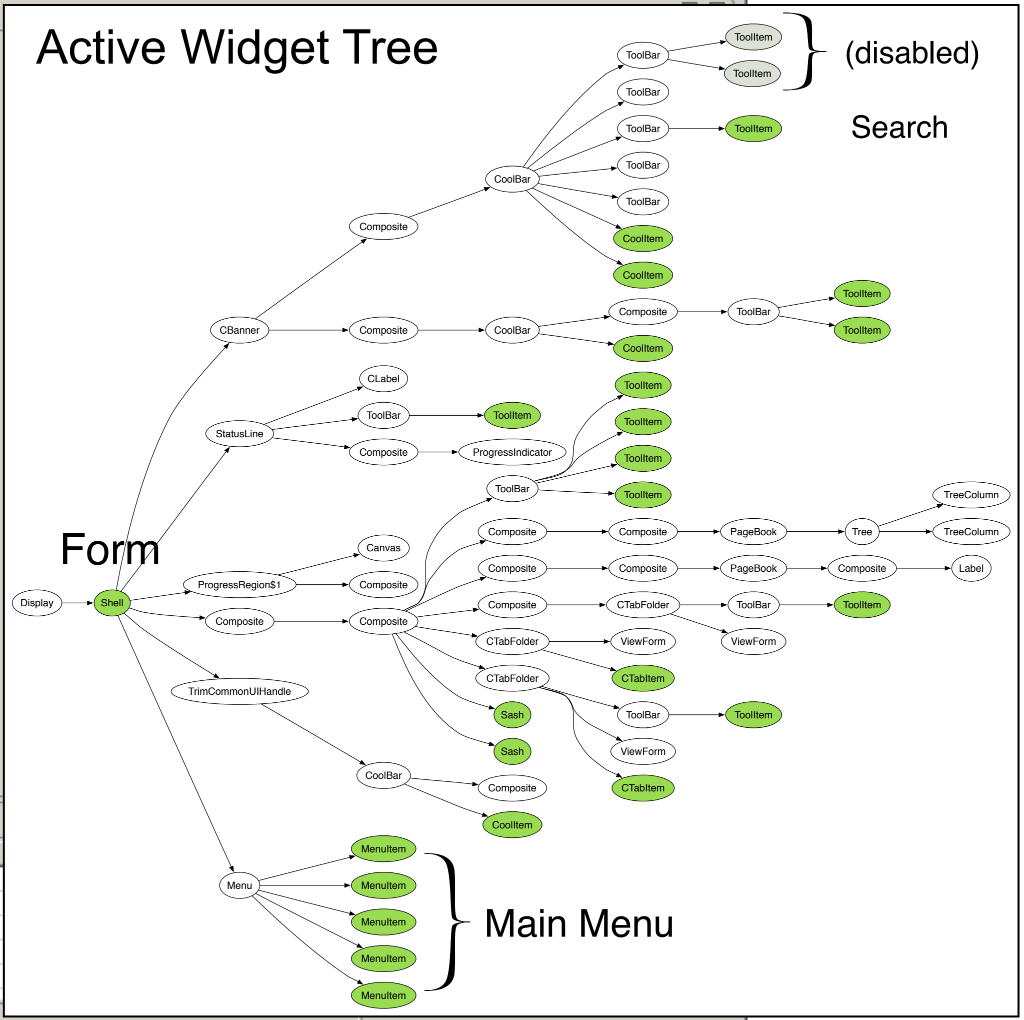


CTE XL Professional

berner & mattner
optimizing your development



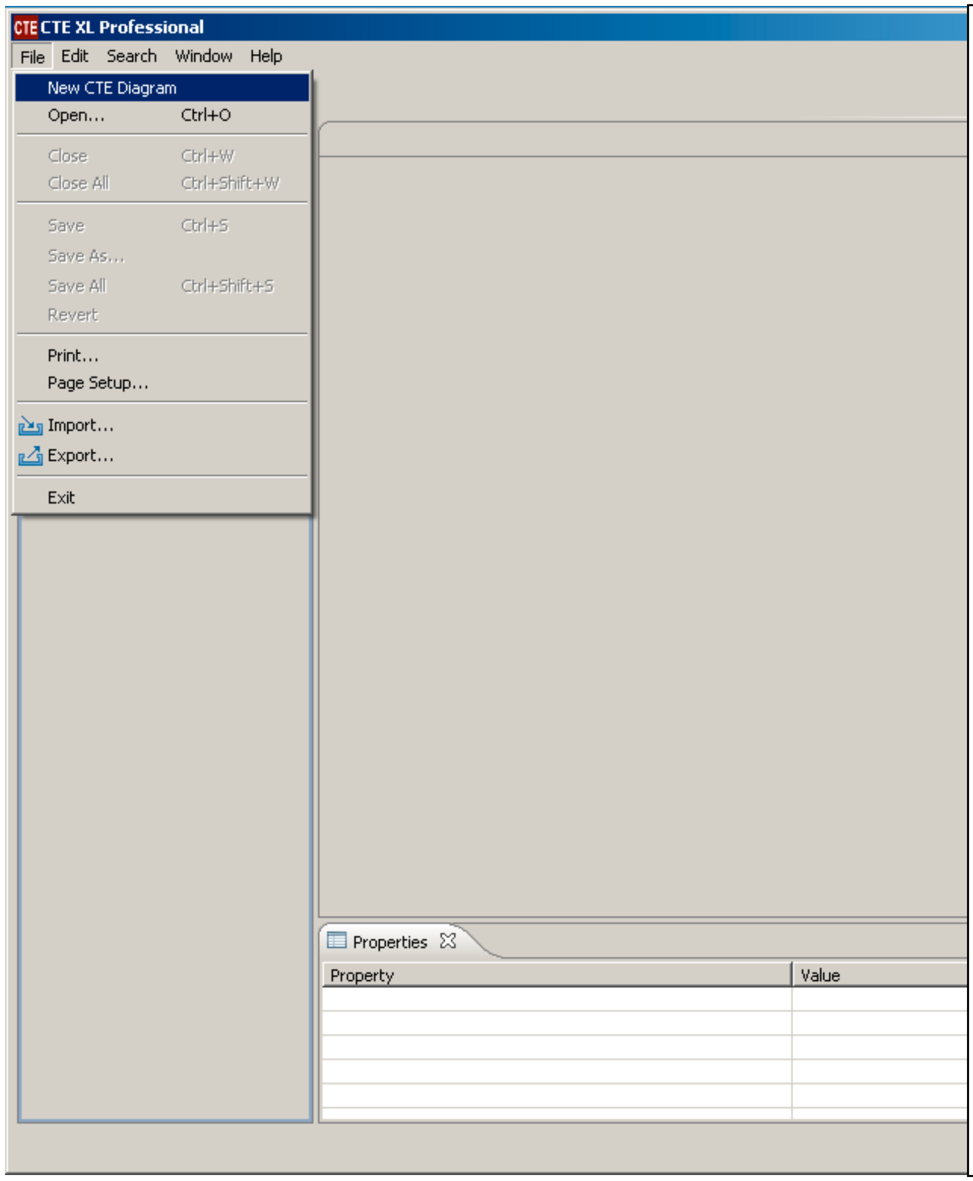
An outline is not available.



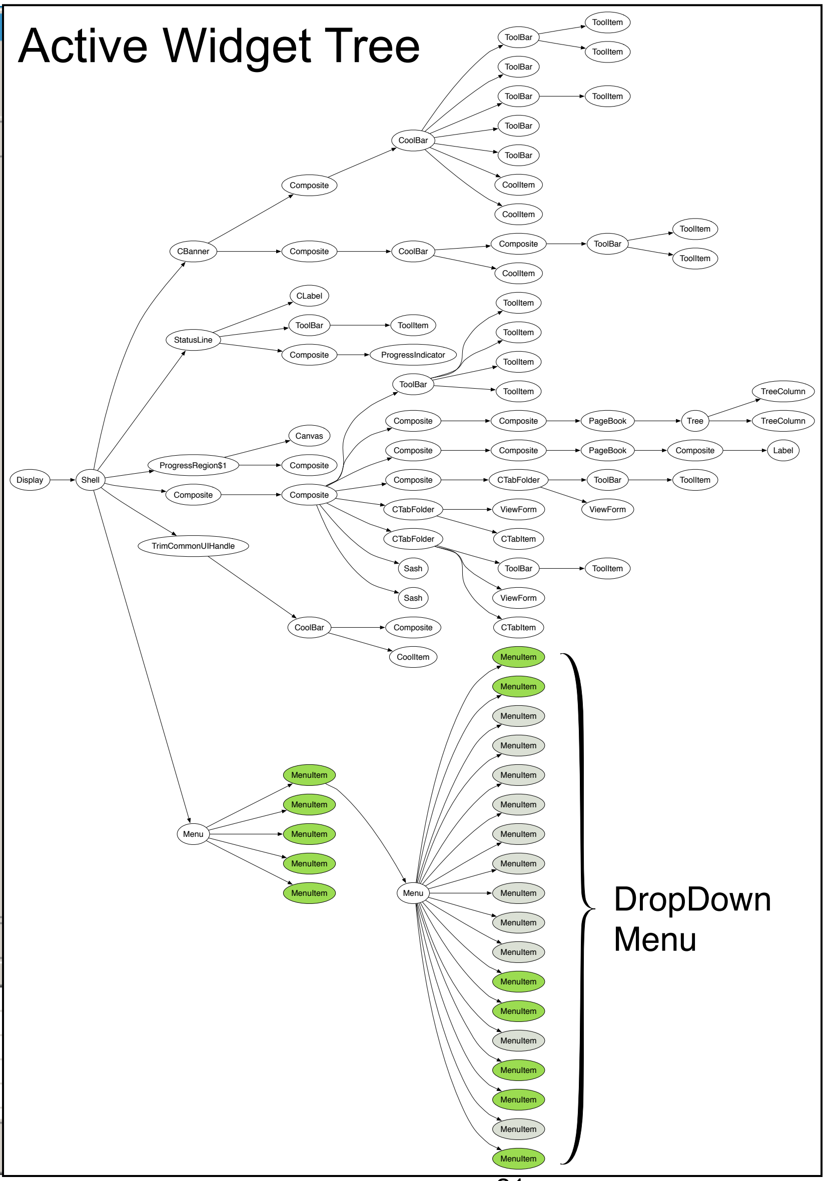
20

CTE XL Professional

berner & mattner
optimizing your development

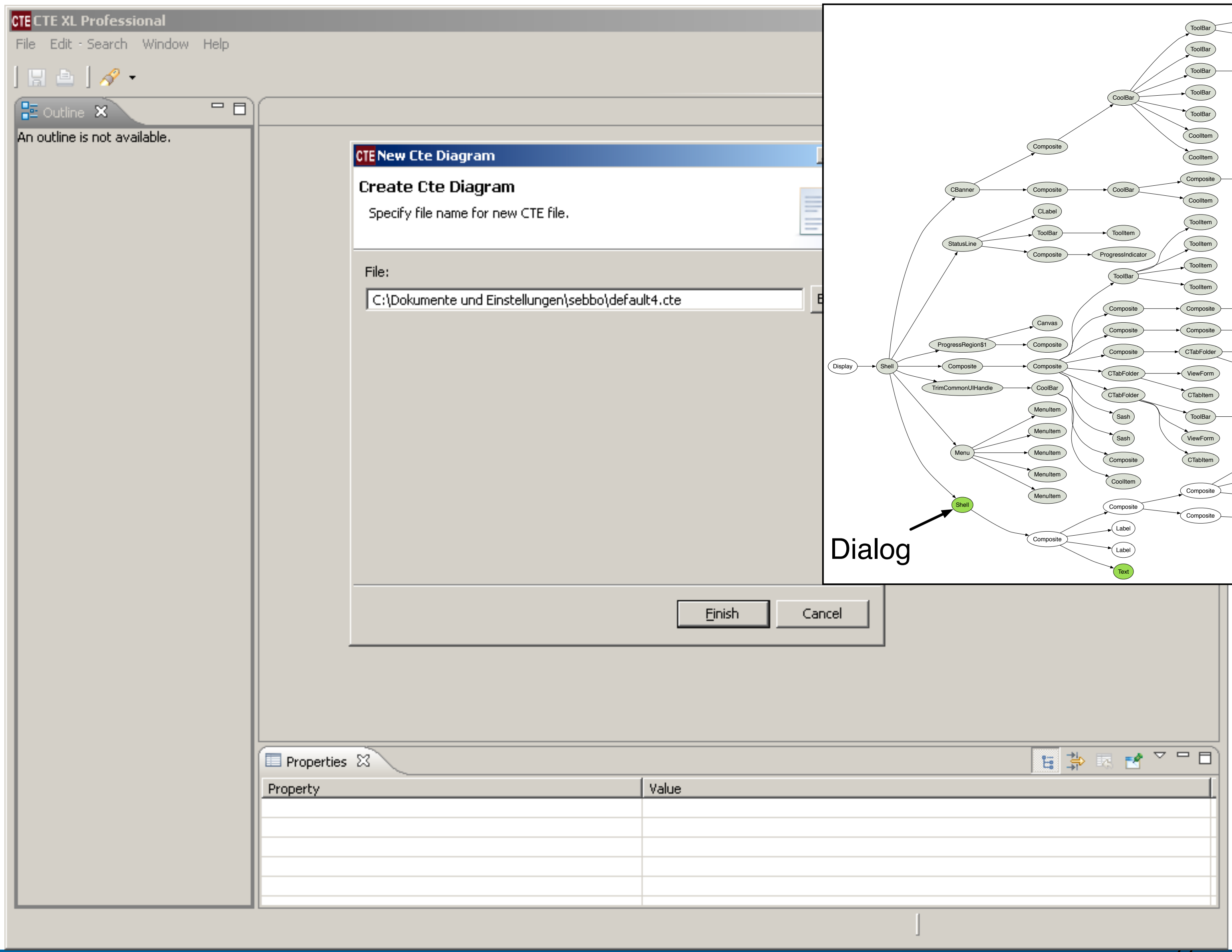


Active Widget Tree



CTE XL Professional

berner & mattner
optimizing your development



An outline is not available.

New Cte Diagram

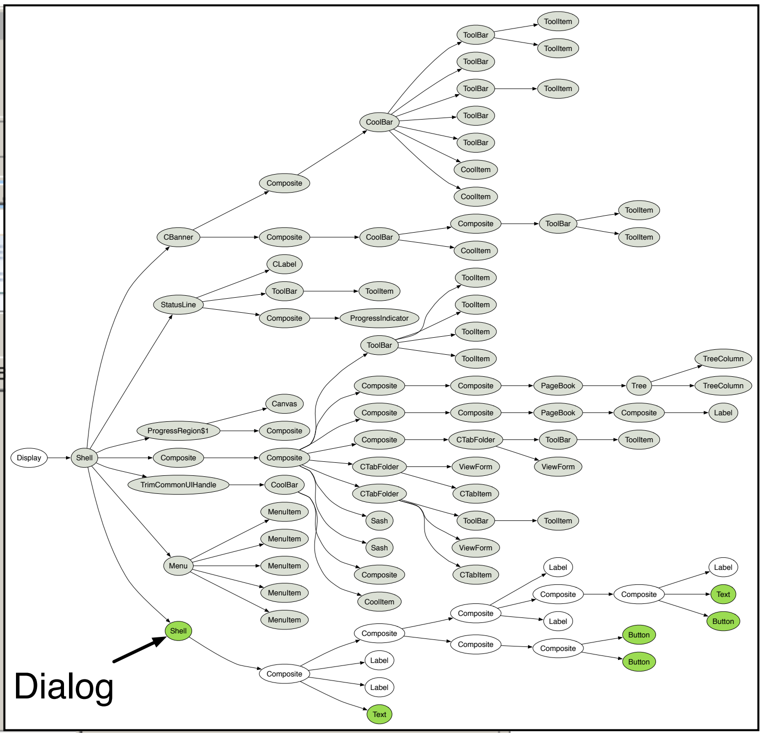
Create Cte Diagram

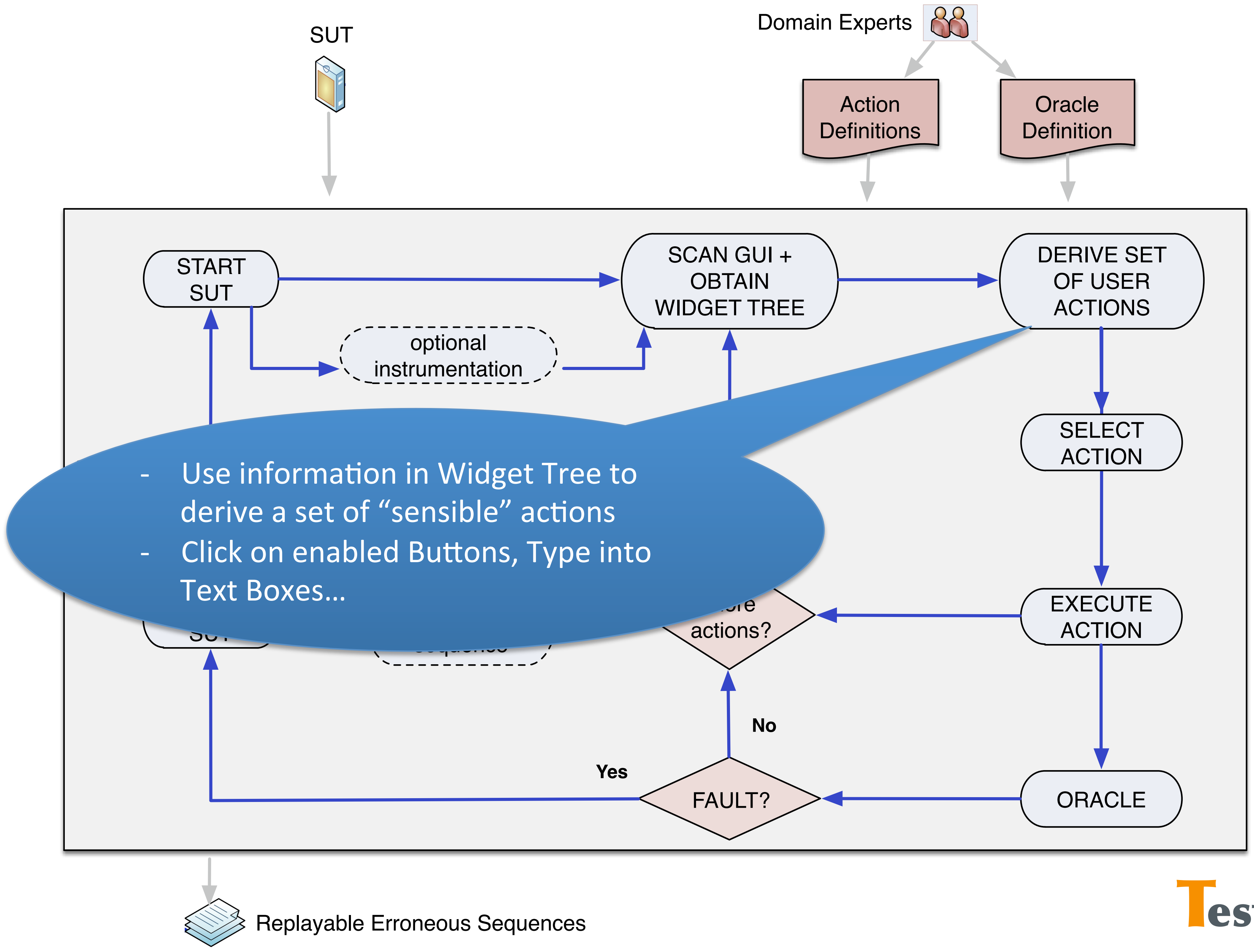
Specify file name for new CTE file.

File: C:\Dokumente und Einstellungen\sebbo\default4.cte

Finish Cancel

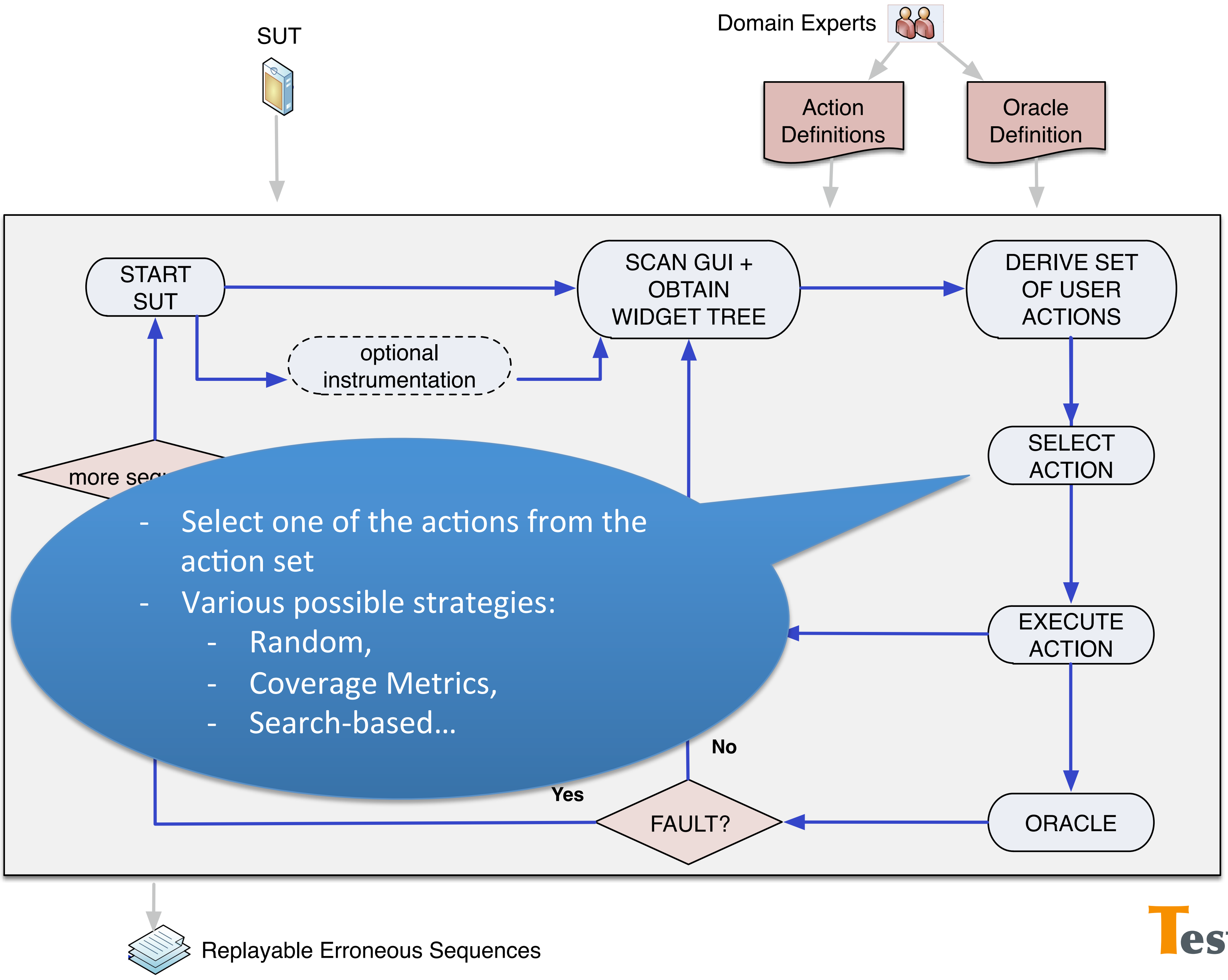
Property	Value





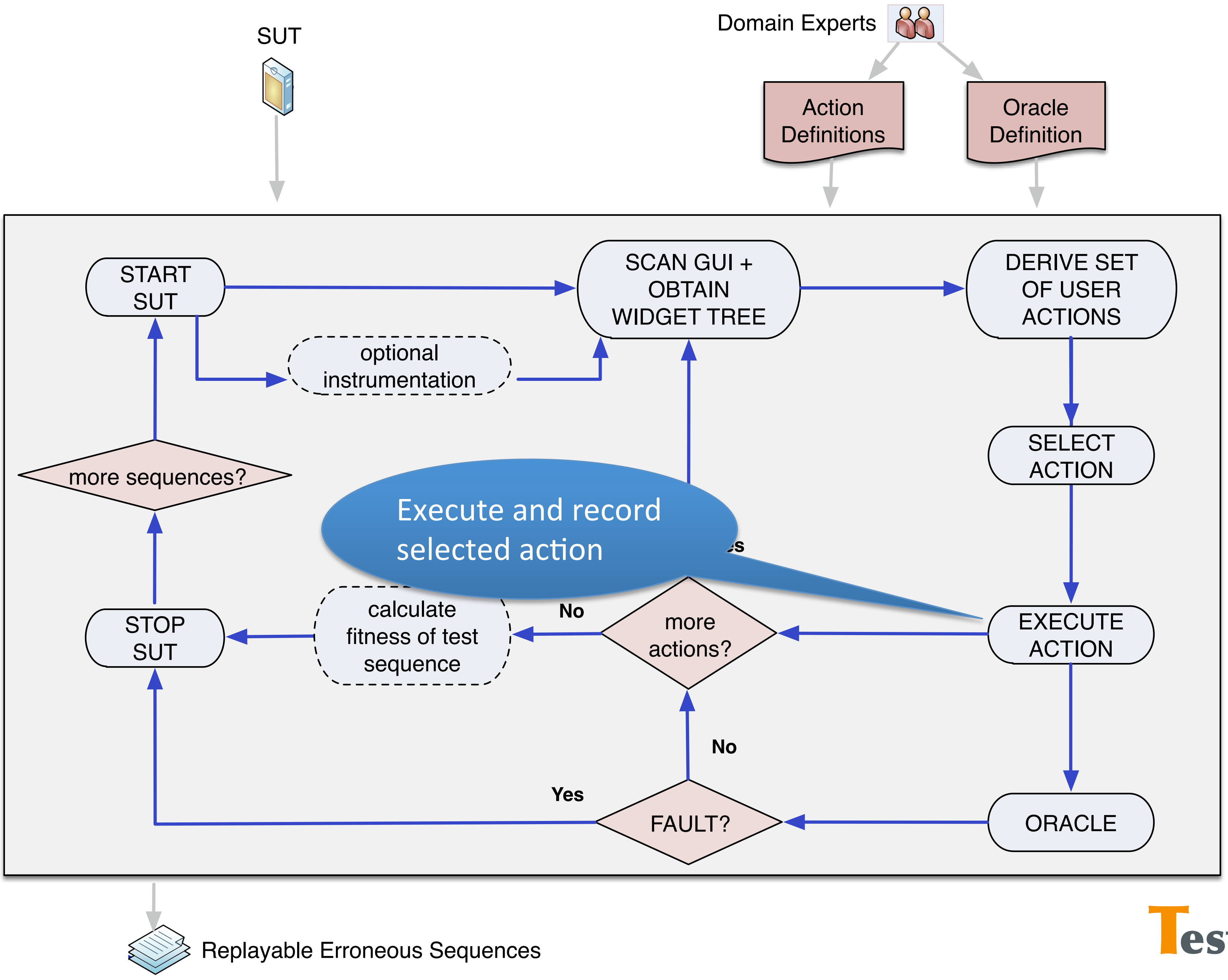
- Use information in Widget Tree to derive a set of “sensible” actions
- Click on enabled Buttons, Type into Text Boxes…

Test*

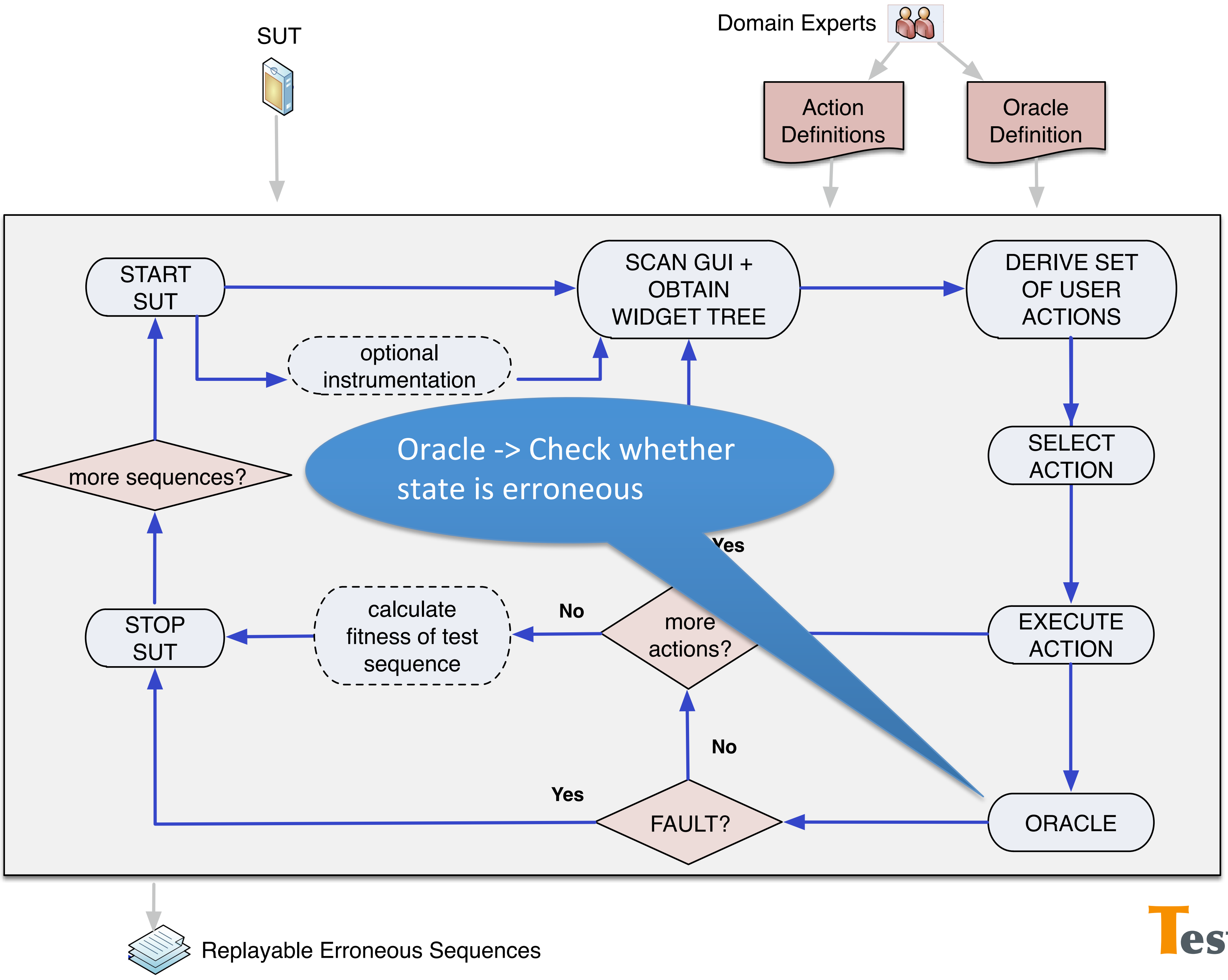


- Select one of the actions from the action set
- Various possible strategies:
  - Random,
  - Coverage Metrics,
  - Search-based…

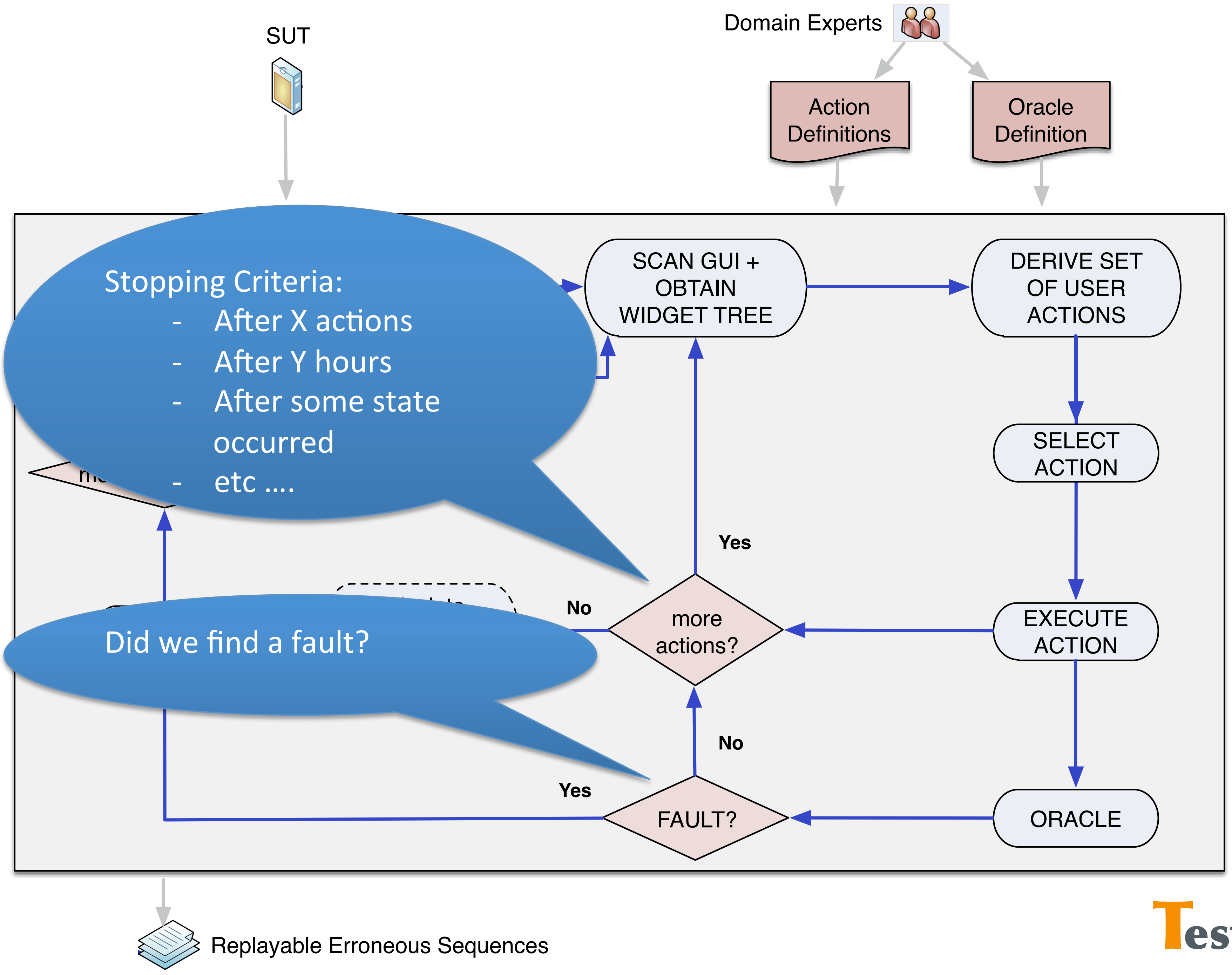
Test*


SUT
Domain Experts
Action Definitions
Oracle Definition
START SUT
SCAN GUI + OBTAIN WIDGET TREE
DERIVE SET OF USER ACTIONS
optional instrumentation
SELECT ACTION
more sequences?
Execute and record selected action
es
EXECUTE ACTION
No
more actions?
calculate fitness of test sequence
STOP SUT
No
Yes
FAULT?
ORACLE
Replayable Erroneous Sequences

Test*

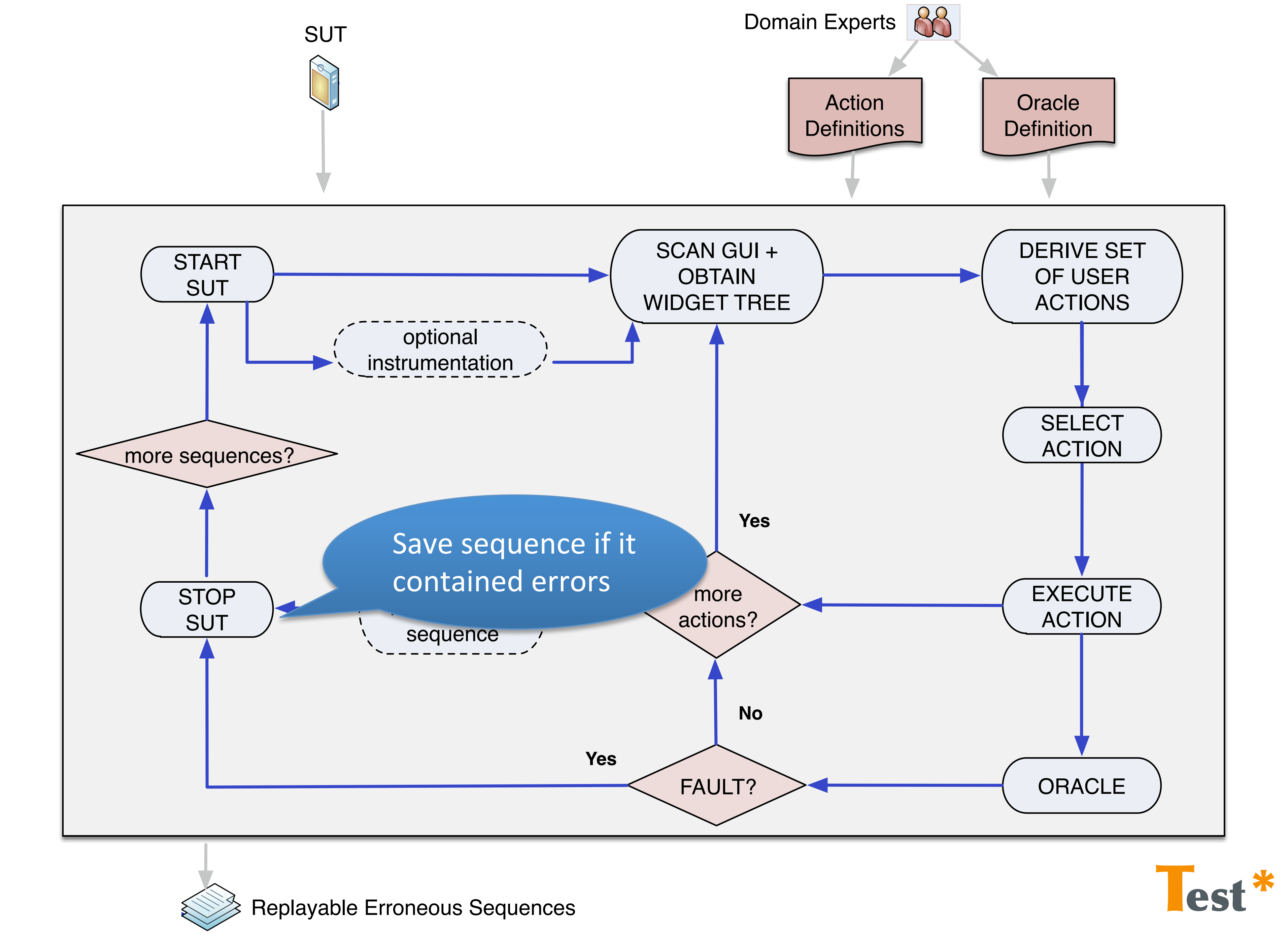

SUT
Domain Experts
Action Definitions
Oracle Definition
START SUT
SCAN GUI + OBTAIN WIDGET TREE
DERIVE SET OF USER ACTIONS
optional instrumentation
more sequences?
Oracle -> Check whether state is erroneous
SELECT ACTION
Yes
No
calculate fitness of test sequence
more actions?
EXECUTE ACTION
STOP SUT
No
Yes
FAULT?
ORACLE
Replayable Erroneous Sequences

Test*

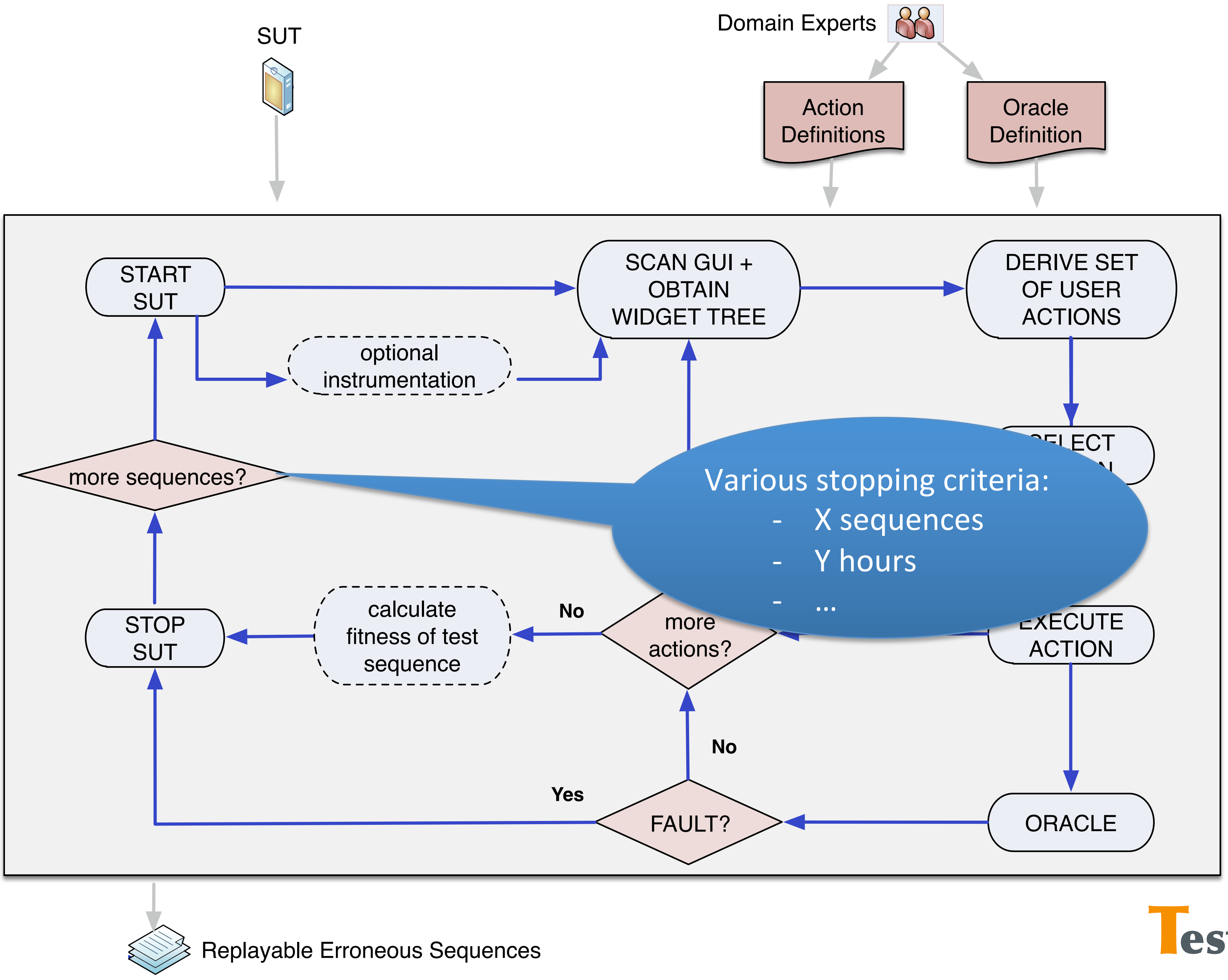


Stopping Criteria:
- After X actions
- After Y hours
- After some state occurred
- etc ....

Did we find a fault?


SUT
Domain Experts
Action Definitions
Oracle Definition
START SUT
SCAN GUI + OBTAIN WIDGET TREE
DERIVE SET OF USER ACTIONS
optional instrumentation
SELECT ACTION
more sequences?
Yes
Save sequence if it contained errors
STOP SUT
sequence
more actions?
EXECUTE ACTION
No
Yes
FAULT?
ORACLE
Replayable Erroneous Sequences

Test*

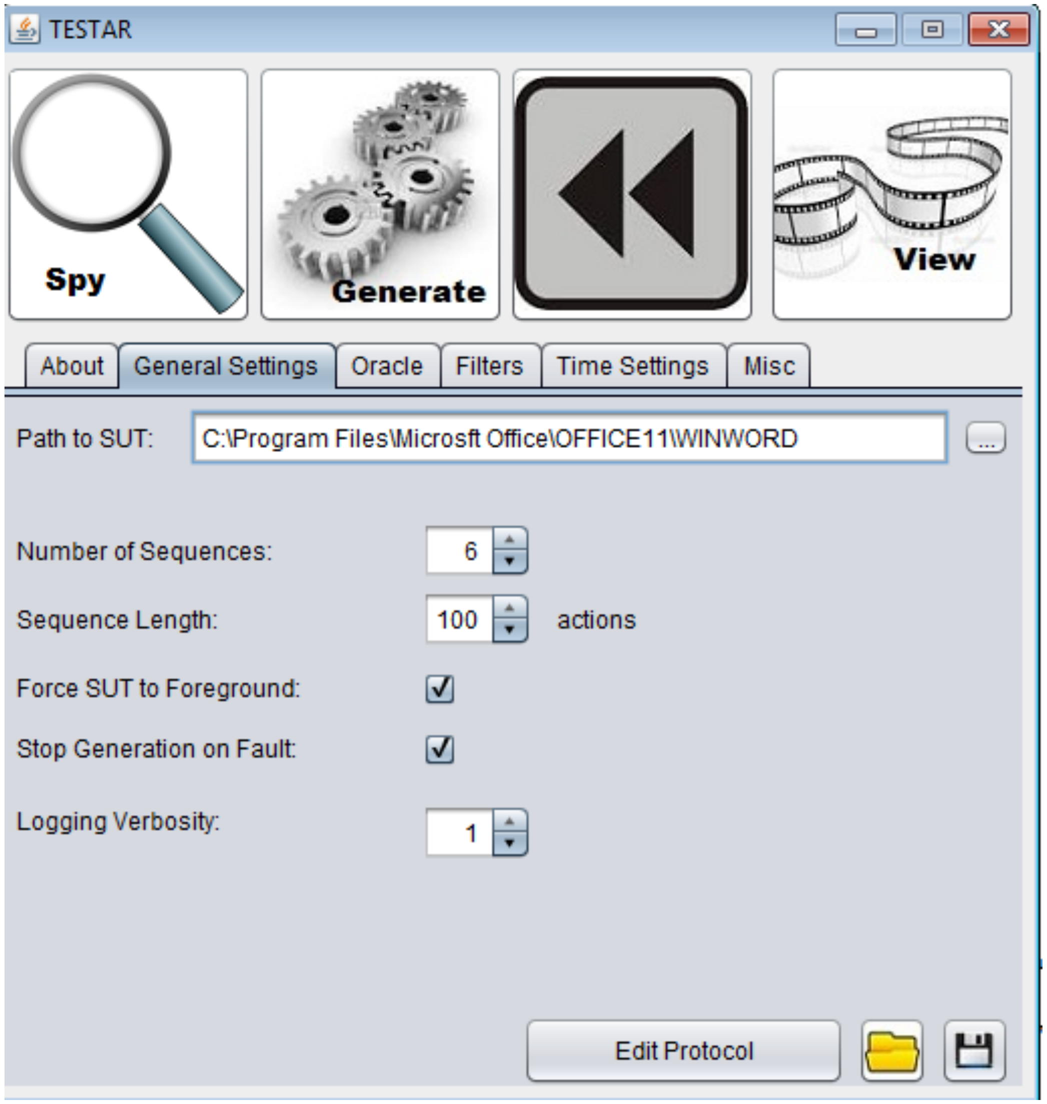


Various stopping criteria:
- X sequences
- Y hours
- …

TESTAR tool

READY

Set path the SUT



Path to SUT: C:\Program Files\Microsft Office\OFFICE11\WINWORD

Number of Sequences: 6

Sequence Length: 100 actions

Force SUT to Foreground: ✓

Stop Generation on Fault: ✓

Logging Verbosity: 1

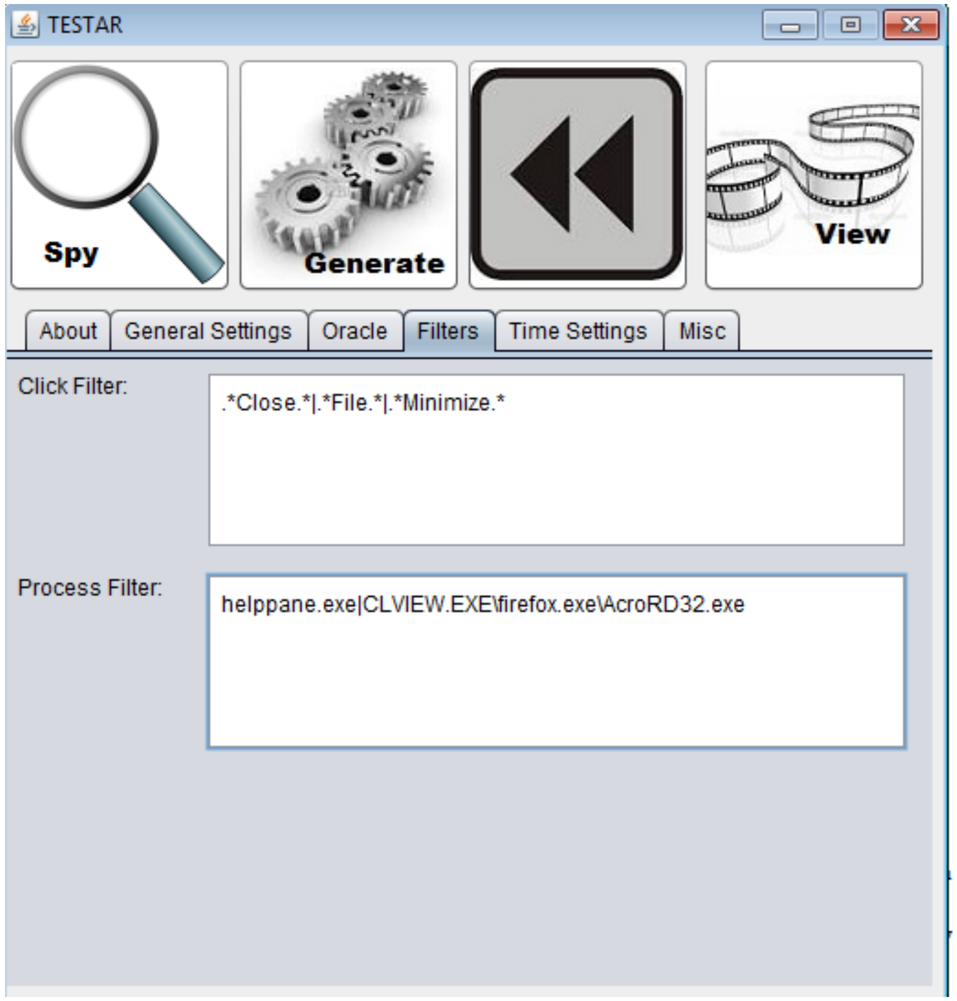
TESTAR tool

SET

Filter:

1) undesirable actions,
i.e. closing the application
al the time

2) Undesirable processes, for
example help panes in
acrobat, etc…….



Click Filter: .*Close.*|.*File.*|.*Minimize.*

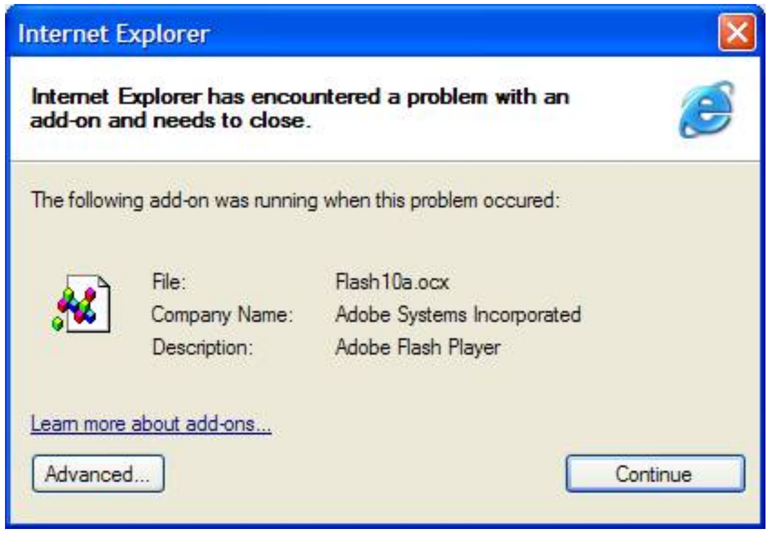
Process Filter: helppane.exe|CLVIEW.EXE\firefox.exe\AcroRD32.exe

GO!

See video at https://www.youtube.com/watch?v=PBs9jF_pLCs

Oracles for free

- What can we easily detect?
- Crashes



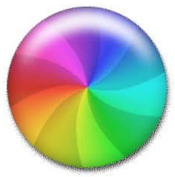
Internet Explorer

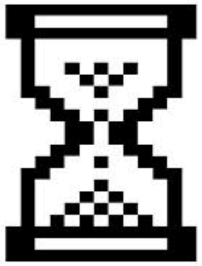
Internet Explorer has encountered a problem with an add-on and needs to close.

The following add-on was running when this problem occured:

File:	Flash10a.ocx
Company Name:	Adobe Systems Incorporated
Description:	Adobe Flash Player

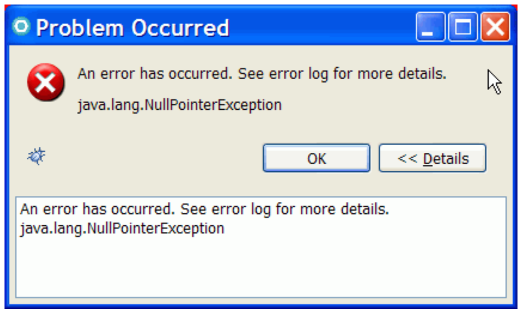
- Program freezes





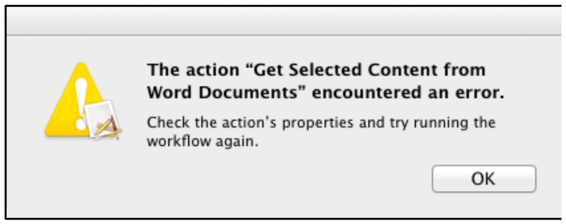
Cheap Oracles

- Critical message boxes
- Suspicious stdout / stderr

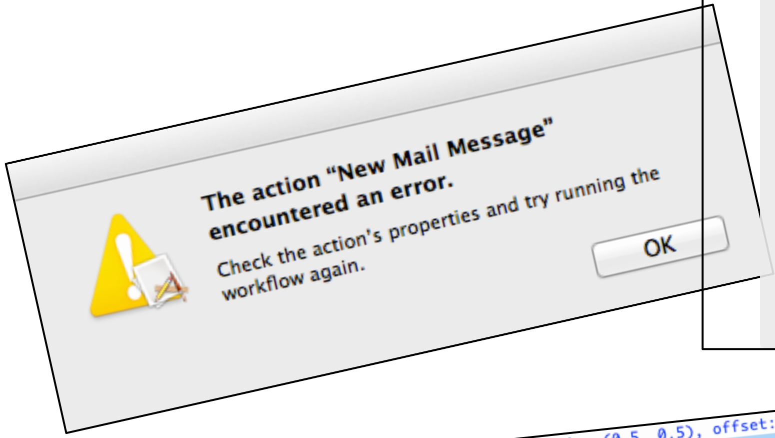


An error has occurred. See error log for more details.
java.lang.NullPointerException

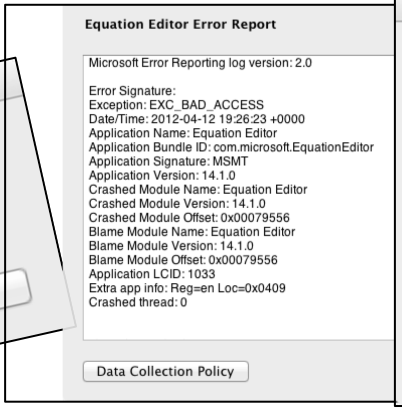
An error has occurred. See error log for more details.
java.lang.NullPointerException



The action "Get Selected Content from Word Documents" encountered an error.
Check the action's properties and try running the workflow again.

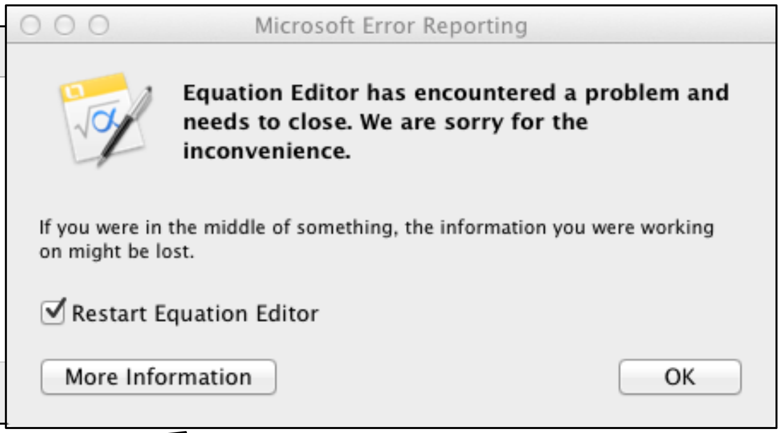


The action "New Mail Message" encountered an error.
Check the action's properties and try running the workflow again.



Equation Editor Error Report

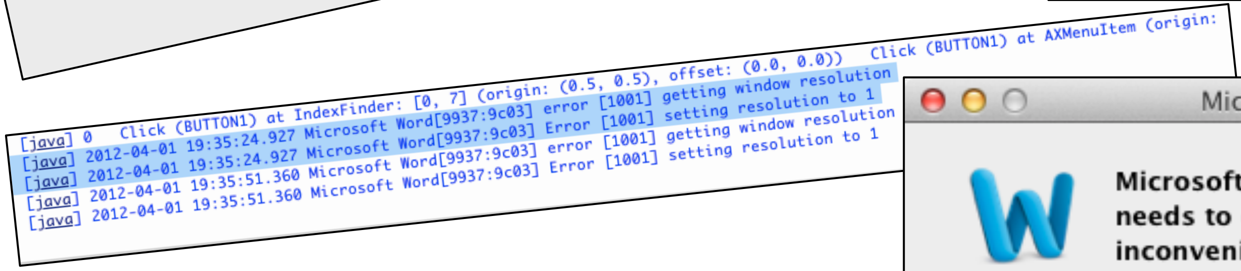
```
Microsoft Error Reporting log version: 2.0

Error Signature:
Exception: EXC_BAD_ACCESS
Date/Time: 2012-04-12 19:26:23 +0000
Application Name: Equation Editor
Application Bundle ID: com.microsoft.EquationEditor
Application Signature: MSMT
Application Version: 14.1.0
Crashed Module Name: Equation Editor
Crashed Module Version: 14.1.0
Crashed Module Offset: 0x00079556
Blame Module Name: Equation Editor
Blame Module Version: 14.1.0
Blame Module Offset: 0x00079556
Application LCID: 1033
Extra app info: Reg=en Loc=0x0409
Crashed thread: 0
```



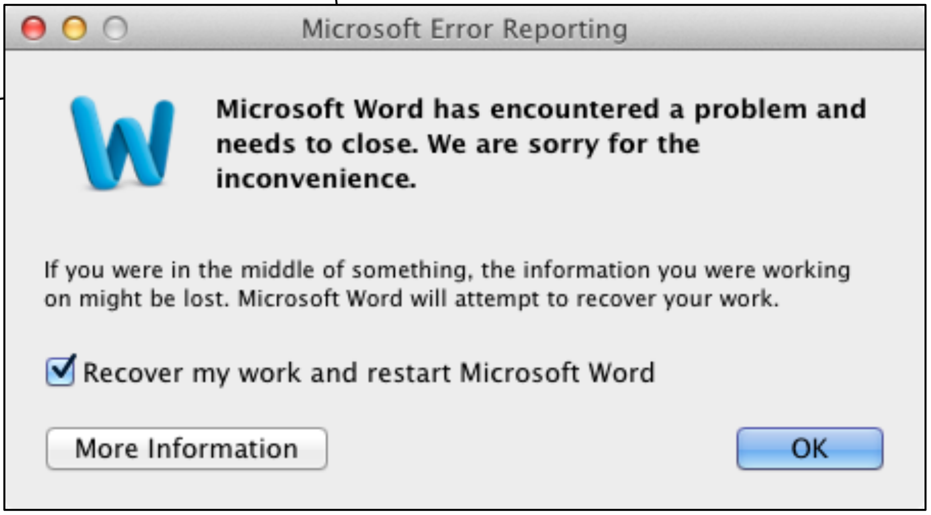
Equation Editor has encountered a problem and needs to close. We are sorry for the inconvenience.

If you were in the middle of something, the information you were working on might be lost.

[✓] Restart Equation Editor



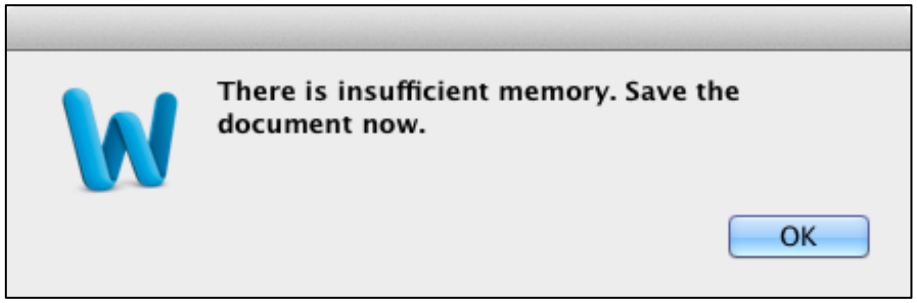
```
                                                                                       Click (BUTTON1) at AXMenuItem (origin:
[java] 0   Click (BUTTON1) at IndexFinder: [0, 7] (origin: (0.5, 0.5), offset: (0.0, 0.0))
[java] 2012-04-01 19:35:24.927 Microsoft Word[9937:9c03] error [1001] getting window resolution
[java] 2012-04-01 19:35:24.927 Microsoft Word[9937:9c03] Error [1001] setting resolution to 1
[java] 2012-04-01 19:35:51.360 Microsoft Word[9937:9c03] error [1001] getting window resolution
[java] 2012-04-01 19:35:51.360 Microsoft Word[9937:9c03] Error [1001] setting resolution to 1
```



Microsoft Word has encountered a problem and needs to close. We are sorry for the inconvenience.

If you were in the middle of something, the information you were working on might be lost. Microsoft Word will attempt to recover your work.

[✓] Recover my work and restart Microsoft Word



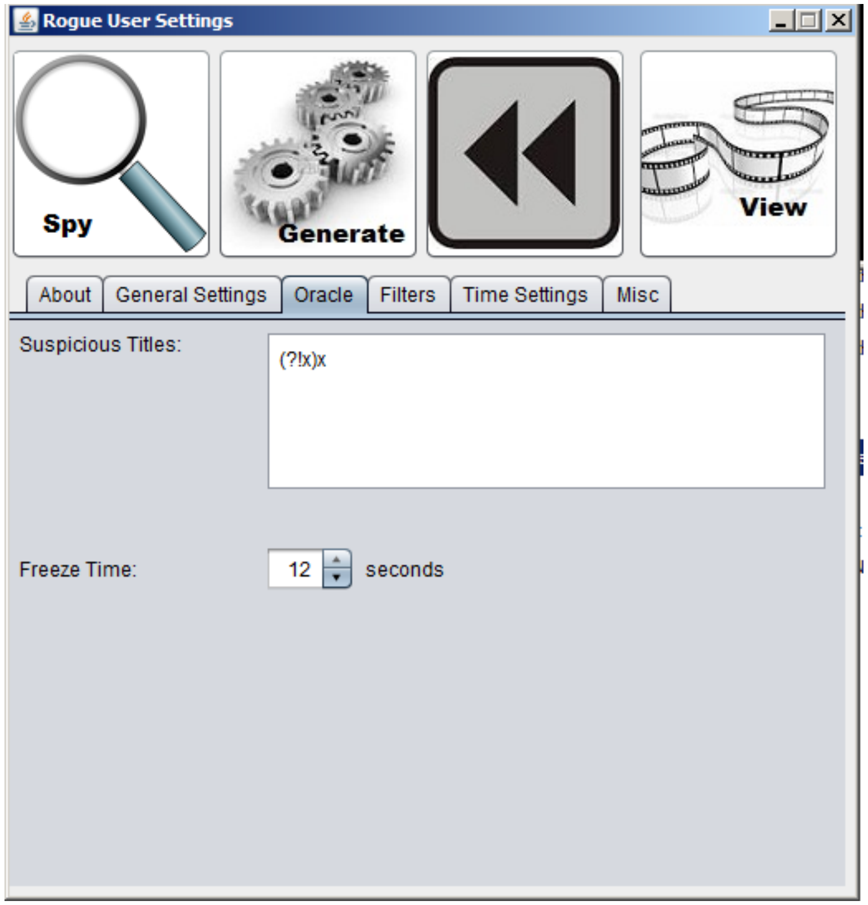
There is insufficient memory. Save the document now.

Specifying Cheap Oracles

- Simply with regular Expressions
- For example:

  .*NullPointerException

  .*|[Ee]rror|[Pp]roblem



Suspicious Titles: (?!x)x

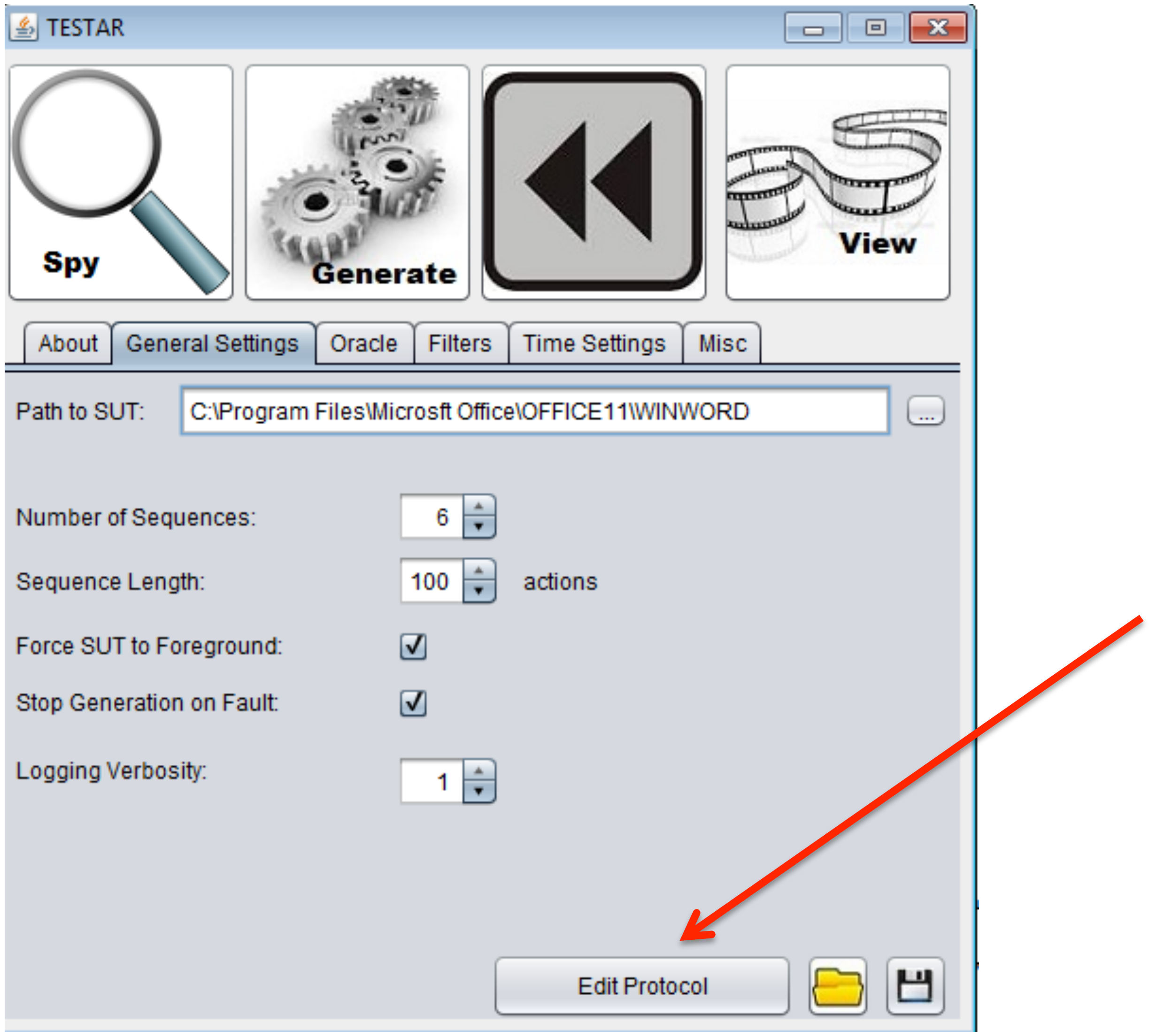
Freeze Time: 12 seconds

More sophistication needs work

- Actions
  - Action detection
  - Action selection
  - Sometimes a trial/error process
    - Random selection = like a child, just much faster
    - Printing, file copying / moving / deleting
    - Starts other Processes
    - Rights management, dedicated user accounts, disallow actions

- Oracles that need programming

How? Edit the protocol



Path to SUT: C:\Program Files\Microsft Office\OFFICE11\WINWORD

Number of Sequences: 6

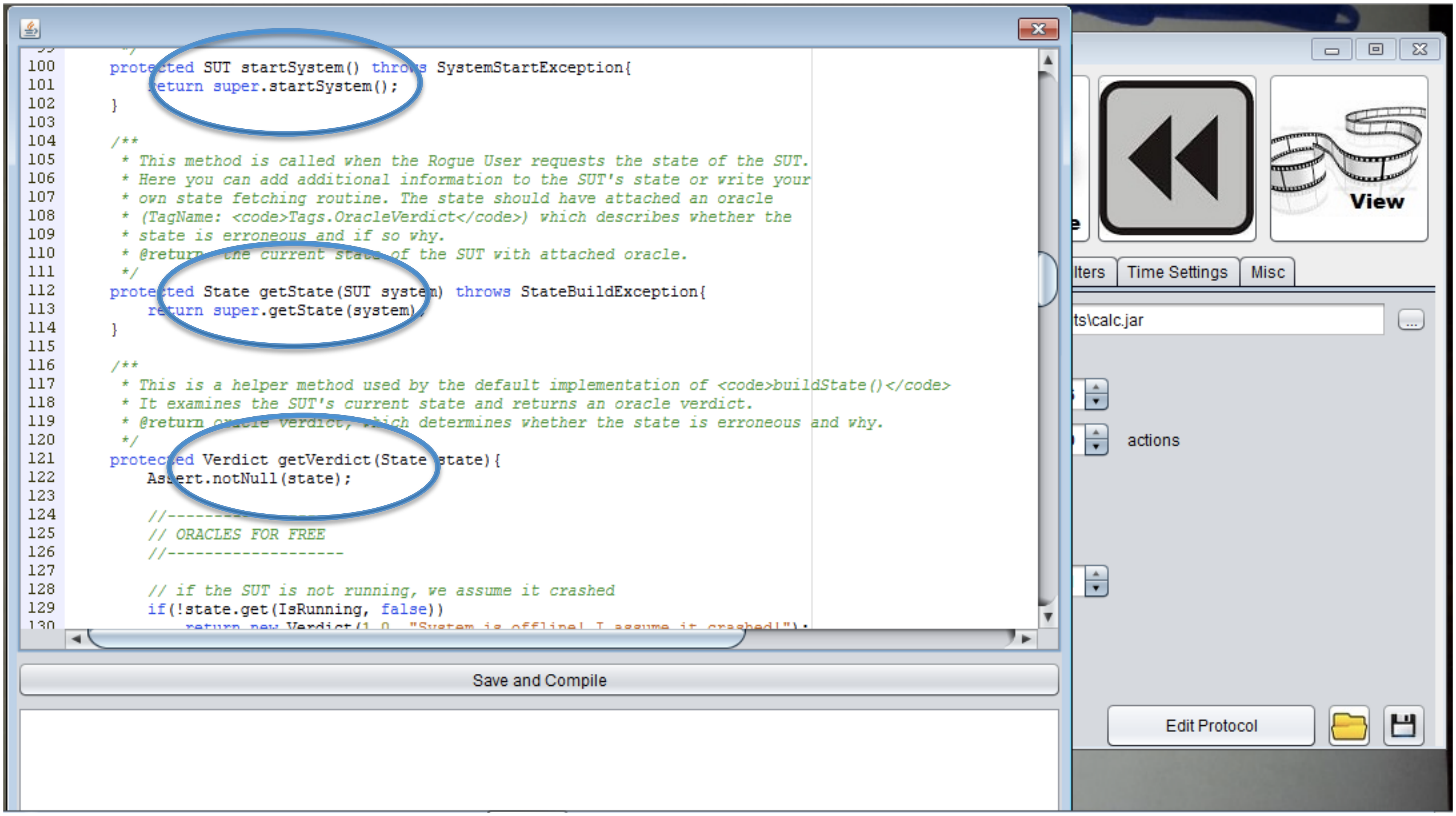
Sequence Length: 100 actions

Force SUT to Foreground: ✓

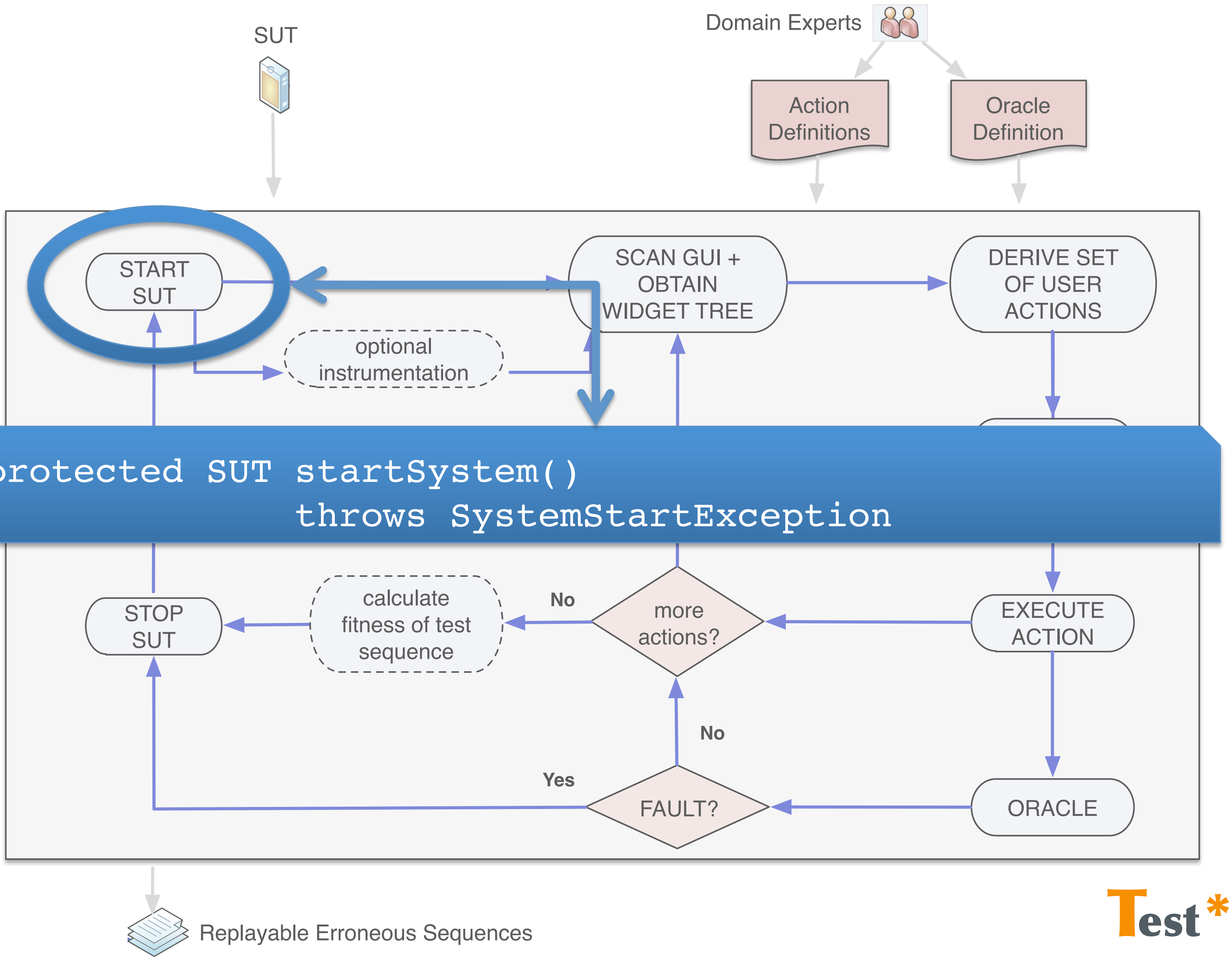
Stop Generation on Fault: ✓

Logging Verbosity: 1

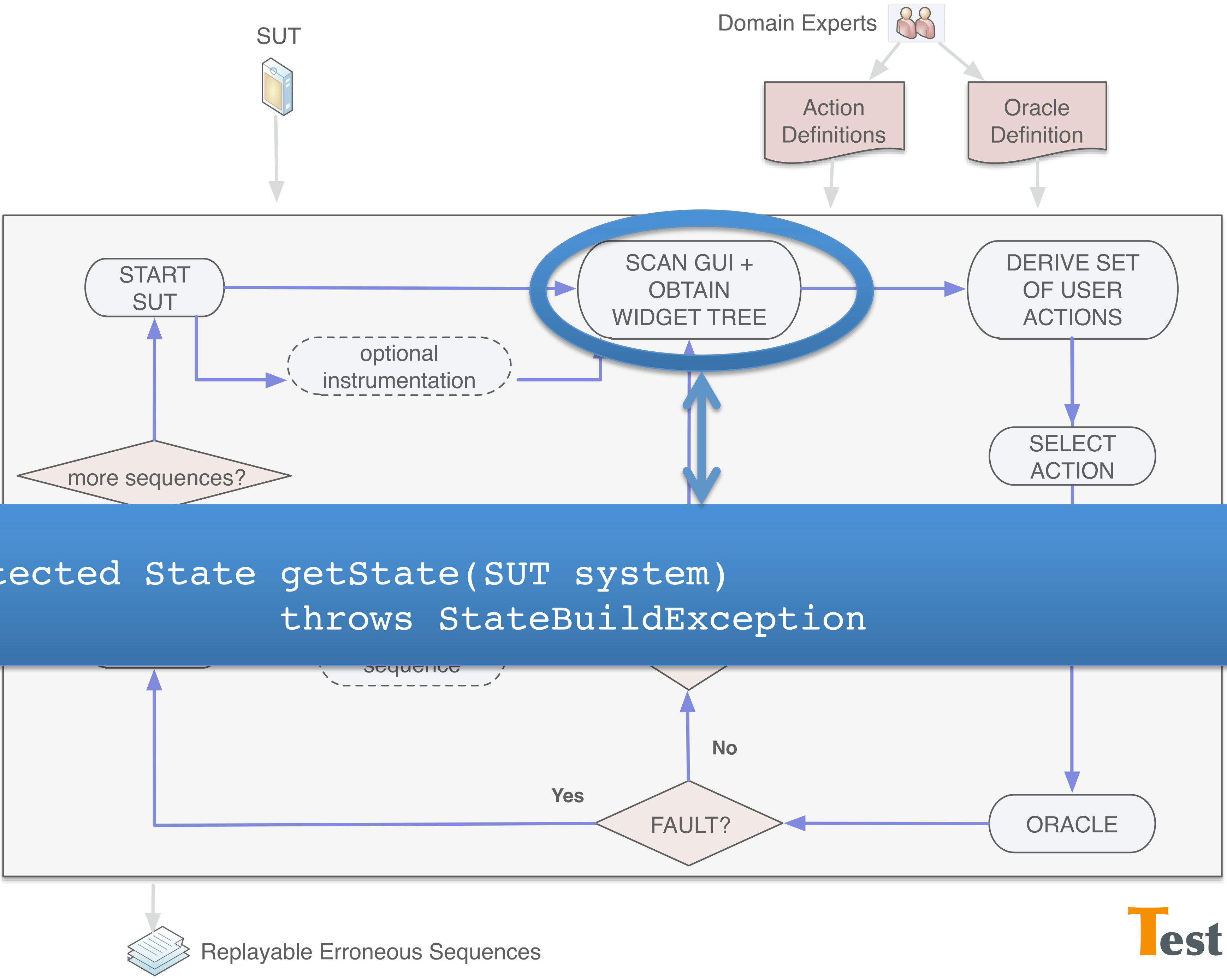
The protocol editor



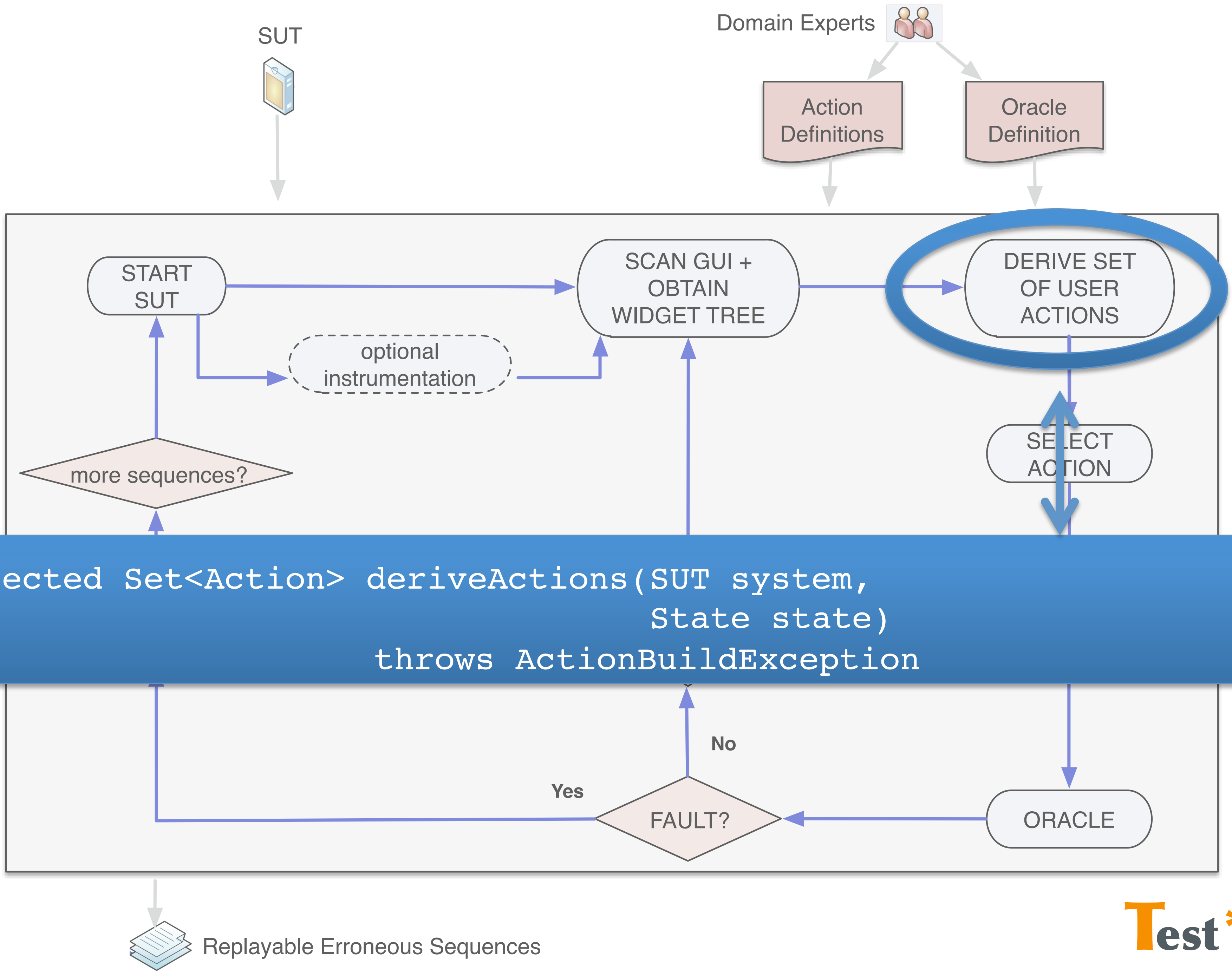
```
     */
    protected SUT startSystem() throws SystemStartException{
        return super.startSystem();
    }

    /**
     * This method is called when the Rogue User requests the state of the SUT.
     * Here you can add additional information to the SUT's state or write your
     * own state fetching routine. The state should have attached an oracle
     * (TagName: <code>Tags.OracleVerdict</code>) which describes whether the
     * state is erroneous and if so why.
     * @return  the current state of the SUT with attached oracle.
     */
    protected State getState(SUT system) throws StateBuildException{
        return super.getState(system);
    }

    /**
     * This is a helper method used by the default implementation of <code>buildState()</code>
     * It examines the SUT's current state and returns an oracle verdict.
     * @return oracle verdict, which determines whether the state is erroneous and why.
     */
    protected Verdict getVerdict(State state){
        Assert.notNull(state);

        //-------------------
        // ORACLES FOR FREE
        //-------------------

        // if the SUT is not running, we assume it crashed
        if(!state.get(IsRunning, false))
            return new Verdict(1.0, "System is offline! I assume it crashed!");
```



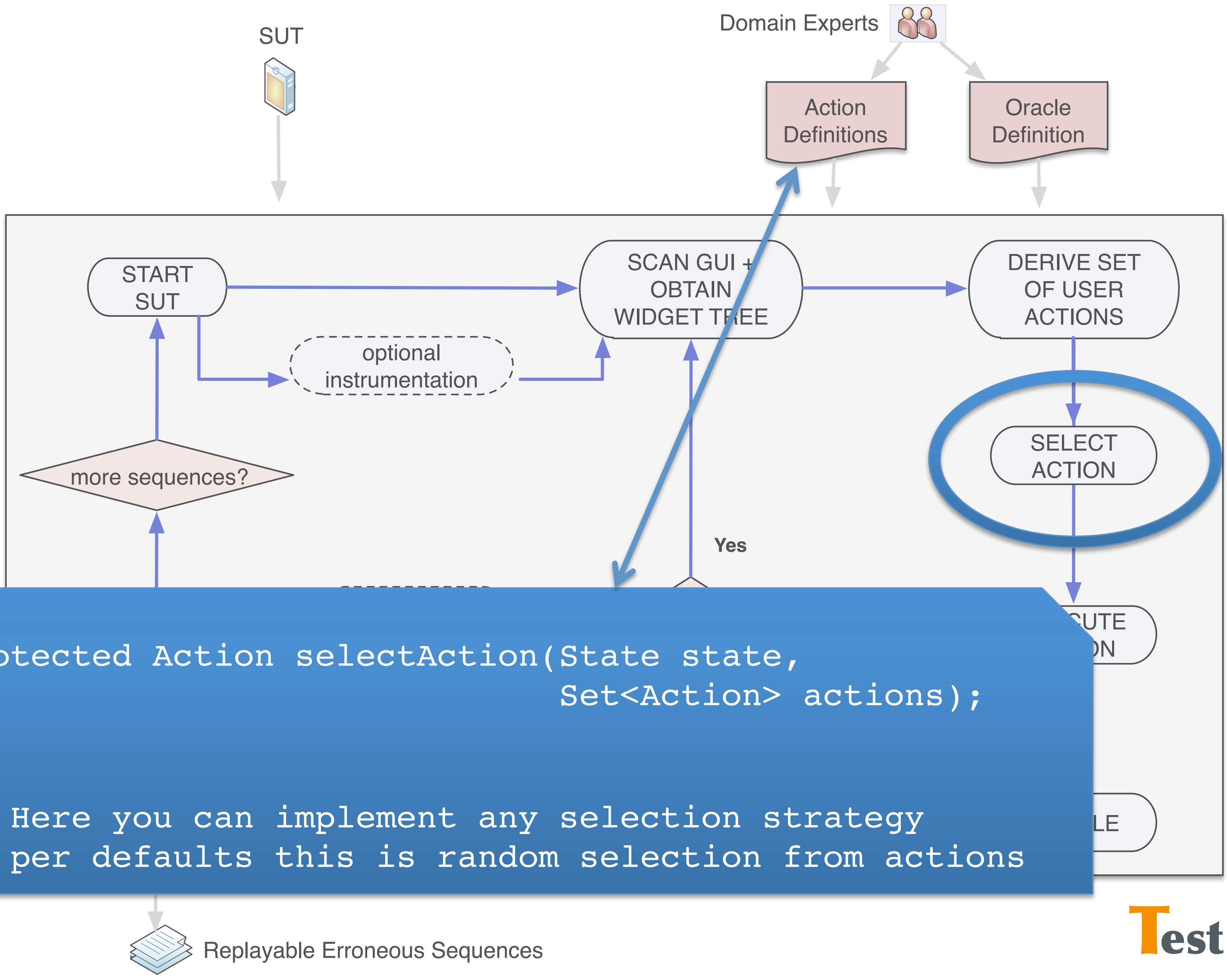
```
protected SUT startSystem()
              throws SystemStartException
```



```
protected State getState(SUT system)
                throws StateBuildException
```

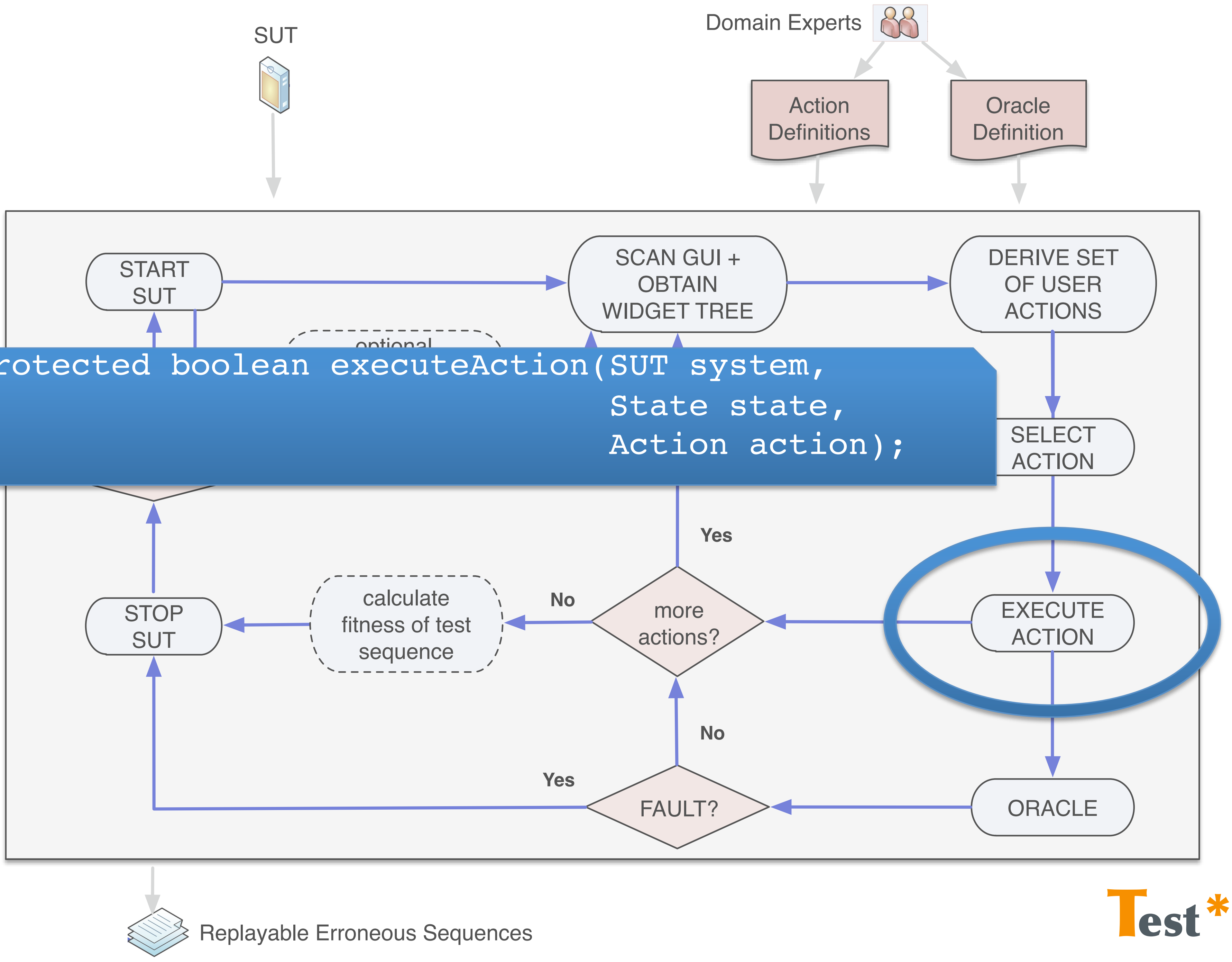


```
protected Set<Action> deriveActions(SUT system,
                                    State state)
                    throws ActionBuildException
```

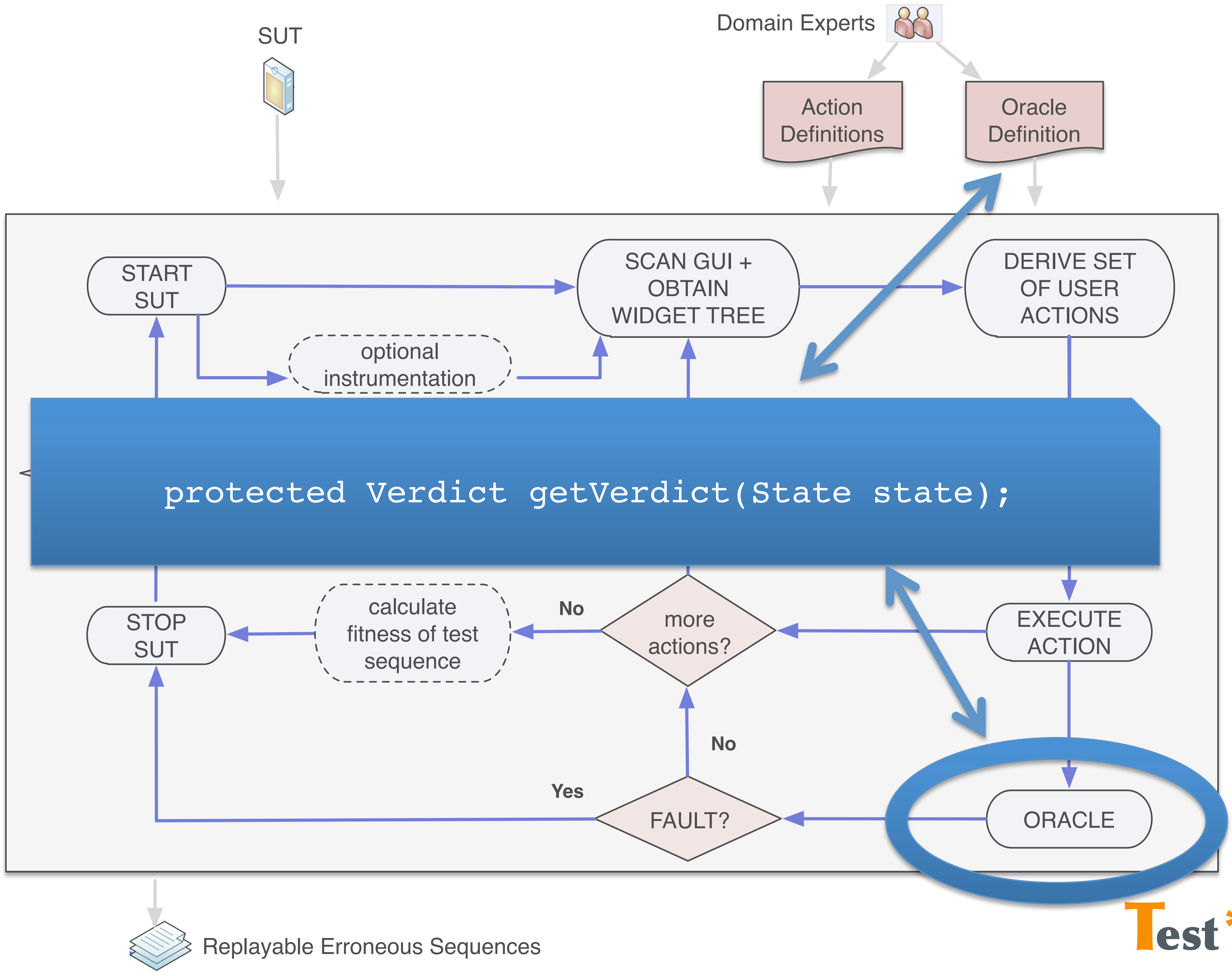


```
protected Action selectAction(State state,
                              Set<Action> actions);


// Here you can implement any selection strategy
// per defaults this is random selection from actions
```

Test*



```
protected boolean executeAction(SUT system,
                                State state,
                                Action action);
```



```
protected Verdict getVerdict(State state);
```

getVerdict

```
public final class Verdict {
        ….
        private final String info;
        private final double severity;
        private final Visualizer visualizer;

public Verdict (double severity,
                String info)
public Verdict(double severity,
               String info,
               Visualizer v)
```

```
protected Verdict getVerdict(State state){
        Assert.notNull(state);

        //-------------------
        // ORACLES FOR FREE
        //-------------------

        // if the SUT is not running, we assume it crashed

        if(!state.get(IsRunning, false))
                return new Verdict(1.0, "System is offline! I assume it crashed!");

        // if the SUT does not respond within a given amount of time, we assume it crashed

        if(state.get(NotResponding, false))
                return new Verdict(0.8, "System is unresponsive! I assume something is wrong!");
```

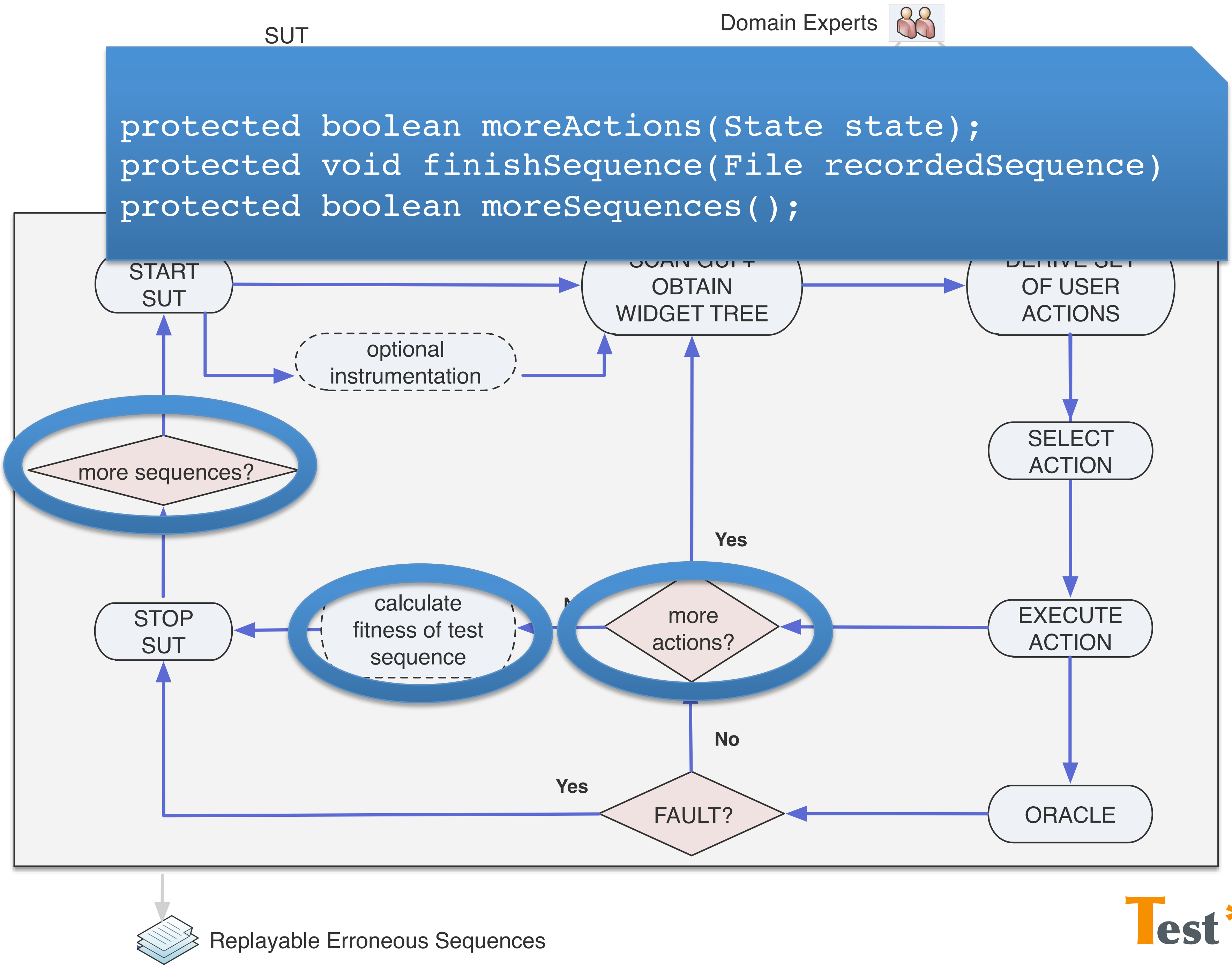
getVerdict

```
//------------------------
// ORACLES ALMOST FOR FREE
//------------------------

String titleRegEx = settings().get(SuspiciousTitles);

// search all widgets for suspicious titles
for(Widget w : state){
    String title = w.get(Title, "");
    if(title.matches(titleRegEx)){

        // visualize the problematic widget, by marking it with a red box
        Visualizer visualizer = Util.NullVisualizer;
        if(w.get(Tags.Shape, null) != null){
            Pen redPen = Pen.newPen().setColor(Color.Red).(…).build();
            visualizer = new ShapeVisualizer(redPen, ….., "Suspicious Title", 0.5, 0.5);
        }
        return new Verdict(1.0, "Discovered suspicious widget title: '" + title + "'.", visualizer);
    }
}
```

getVerdict

```
//--------------------------------------------------------------------------------
// MORE SOPHISTICATED ORACLES CAN BE PROGRAMMED HERE
//--------------------------------------------------------------------------------

The sky is the limit ;-)


// if everything was ok...
return new Verdict(0.0, "No problem detected.", Util.NullVisualizer);;

}
```

```
protected boolean moreActions(State state);
protected void finishSequence(File recordedSequence)
protected boolean moreSequences();
```



How has it been used?

MS Office

- Subject application: Microsoft Word 2011
- Robustness test: random action selection
- 18 hour run
- 672 sequences à 200 actions
- 9 crashes
- 6 reproducable crashes
- Effort was approx 1 hour to:
  - System setup (location, configuration files)
  - Augment Action Set (Drag Sources, Drop Targets, Clicks, Double Clicks, Right Clicks, Text to type, …)
  - Configure cheap oracle (crashes, timeouts, evident error messages)

CTE XL Profesional

berner & mattner
optimizing your development

- CTE XL Professional is a commercial tool for test case design
- Draw a combinatorial tree modeling test relevant aspects
- Generate a set of abstract test cases
- Java application - Eclipse Rich Client Platform (RCP) using Standard Widget Toolkit (SWT)
- Developed and commercialized by Berner&Mattner

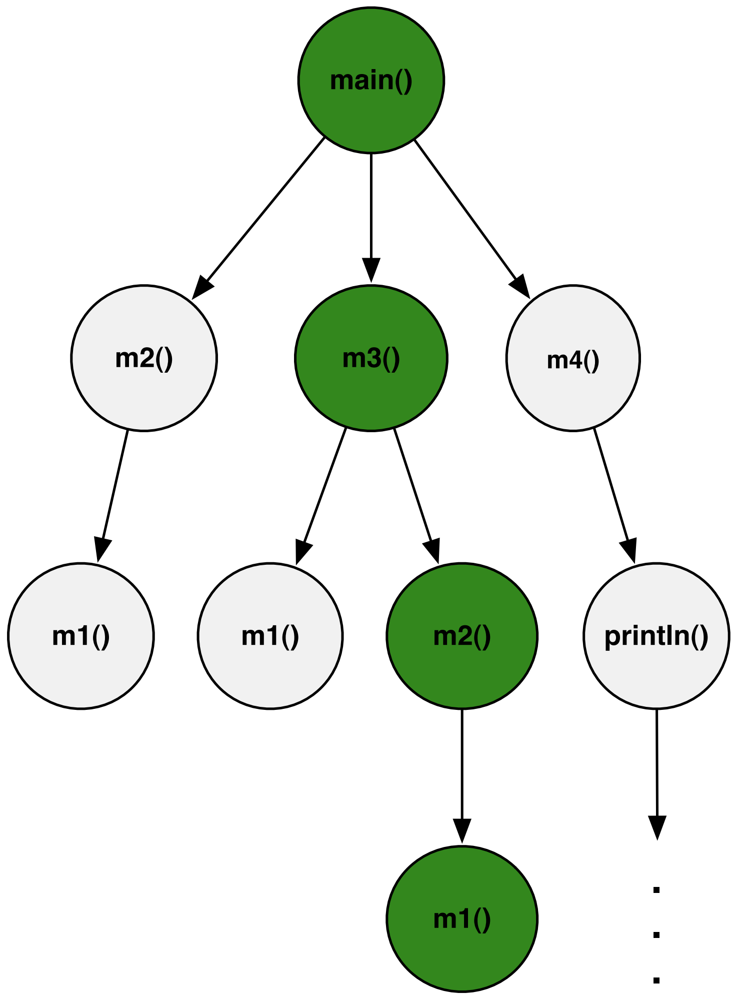
- TESTAR was used to test it.

Do experiments with more sophisticated action selection

- What is a “good” test sequence?

➔ One that generates lots of Maximum Call Stacks (MCS)

- MCS: root-leaf-path through call tree
- Intuition: the more MCSs a sequence generates, the more aspects of the SUT are tested (McMaster et al.)
- #MCS = number of leaves
- Obtainable through bytecode instrumentation (no source code needed)



Do experiments with more sophisticated action selection

- Select actions in such a way that sequences are formed that generate large amounts of “Maximum Call Stacks” within the system under test (SUT)

- Optimization algorithm used:
  - Ant Colony Optimization

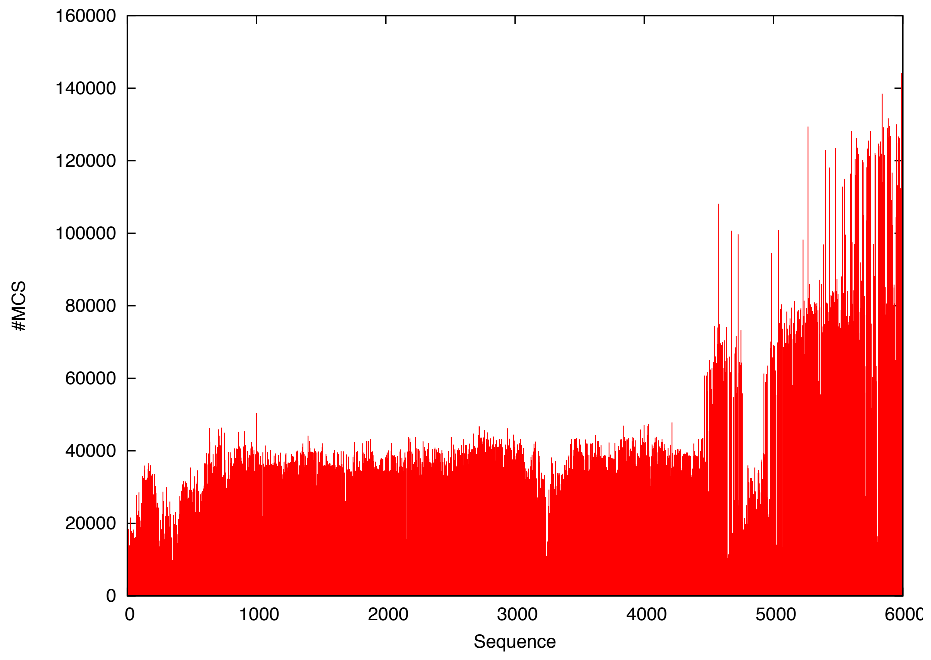
Ant Colony Optimization

- C = component set (here: C = set of feasible actions)
- The likelihood that $c_i \in C$ is chosen is determined by its pheromone value p_{ci}
- Generate trails (sequences) by selecting components according to pheromone values p_i
- Assess fitness of trails (i.e. MSC)
- Reward components c_i that appear in “good” trails by increasing their pheromones p_i

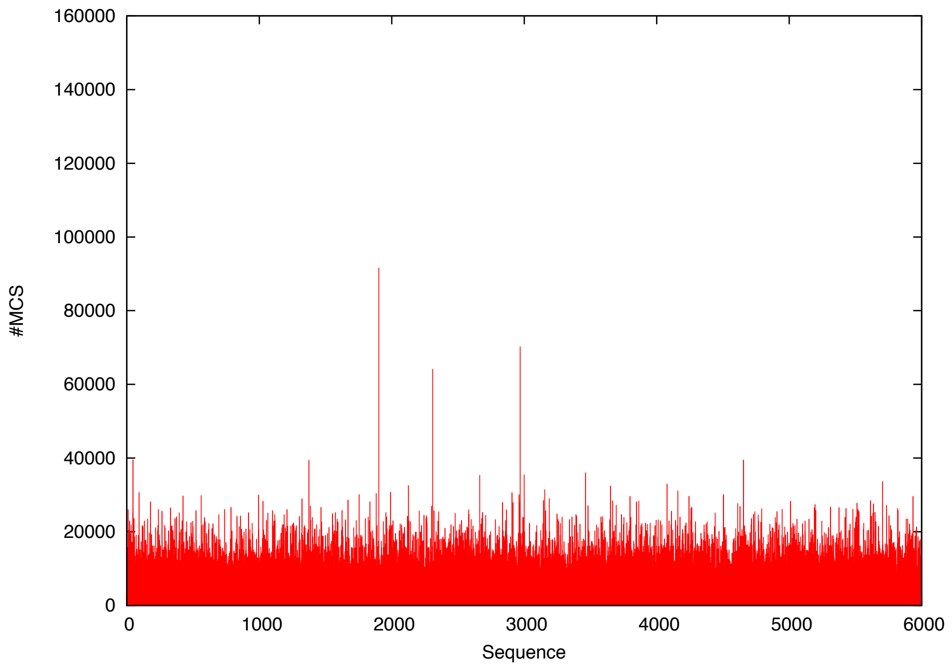
(Upon construction of subsequent trails, prefer components with high pheromone values)

Initial experiment results

ACO Run



Random Run



- Fixed stopping criteria -> 6000 generated sequences

Conclusion

- Implementation works
  - Better than random
  - Solutions improve over time
  - Letting it run unti
- Efficiency
  - Sequence generation is expensive ➔ parallelization
  - Frequent restarts of the SUT ➔ might not be suitable for large applications with a significant startup time, e.g. Eclipse
  - ACO good choice?
- Fault sensitivity? ➔ Empirical evaluation needed

Clave Informática

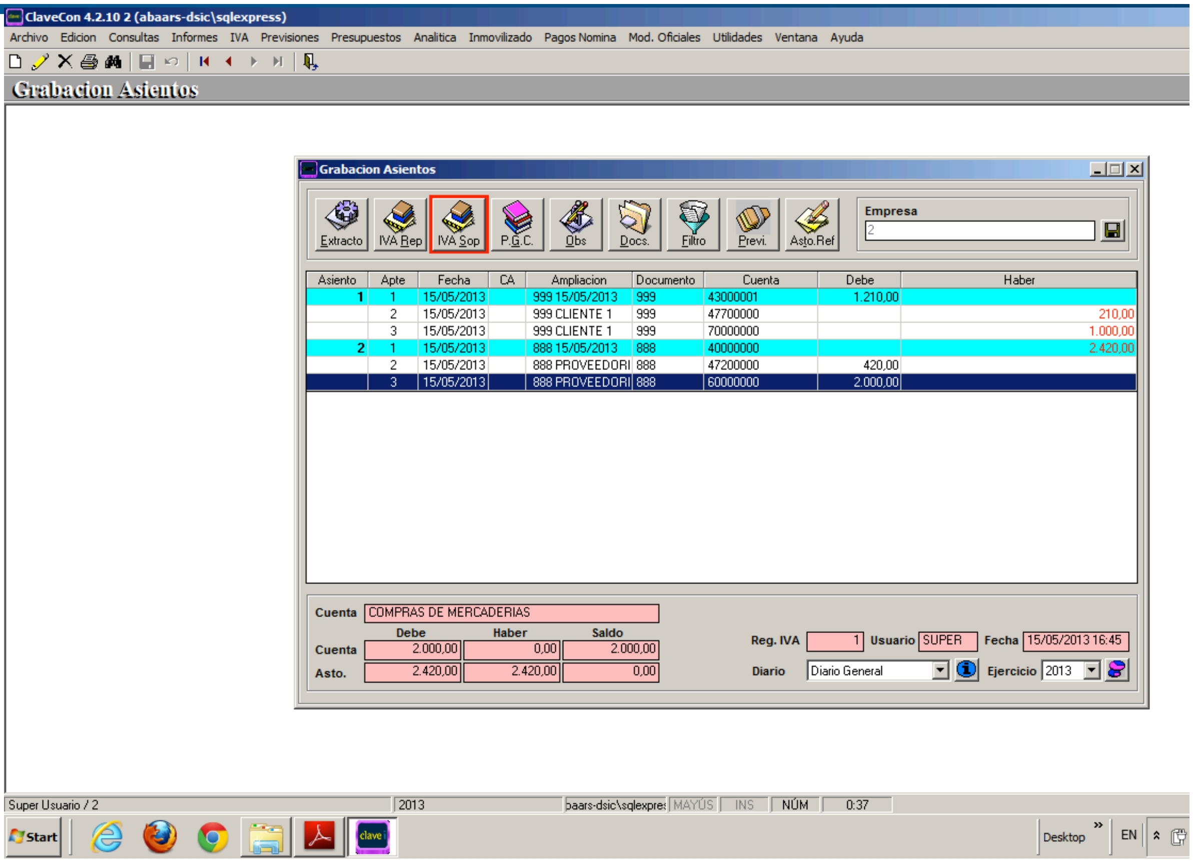
- We met this company at some local test event in Valencia
- Clavei is a private software vendor from Alicante, which
- Specialized for over 26 years in the development Enterprise Resource Planning (ERP) systems for SMEs.
- Main products is ClaveiCon a software solution for SMEs for accounting and financing control
- Current testing is done manually
- Amount of faults found by clients is too high
- Testing needs to be improved

Objectives of the study

- Can our tool be useful for Clave Informatica?
- Can it help them be more effective in finding faults?
- Can this be done in an efficient way, i.e. not taking too much time.
- Restrictions:
  - Clave had no budget to apply the tool themselves
  - So we, the tool developing researchers did that

ClaveiCon

- Written in Visual Basic
- Microsoft SQL Server 2008 database
- Targets the Windows operating systems.


ClaveCon 4.2.10 2 (abaars-dsic\sqlexpress)
Archivo Edicion Consultas Informes IVA Previsiones Presupuestos Analitica Inmovilizado Pagos Nomina Mod. Oficiales Utilidades Ventana Ayuda
Grabacion Asientos
Extracto
IVA Rep
IVA Sop
P.G.C.
Obs
Docs.
Filtro
Previ.
Asto.Ref
Start
Desktop
EN

Empresa: 2

Asiento	Apte	Fecha	CA	Ampliacion	Documento	Cuenta	Debe	Haber
1	1	15/05/2013		999 15/05/2013	999	43000001	1.210,00	
	2	15/05/2013		999 CLIENTE 1	999	47700000		210,00
	3	15/05/2013		999 CLIENTE 1	999	70000000		1.000,00
2	1	15/05/2013		888 15/05/2013	888	40000000		2.420,00
	2	15/05/2013		888 PROVEEDORI	888	47200000	420,00	
	3	15/05/2013		888 PROVEEDORI	888	60000000	2.000,00	

Cuenta: COMPRAS DE MERCADERIAS

	Debe	Haber	Saldo
Cuenta	2.000,00	0,00	2.000,00
Asto.	2.420,00	2.420,00	0,00

Reg. IVA: 1 Usuario: SUPER Fecha: 15/05/2013 16:45
Diario: Diario General Ejercicio: 2013

Super Usuario / 2 | 2013 | baars-dsic\sqlexpres | MAYÚS | INS | NÚM | 0:37

- Store data about product planning, cost, development and manufacturing.
- Provides a realtime view on a company's processes and enables controlling inventory management, shipping and payment as well as marketing and sales.

Case Study Procedure

1) Planning Phase:
   a) Implementation of Test Environment
   b) Error Definition: Anticipate and identify potential fault patterns.

2) Implementation Phase:
   a) Oracle Implementation: Implement the detection of the errors defined in the previous step.
   b) Action Definition Implementation
   c) Implementation of stopping criteria

3) Testing Phase: run the test

4) Evaluation Phase:
   a) Identify the most severe problems encountered during the run.
   b) The collected information will be used for the refinement of the setup during the next iteration.

Results

- The pre-testing activities:
  - the development or actions, oracles and stopping criteria to setup TESAR

  takes some initial effort (in our case approximately **26 hours**) but will pay off the more often the test is run.

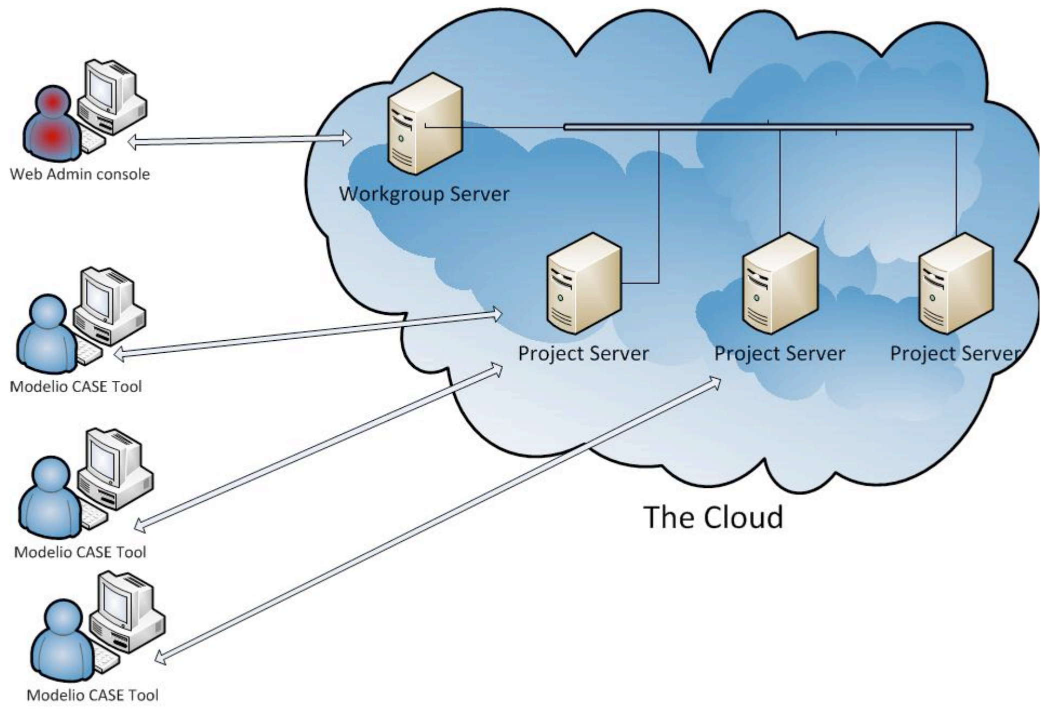
- The manual labor associated to post-testing:
  - inspection of log files,
  - reproduction and comprehension of errors

  Are only a tiny fraction of the overall testing time (we spent **1,5 hour** of manual intervention during and after tests, compared to over **91 hours** of actual unattended testing).

- TESTAR detected **10 previously unknown critical faults**, makes for a surprisingly positive result towards believing that TESTAR can be a valuable and resource-efficient supplement for manual testing.

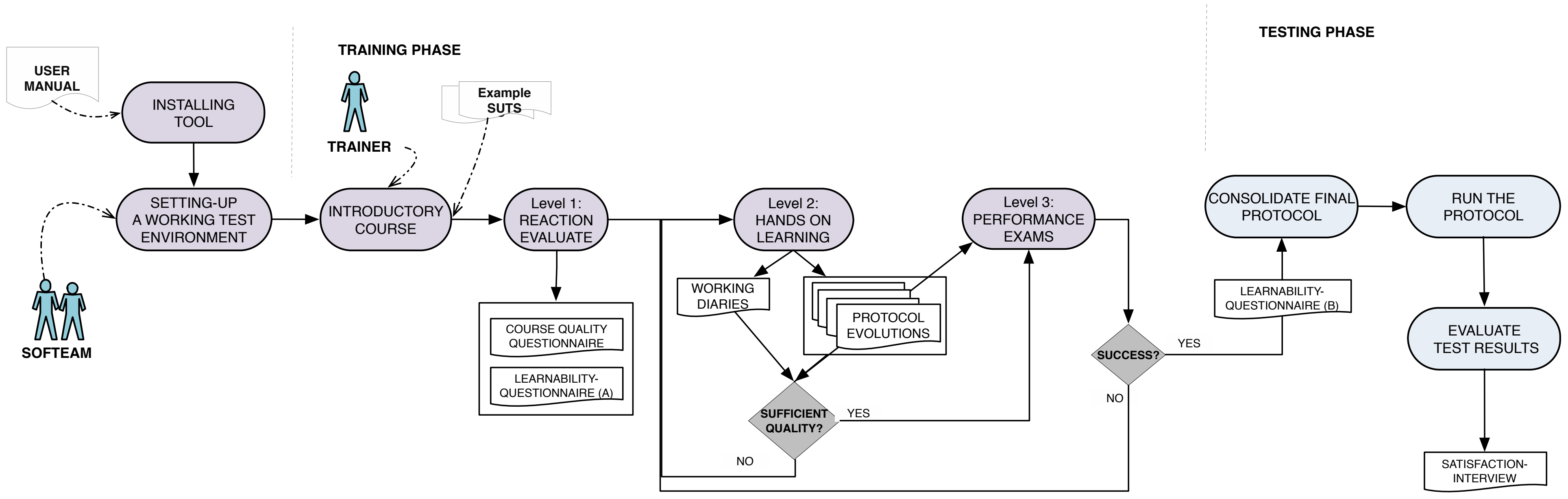
See a video here:
http://www.pros.upv.es/index.php/es/videos/item/1398-testar-rogue-user

Softeam



- FITTEST partner from France
- Big software company
- SUT selected for evaluating TESTAR: Modelio SaaS
- Modelio SaaS:
  - PHP web application
  - For the transparent configuration of distributed environments that run projects created with SOFTEAM's Modelio Modeling tool
  - Administrators use this application to manage servers and projects that run in virtual environments on different cloud platforms
- Current testing done manually

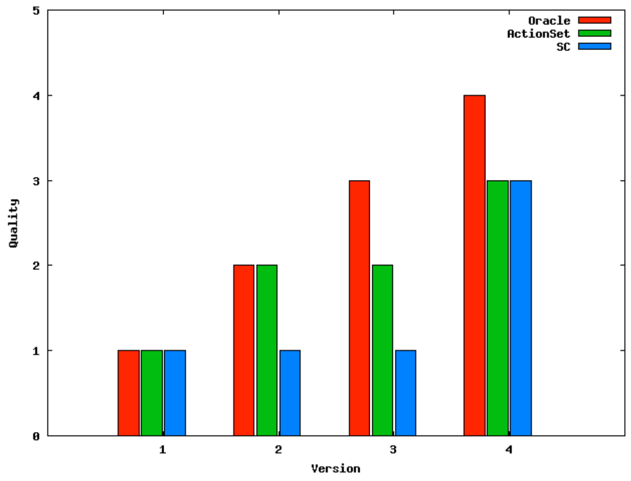
Case Study Procedure


TRAINING PHASE
TESTING PHASE
USER MANUAL
INSTALLING TOOL
TRAINER
Example SUTS
SETTING-UP A WORKING TEST ENVIRONMENT
INTRODUCTORY COURSE
Level 1: REACTION EVALUATE
Level 2: HANDS ON LEARNING
Level 3: PERFORMANCE EXAMS
CONSOLIDATE FINAL PROTOCOL
RUN THE PROTOCOL
SOFTEAM
COURSE QUALITY QUESTIONNAIRE
LEARNABILITY-QUESTIONNAIRE (A)
WORKING DIARIES
PROTOCOL EVOLUTIONS
SUFFICIENT QUALITY?
YES
NO
SUCCESS?
YES
NO
LEARNABILITY-QUESTIONNAIRE (B)
EVALUATE TEST RESULTS
SATISFACTION-INTERVIEW

We measured:

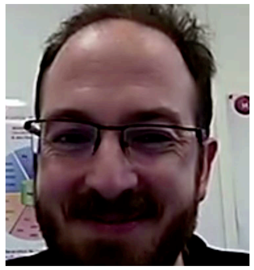
- Learnability (questionnaires, work-diaries, performance evaluations)
- Effectiveness
  - 17 faults were re-injected to evaluate
  - Code coverage
- Efficiency
  - Time for set-up, designing and develop
  - Time for running tests

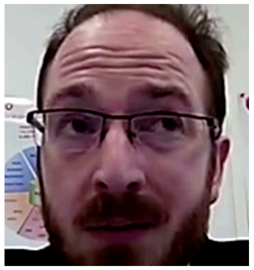
Results



Description	Test Suite	
	$\mathbf{TS}_{Soft}$	$\mathbf{TS}_{Testar}$
Faults discovered	14 + 1	10 + 1
Did not find IDs	1, 9, 12	1,4,8,12,14,15,16
Code coverage	86.63%	70.02%
Time spent on development	40h	36h
Run time	manual	automated
	1h 10m	77h 26m
Faults diagnosis and report	2h	3h 30m
Faults reproducible	100%	91.76%
Number of test cases	51	dynamic

- Some difficulties/resistance/ misunderstanding during the learning of programming for powerful oracles
- Testing artifacts produced increased in quality
  - Red = Oracle
  - Green = Action Set
  - Blue = Stopping Criteria


Would you recommend the tool to your colleagues?


Could you pursuade your management to invest?

Student course

- Course: 1st year Master
- “Developing Quality Software”
- 34 students working in groups of 2
- Introduction: 10 minutes
- Going through the user manual (10 pages) while doing a small exercise on a calculator: 50 minutes
- After 1 hour the students were setting up tests for MS paint

Future Work

- Still lot that needs to be done!
- Accessibility API works if UI has been programmed “well”
- Research more search-based approaches for action selection
- Research the integration of other test case generation techniques (model-based, combinatorial-based) for action selection
- Design a test spec language that makes it possible to specify actions and oracles without programming Java
- Do more industrial evaluations to compare maintenance costs during regression testing with our tool and capture/replay or visual testing tools
- Extend the tool beyond PC applications (for now we have Mac and Windows plug-ins) to mobile platforms

- **Tanja E. J. Vos**
- **email**: tvos@pros.upv.es
- **skype**: tanja_vos
- **web**: http://tanvopol.webs.upv.es/
- **telephone**: +34 690 917 971