



Saxonia Systems

Wir lieben IT.

michel.loehr@saxsys.de www.saxsys.de

Michel Löhr

Highspeed Test Automation



Dresden · Frankfurt/Main · Leipzig · München · Hamburg · Görlitz · Berlin



Highspeed Test Automation

Domain Specific Languages (DSLs)

- Dedicated languages for a specific purpose like SQL for handling data in databases
- Macros such as SUM() in Spreadsheets (EXCEL)
- Raised abstraction level and (re)using domain knowledge
- Opposite of general purpose languages as Java or C#
- Internal / External DSLs



Highspeed Test Automation

Goal of presentation

- Using a Domain Specific Language approach with automatic testing will increase productivity and quality in creating and maintaining testscripts
- I will showcase this with two examples





Highspeed Test Automation

Software Testing History

- | | |
|-------------|----------------------------|
| → Bis 1956 | - Debugging oriented |
| → 1957-1978 | - Demonstration oriented |
| → 1979-1982 | - Destruction oriented |
| → 1983-1987 | - Evaluation oriented |
| → 1988-2000 | - Prevention oriented (*) |
| → 2000- | - Automation oriented (**) |



A study conducted by NIST in 2002 reports that software bugs cost the U.S. economy \$59.5 billion annually. More than a third of this cost could be avoided if better software testing was performed.

Quellen:

(*) wikipedia

(**) M. Löhr



Highspeed Test Automation

Rationale

In software development practice, as much as 50% of the total cost is spent on testing to ensure its quality. To reduce such cost, it is imperative to have a solution to automate the testing process.

Automating testing within a software development project shouldn't be itself a software development project within its original project

→ ROI



"So, its fair to say you're very confident that you'll hit these testing milestones?"



Highspeed Test Automation

Test automation types

- White box testing
 - Code-driven testing
- Black box testing
 - (G)UI testing



Platform and OS independence
Data driven capability (Input Data, Output Data, Meta Data)
Customizable Reporting (DB Access, crystal reports, etc..)
Email Notifications (Automated notification on failure or threshold levels)
Easy debugging and logging
Version control friendly – minimum or zero binary files
Extensible & Customizable (Open APIs to be able to integrate with other tools)
Common Driver (Ant or Maven)
Headless execution for unattended runs (For integration with build process or batch runs)
Support distributed execution environment (distributed test bed)
Distributed application support (distributed SUT)
...





Highspeed Test Automation

Testautomation Approaches

- Data-driven testing
- Modularity-driven testing
- Keyword-driven testing
- Hybrid testing
- Model-based testing





Highspeed Test Automation

Keyword-driven testing

Object	Action	Data
Textfield(username)	Enter Text	„Max“
Textfield(password)	Enter Text	„Secret“
Button(login)	Click	One left click



Highspeed Test Automatisation

Keyword-driven testing

http://localhost:81/JavaWorld?test

Sun Java 2 Platform SE ...

FitNesse

ACTIONFIXTURE TO CALCULATE CHILD ALLOWANCE/MONTH

fit.ActionFixture		
start	stephanwiesner.javaworld.ChildAllowanceFixture	
press	personButton	
enter	securityNumber	1234567
press	modifyButton	
press	addChildButton	
enter	firstName	Lisa
enter	lastName	Schmith
enter	birthDay	01.01.1989
press	finishButton	
press	calculateChildAllowanceButton	
check	childAllowance	170 <i>expected</i>
		190 <i>actual</i>



Highspeed Test Automation

Domain Specific Languages (DSLs)

Editing Domain specific language: [/MyRules/HR-rules/hr-lang.dsl]

Description:

Language Expression	Rule language mapping	Scope
There exists a Person with name of {name}	Person(name=="{name}")	when
Person is at least {age} years old and lives in {location}	Person(age > {age}, locatio...	when
Log {message}	System.out.println("{messa...}	then
Send a message to {Person} with message {Message}	EmailUtil.sendEmail("{Person...}	then

Expression:

Mapping:



Highspeed DSL Example

Place an Order

Order
Alt

Item: Sub-Total: \$18.99
 Related Items: \$0.00

Quantity: S&H: \$1.00

Total: \$19.99

Card Number (include the spaces):

Card Type: Expiration Date:

Name:

Street:

City, State, Zip:

Phone:



Example

- 1.
- 2.
- 3.
- 4.
- 5.
- 6.
- 7.
- 8.
- 9.

```
21 public void testMain(Object[] args) throws IOException, BSFException
22 {
23     startApp("ClassicsJavaB");
24
25     // Frame: ClassicsCD
26     placeOrder().click();
27
28     // Frame: Member Logon
29     nameCombo().click(atText("Bill Wu"));
30     passwordText().click(atPoint(6,9));
31     memberLogon().inputChars("password");
32     rememberThePassword().clickToState(SELECTED);
33     ok().click();
34
35     // Frame: Place an Order
36     cardNumberIncludeTheSpacesText().click(atPoint(18,5));
37     placeAnOrder().inputKeys("1234 1234 1234 1234(TAB)");
38     creditCombo().click();
39     creditCombo().click(atText("Mastercard"));
40     expirationDateText().click(atPoint(9,10));
41     placeAnOrder().inputChars("09/09");
42     placeOrder2().click();
43
44     //
45     yourOrderHasBeenReceivedYourOrderNumberIs54().performTest(YourOrderHasBeenReceived_textVP());
46     ok2().click();
47
48     // Frame: ClassicsCD
49     jmb().click(atPath("Order"));
50     jmb().click(atPath("Order->View Existing Order Status..."));
51
52     // Frame: View Order Status
53     nameComboB().click();
54     nameComboB().click(atText("Bill Wu"));
55     passwordText2().click(atPoint(27,8));
56     viewOrderStatus().inputChars("password");
57     rememberPassword().clickToState(SELECTED);
58     ok3().click();
59
60     // Frame: View Existing Orders
61     existingTable().performTest(existingTable_contentsVP());
62     existingTable().click(atCell(atRow("ORDER ID", "11", "ORDER DATE",
63                                     "25/08/08", "STATUS",
64                                     "Order Initiated"),
65                               atColumn("ORDER ID"),
66                               atPoint(47,11)));
67     cancelSelectedOrder().click();
68     close().click();
69 }
```

```
34 1234",
.
al)).
```



How does this Testing DSL looks like?

```
clerk.starts_app.
places_order.for_customer(bill)
```

```
5 class Main < Window
6
7 def places_order
8   place_order.click
9   Logon.new
10 end
```

```
7 def for_customer(customer)
8   new = ! name.has?(customer[:name])
9   remember("customer", customer)
10
11   if new
12     new_customer.select
13   else
14     name.select(customer[:name])
15   end
16
17   ok.click
18   order = Order.new({:actor => actor})
19   order.sets(customer) if new
20   order
21 end
```

Metaprogramming

```

Logon
  cancel()
  existing_customer()
  name()
  new_customer()
  ok()
  password()
  region()
  remember_password()

```

```

: logon_win: javax.swing.JFrame
  : logon_cancel: javax.swing.JButton
  : logon_existing_customer: javax.swing.JRadioButton
  : logon_name: javax.swing.JComboBox
  : logon_new_customer: javax.swing.JRadioButton
  : logon_ok: javax.swing.JButton
  : logon_password: javax.swing.JPasswordField
  : logon_region: javax.swing.JComboBox
  : logon_remember_password: javax.swing.JCheckBox

```






How does such an architecture look like?

 Test case Scripter

Test case scripts `place_and_cancel_order` **Ruby DSL**

 Testing DSL Developer

Testing DSL

- StoreClerk**
 - starts_app()
 - closes_app()
 - calculate_total()
- Main**
 - places_order()
 - selects_album(album)
 - verifies_existing_orders()
- Logon**
 - for_customer(customer)
- Order**
 - confirms_order()
 - expects_incomplete_order()
 - expects_complete_order()

Ruby

GUI Test Support Framework

- Button**
- CheckBox**
- ComboBox**
 - initialize(test_obj)
 - create_by(selector, ancestor = root)
 - select(text)
 - each()
 - has?(to_find)
 - selected()
 - val()
 - val=(value)
- Window**
 - initialize(*arg)
 - generate_guiobj_methods(prefix)
 - actor()
 - expects(values)
 - expects_table_row(sym_table, sym_header, values)
 - remember(name, value)
 - sets(values)


Ruby Java

GUI Test Technology

- Swing/SWT
- HTML
- .Net
- Etc..

GUI Objects Map

- `order_cancel`: javax.swing.JButton
- `order_card_number`: javax.swing.JTextField
- `order_card_type`: javax.swing.JComboBox

 **IBM Rational Functional Tester**
Java - API



Highspeed Test Automation

DSL Example: Inventory Management

The screenshot displays the COBRA software interface. The main window is titled "Auftrag (Hanau)" and shows a table of orders. The table has columns for Auftragsnummer, Auftragsart, Status, AufbereitungszKZ, artikelreinzKZ, Erzeugungsdatum, Auslieferdatum, and Fakturierdatum. The row for order number 68 is highlighted. To the right, a "Datenabfrage" panel shows a query configuration with dropdown menus for Auftragsnum... (set to "gefüllt") and Auftragsstatus (set to "="). A dropdown menu is open, showing options: Abholbereit, Abrufbereit, Ausgeliefert, Einlagerung komplett, Neu, and Storno. The status "Neu" is selected. The bottom status bar shows "NGS: Stammdaten;" and page numbers "1 200 200".

Auftragsnummer	Auftragsart	Status	AufbereitungszKZ	artikelreinzKZ	Erzeugungsdatum	Auslieferdatum	Fakturierdatum
27	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	08.01.2010 02:13:38	05.12.2008 11:00:00	
55	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	16.11.2011 01:35:20	16.12.2008 11:00:00	
68	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	18.11.2011 09:35:29	18.12.2008 11:00:00	
56111324	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111330	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111358	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111412	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111458	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111459	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111501	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111503	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111515	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111517	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111523	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111526	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111527	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111533	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111540	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111610	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111617	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111624	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56111634	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56150958	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		
56151030	B2B-Webshop	Neu	keine Aufbereitung	artikelrein	06.10.2009 02:22:30		



Highspeed Test Automation

Test automation using Groovy & internal DSL

```
AuslagerungsAuftragErzeugen(lager) {
    super("Auslagerungsauftrag erzeugen ." + lager + ".")
    set_fixtures (
        auslagerungstyp      : {comboBox(name: "baseCom
        auftragszuordnung    : {comboBox(name: "baseCom
        auftrag               : {textBox(name:"jTextFiel
        vorgang              : {textBox(leftlabel: "Vo
        fachentnahmen        : {checkBox(name: "jCheckB
        auslagerungskriterien : {tabbedPane()},
        auslagerbare_menge    : {textBox(name:"jTextFiel
        artikel_kriterien     : {table("jTableItemData")
        ...
    )
}
```

The screenshot shows the COBRA application interface. The title bar indicates the application is running in a test environment: "COBRA [Test - 14022 (DEVELO1_TESTPRODUCTIONNEW.OV.OTTO.DE)] Version 10.0 Build 170 Benutzer: mloehr". The main window is titled "Auslagerungsauftrag erzeugen [Lager Hanau] Ticket: 240710287".

The interface is divided into several sections:

- Auslagerungsvorgang bestimmen:** This section contains several input fields and dropdown menus. "Auslagerungstyp" is set to "362 AUS-NGS-AUS-ZLAWd (NGS Abgabe Web-Shop dez)". "Vorgang" is "153626868". "Auftrag" is empty, and "Auftrag zuordnen" is set to "0 - keine Vorgabe". "Priorität" is "0". There is a "Notizen" text area. Checkboxes for "Komplette Trolleys" (unchecked) and "Entnahmen aus Containern zulassen" (checked) are present. "Auslagerungsstrategie" is set to "70 FIFO (HAN)". "Sammeletiketten drucken" is unchecked.
- Auslagerungskriterien bestimmen:** This section features a table with columns: "Artikel", "Erweiterte", "Anliefe...", "Lagerort", "ClientNo", "Firma", "Lieferant", "Artikel", "Größe", "Saison", "FT", "OT", "gesperrt", "Sperrtyp", "Menge", "Trolley", and "Artikels...". The "gesperrt" column contains the text "nur freie".
- Auslagerungsvorschau:** This section contains several summary fields: "Betroffene Gebäude", "Betroffene Etagen", "Gefundene KS-Plätze", "Vollständige Trolley", "Gefundene RL-Plätze", "Entnahme Kartons", "Gefundene Bügel", "Entnahme Retouren", "Gefundene Bügel auf Trolleys", and "Auszulagernde Menge".



Highspeed Test-Automation

DSL Example: Groovy Testscript snippets

```
def artikeln = [[artikel:6199, grÖÙe:38, menge: 12, trolli: [10, 2]],
                [artikel:6198, grÖÙe:38, menge: 12, trolli: [10, 2]]
                ]
vorgang = erzeugeAuslagerung([auslagerungstyp : "362 .*",
                              auftrag         : auftragNo,
                              auftragszuordnung : "1 - Auftrag zuordnen",
                              notizen         : "Guitest: Hanau -> Webshop",
                              sammeletiketten : false
                              ],
                              [artikeln],
                              WLS
                              )

cobra.auslagerungsauftrag_erzeugen().with {
    erzeuge(auslagerung)
    artikeln.each { artikel ->
        artikel_kriterien([ [Artikel: artikel.artikel,
                            GrÖÙe: artikel.grÖÙe,
                            Menge: artikel.menge]
                            ])
    }
}

def artikel_kriterien(List arg) {
    auslagerungskriterien.selectTab("Artikel")
    artikel_kriterien.addRow(arg, 0, {it.pressAndReleaseKey(key_enter)})
}
```



Highspeed Test Automation

Conclusion using DSLs

Benefits

- Higher abstraction > better productivity in testscripting
- Better flexibility over Keyword-driven testing (graybox)

Drawbacks

- Need some development experience of (host language) to create testscripts
- Technology dependencies





Highspeed Test Automatisierung

Vision: data-, keyword-, model driven testing

Testplan: sample # 1

```
login user: „foo“, password „bar“  
buy_cds datapool[„cd-pool1“]
```

Testplan: sample # 2

```
login user: „foo“, password „bar“ using advanced
```

Scenario: login user, passwd

```
login_screen {  
    enter username:user, password:passwd  
    [advanced]  
    click login  
}  
alternative: advanced {  
    check advanced_login  
}
```

Context: login_screen

```
fixture: {title: „Login .*“}  
guimap: [username, password: textbox,  
    advanced: checkbox,  
    login: button]
```



Saxonia Systems

Wir lieben IT.

michel.loehr@saxsys.de www.saxsys.de

Thank you for your attention!





```
static String takeScreenshot(String name = "") {  
  
    def screenSize = java.awt.Toolkit.getDefaultToolkit().getScreenSize()  
    def rect = new java.awt.Rectangle(0, 0, screenSize.width.toInteger(), screenSize.height.toInteger())  
    def image = new java.awt.Robot().createScreenCapture(rect)  
  
    if (!log.dir.exists()) {  
        log.dir.mkdir()  
    }  
  
    //Save the screenshot as a png  
    def imgname = (name)?name:"screen${++log.scrn_count}"  
    def imgfile = new File("${log.dir}/${imgname}.png")  
    ImageIO.write(image, "png", imgfile)  
  
    def htmlfile = new FileWriter("${log.dir}/${imgname}.html")  
    def x = new groovy.xml.MarkupBuilder(htmlfile)  
    x.html{  
        body{  
            img(src:"${imgname}.png")  
        }  
    }  
    imgname  
}
```



```
def set_fields(Map args) {
  args.each { key, value ->
    def comp = fixtures[key.toLowerCase]()
    switch (comp) {
      case JCheckBoxFixture:
        if (Eval.me("$value") && !comp.isSelected()) {
          comp.check()
        } else {
          if (!Eval.me("$value") && comp.isSelected()) {
            comp.uncheck()
          }
        }
        break
      case JComboBoxFixture:
        comp.select(value.toString())
        break
      case JTextComponentFixture:
        comp.setText(value.toString())
        break
    }
  }
}
```



```
def methodMissing(String name, args) {
  if (name.startsWith("fenster_")) {
    def action = name.replaceFirst("fenster_", "")
    switch (action) {
      case ~/bewege.*/ : name = "moveTo"
                        args = new Point(args[0], args[1]) ; break
      case ~/breite.*/ : name = "resizeWidthTo" ; break
      case "groesse"   :
      case ~/größe.*/  : name = "resizeTo"
                        args = new Dimension(args[0], args[1]) ; break
      case "herstellen" : name = "normalize" ; break
      case ~/hoehe.*/  :
      case ~/höhe.*/   : name = "resizeHeightTo" ; break
      case "maximieren" : name = "maximize" ; break
      case "minimieren" : name = "iconify" ; break
      case "schliessen" :
      case "schließen"  : name = "close" ; break
      default: fail("unknown action: "+action)
    }
  }
  fixture.invokeMethod(name, args)
  this
}
```

```
Closure addRows = { delegate, rows, col_offset, afterRow, customEditors ->
```

```
  rows.each { row ->
```

```
    int lastrow = delegate.rowCount() - 1
```

```
    String description = ""
```

```
    // when key '_order' is present take this to be the order to set the cells:
```

```
    def order = (row["_order"]) ? : row.keySet()
```

```
    order.each { entry ->
```

```
      def value = row[entry]
```

```
      description += "$entry=$value; "
```

```
      def col_entry = -1
```

```
      delegate.component().getColumnModel().getColumns().eachWithIndex { header, i ->
```

```
        if (header.headerValues.grep(entry)) {
```

```
          col_entry = i
```

```
        }
```

```
      }
```

```
    if (col_entry < 1) {
```

```
      throw new Exception("Column $e
```

```
    }
```

```
    def cell = new TableCell(lastrow,
```

```
    def cf = new JTableCellFixture(delegate,
```

```
    def editor = cf.editor()
```

```
    def done = false
```

```
    customEditors.each { type, edit ->
```

```
      if (editor.class.name =~ ".$*$
```

```
        edit(delegate, cell, value
```

```
        done = true
```

```
      }
```

```
    }
```

```
    if (!done) {
```

```
      delegate.enterValue(cell, value
```

```
    }
```

```
  }
  afterRow.call(delegate)
```

```
}
```

```
JTableFixture.metaClass.addRows = {rows -> addRows(delegate, rows, 0, {}, [:]) }
JTableFixture.metaClass.addRows = {rows, col_offset, afterRow ->
```

```
  addRows(delegate, rows, col_offset, afterRow, [:]) }
```

```
JTableFixture.metaClass.addRows = {rows, col_offset, afterRow, customEditors ->
addRows(delegate, rows, col_offset, afterRow, customEditors) }
```

```
Closure tabLastCol = { delegate ->
```

```
  def row = delegate.component().getSelectedRow()
```

```
  def col = delegate.component().getColumnCount()-1
```

```
  def cell = new TableCell(row, col)
```

```
  delegate.selectCell(cell)
```

```
  delegate.pressAndReleaseKey(key_tab)
```

```
}
```

```
Closure comboBoxEditor = { delegate, cell, value ->
```

```
  delegate.selectCell(cell)
```

```
  // be sure to get only the codenumber
```

```
  def code = value.toString() =~ /(\d+)/
```

```
  // press every num key of the code:
```

```
  code[0][1].getChars().each { key ->
```

```
    delegate.pressAndReleaseKey(keyCode((key as int)))
```

```
  }
```

```
}
```




```
static def telnet() {  
  
telnet = new Telr  
telnet.connect(hc  
rin = telnet.getI  
rout = telnet.get  
  
setupGUI("Scanner  
  
server = new XMLF  
server.isReady =  
  def isReady = r  
  ready = false  
  return isReady  
}  
  
socket = new Serv  
server.startServe  
telnet()  
  
  def writer = Thread.start {  
    BufferedReader br = new BufferedReader(new  
      InputStreamReader(telnet.getInputStream(), "ISO-8859-1"))  
  
    char[] buff = new byte[1024]  
    int ret_read  
    while ( (ret_read = br.read(buff,0,1023)) )  
    {  
      if (ret_read > 0) {  
        def output = new String(buff, 0, ret_read)  
  
        def parts = output.split(/[\\x1b]\\x5b\\x32\\x4a\\x1b\\x5b\\x48/)  
        parts.each { part ->  
          if (part.size() > 2) {  
            text.text = ""  
            println "-----EOS-----"  
          }  
          text.append(part)  
          println "${part}"  
        }  
        // mark output as ready  
        ready = true  
      }  
    }  
  }  
}  
  
remote = new XMLRPCServerProxy("http://localhost:8008/")  
while (!remote.isReady()) {  
  sleep(200)  
}
```