ETALIQ
intelligent quality

Etaliq @ RTAC09

innovation in automation

innovation in automation

www.etaliq.com

ETALIQ
intelligent quality

Agenda

ETALIQ
intelligent quality

Why Automation?

innovation in automation

innovation in automation

www.etaliq.com

ETALIQ
intelligent quality

Why Automation?

Everyone needs to...

Automation must...

ETALIQ
intelligent quality

Why Automation?

Everyone needs to...

- Save time

Automation must...

ETALIQ
intelligent quality

Why Automation?

Everyone needs to...

- Save time
- Increase quality

Automation must...

ETALIQ
intelligent quality

Why Automation?

Everyone needs to...

- Save time
- Increase quality
- Find *bugs* sooner

Automation must...

ETALIQ
intelligent quality

Why Automation?

Everyone needs to...

- Save time
- Increase quality
- Find *bugs* sooner

Automation must...

- Reduce effort

ETALIQ
intelligent quality

Why Automation?

Everyone needs to...

- Save time
- Increase quality
- Find *bugs* sooner

Automation must...

- Reduce effort
- Be *reliable*

ETALIQ
intelligent quality

Why Automation?

Everyone needs to...

- Save time
- Increase quality
- Find *bugs* sooner

Automation must...

- Reduce effort
- Be *reliable*
- Be *fast*

ETALIQ
intelligent quality

Why Automation?

Everyone needs to...

- Save time
- Increase quality
- Find *bugs* sooner

Automation must...

- Reduce effort
- Be *reliable*
- Be *fast*
- Be *easy*

ETALIQ
intelligent quality

Automation Today

innovation in automation

innovation in automation

innovation in automation

www.etaliq.com

ETALIQ
intelligent quality

Automation Environment

Manual DevTest Engineer

- Test strategies/plans
- On-demand scripting

Automation Engineer

- Translate test case to code
- Execute and review
- First-line problem determination

Manual DevTest

- Subject Matter Expert (SME)
- Defines how to test
- Defines what to test
- Creates *personal*...
  - automation as required
  - Tcl regular expressions

ETALIQ
intelligent quality

Automation Environment

Manual DevTest Engineer

- Test strategies/plans
- On-demand scripting

Automation Engineer

- Translate test case to code
- Execute and review
- First-line problem determination

Automation Engineer

- *Not* SME
- Translates *verbage* test plan to running code
- Creates log output
- Reviews log output
- Determines if...
  - *environmental* error or
  - *real* problem

ETALIQ
intelligent quality

Automation Environment Infrastructure

Many files:

- libraries, scripts, job files,
- node definitions, map files,
- log files, environment settings

Many systems:

- code libraries, device control,
- TGEN libraries, analysis,
- logging, reporting, revision control,
- reservation, scheduling, ...

Expertise required

- Unix, Tcl, regular expressions,
- basic DUT behavior,
- many file formats,
- TGEN functions,
- libraries,
- log interpretation

ETALIQ
intelligent quality

Etaliq – Easy Test Automation

innovation in automation

innovation in automation

www.etaliq.com

ETALIQ
intelligent quality

ETA – Primary ROI Methods

Features:

- Test plan & case scripting
  - 25 instructions, coding time
- Resource usage
  - Syntax check
  - Simulated nodes
  - Hierarchy: Groups that FAIL don't run tests
- Log review & results reporting
  - Fully integrated indexed logs
  - Customizable summarized results reporting

ETALIQ
intelligent quality

ETA – Automation Environment Infrastructure

Files:

- node table,
- execution table,
- test plan

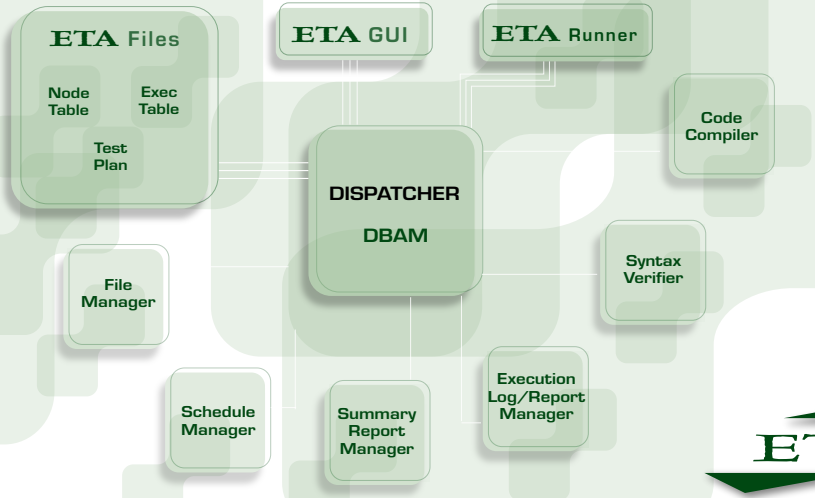
Single system:

- Execution Engine, TGEN types,
- log reports, summarized reporting,
- File Manager,
- Scheduler

Expertise required

- basic DUT behavior,
- minimal file formats,
- TGEN functions

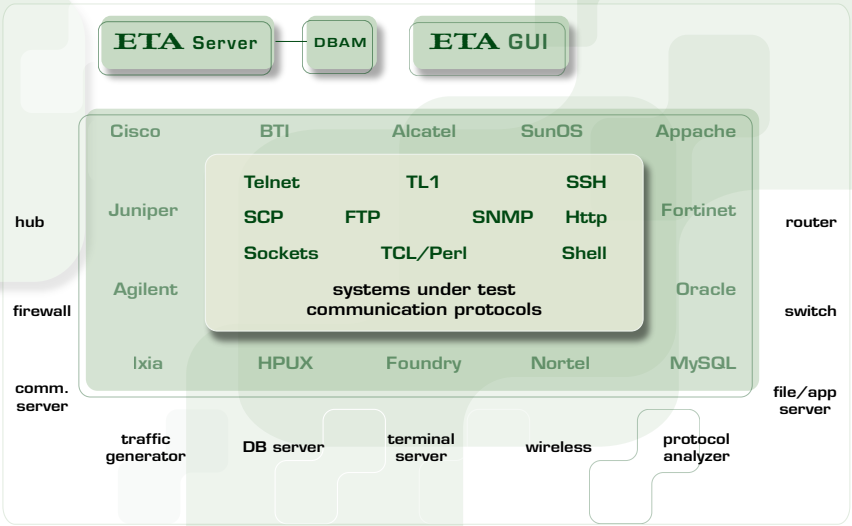
|system achitecture|



ETA 2.1

|target environment|

ETA 2.1



ETALIQ

intelligent quality

ETA vs Tcl

innovation in automation

innovation in automation

www.etaliq.com

Send, Receive, and Verify!

- The ETA test plan template includes all related documentation.
- The **SEND** verb allows sending commands to nodes and handles all communications.
- The **RESULTLIST** verb defines expected results, no need for complicated parsing!
- **RESULTLIST** variables are associated with **SEND** instances and automate verification.
- Values are easily extracted from node output simply by referencing the **SEND** verb, no need for complicated parsing!
- The **EXPR** sub-verb provides powerful access to Boolean expressions and arithmetics, including seamless IP expressions.

ETA

```
Test Case Identifier: etaVsTcl_1
Title: ETA vs Tcl Comparison
Keywords: demo send receive verify
Objectives:
    Compare the ease and power of ETA to traditional Tcl code.
Common:
    RESULTLIST(CheckIntfUp)
        "line protocol = up"
        "input errors = 0"
        "AFTER 5 minute input rate  ATTR packets/sec > 0"
    RESULTLIST(CheckPingSuccess)
        "Success rate = 100 percent"
Setup:
Steps:
    # Show and check the local interface and statistics
    SEND RouterNode1 "show int Serial0" dispIntf CheckIntfUp

    # Derive remote IP based on local IP found in "dispIntf"
    # (p2p, ^.3 does .1 <-> .2)
    SET SrcIPnet dispIntf(Internet address)
    SET DestIP   EXPR(ipHost(SrcIPnet ^ 0.0.0.3))

    # Ping the remote interface
    SEND RouterNode1 "ping DestIP" CheckPingSuccess
Cleanup:
```

ETA

```
Test Case Identifier: etaVsTcl_1
Title: ETA vs Tcl Comparison
Keywords: demo send receive verify
Objectives:
    Compare the ease and power of ETA to traditional Tcl code.
Common:
    RESULTLIST(CheckIntfUp)
        "line protocol = up"
        "input errors = 0"
        "AFTER 5 minute input rate  ATTR packets/sec > 0"
    RESULTLIST(CheckPingSuccess)
        "Success rate = 100 percent"
Setup:
Steps:
    # Show and check the local interface and statistics
    SEND RouterNode1 "show int Serial0" dispIntf CheckIntfUp

    # Derive remote IP based on local IP found in "dispIntf"
    # (p2p, ^.3 does .1 <-> .2)
    SET SrcIPnet dispIntf(Internet address)
    SET DestIP   EXPR(ipHost(SrcIPnet ^ 0.0.0.3))

    # Ping the remote interface
    SEND RouterNode1 "ping DestIP" CheckPingSuccess
Cleanup:
```

Tcl (1/3)

```
set RouterNode1 csMdNode1
set intf $::nodes_array($RouterNode1,INTF1)

if { ![node_connect $RouterNode1] } {
    error "Cannot continue: Connection to node $RouterNode1 failed!"
}

set cmd "show int $intf"
if { ![node_send_receive $RouterNode1 $cmd\r "# *$" dispIntf] } {
    error "Cannot continue: Failed to send '$cmd' to node $RouterNode1!"
}

if { ![regexp "line protocol is (\[^ \]+)" $dispIntf - valLineProto] } {
    log_failure "line protocol value not found in '$cmd' output!"
    set ::test_result Fail
    set valLineProto "unknown"
} elseif { $valLineProto != "up" } {
    log_failure "line protocol is '$valLineProto' but expected to be 'up'!"
    set ::test_result Fail
} else {
    log_info "line protocol is '$valLineProto', as expected."
}
...
```

ETA

```
Test Case Identifier: etaVsTcl_1
Title: ETA vs Tcl Comparison
Keywords: demo send receive verify
Objectives:
    Compare the ease and power of ETA to traditional Tcl code.
Common:
    RESULTLIST(CheckIntfUp)
        "line protocol = up"
        "input errors = 0"
        "AFTER 5 minute input rate  ATTR packets/sec > 0"
    RESULTLIST(CheckPingSuccess)
        "Success rate = 100 percent"
Setup:
Steps:
    # Show and check the local interface and statistics
    SEND RouterNode1 "show int Serial0" dispIntf CheckIntfUp

    # Derive remote IP based on local IP found in "dispIntf"
    # (p2p, ^.3 does .1 <-> .2)
    SET SrcIPnet dispIntf(Internet address)
    SET DestIP   EXPR(ipHost(SrcIPnet ^ 0.0.0.3))

    # Ping the remote interface
    SEND RouterNode1 "ping DestIP" CheckPingSuccess
Cleanup:
```

Tcl (1/3)

```
set RouterNode1 csMdNode1
set intf $::nodes_array($RouterNode1,INTF1)

if { ![node_connect $RouterNode1] } {
    error "Cannot continue: Connection to node $RouterNode1 failed!"
}

set cmd "show int $intf"
if { ![node_send_receive $RouterNode1 $cmd\r "# *$" dispIntf] } {
    error "Cannot continue: Failed to send '$cmd' to node $RouterNode1!"
}

if { ![regexp "line protocol is (\[^ \]+)" $dispIntf - valLineProto] } {
    log_failure "line protocol value not found in '$cmd' output!"
    set ::test_result Fail
    set valLineProto "unknown"
} elseif { $valLineProto != "up" } {
    log_failure "line protocol is '$valLineProto' but expected to be 'up'!"
    set ::test_result Fail
} else {
    log_info "line protocol is '$valLineProto', as expected."
}
...
```

ETA

```
Test Case Identifier: etaVsTcl_1
Title: ETA vs Tcl Comparison
Keywords: demo send receive verify
Objectives:
    Compare the ease and power of ETA to traditional Tcl code.
Common:
    RESULTLIST(CheckIntfUp)
        "line protocol = up"
        "input errors = 0"
        "AFTER 5 minute input rate  ATTR packets/sec > 0"
    RESULTLIST(CheckPingSuccess)
        "Success rate = 100 percent"
Setup:
Steps:
    # Show and check the local interface and statistics
    SEND RouterNode1 "show int Serial0" dispIntf CheckIntfUp

    # Derive remote IP based on local IP found in "dispIntf"
    # (p2p, ^.3 does .1 <-> .2)
    SET SrcIPnet dispIntf(Internet address)
    SET DestIP   EXPR(ipHost(SrcIPnet ^ 0.0.0.3))

    # Ping the remote interface
    SEND RouterNode1 "ping DestIP" CheckPingSuccess
Cleanup:
```

Tcl (1/3)

```
set RouterNode1 csMdNode1
set intf $::nodes_array($RouterNode1,INTF1)

if { ![node_connect $RouterNode1] } {
    error "Cannot continue: Connection to node $RouterNode1 failed!"
}

set cmd "show int $intf"
if { ![node_send_receive $RouterNode1 $cmd\r "# *$" dispIntf] } {
    error "Cannot continue: Failed to send '$cmd' to node $RouterNode1!"
}

if { ![regexp "line protocol is (\[^ \]+)" $dispIntf - valLineProto] } {
    log_failure "line protocol value not found in '$cmd' output!"
    set ::test_result Fail
    set valLineProto "unknown"
} elseif { $valLineProto != "up" } {
    log_failure "line protocol is '$valLineProto' but expected to be 'up'!"
    set ::test_result Fail
} else {
    log_info "line protocol is '$valLineProto', as expected."
}
...
```

ETA

```
Test Case Identifier: etaVsTcl_1
Title: ETA vs Tcl Comparison
Keywords: demo send receive verify
Objectives:
    Compare the ease and power of ETA to traditional Tcl code.
Common:
    RESULTLIST(CheckIntfUp)
        "line protocol = up"
        "input errors = 0"
        "AFTER 5 minute input rate  ATTR packets/sec > 0"
    RESULTLIST(CheckPingSuccess)
        "Success rate = 100 percent"
Setup:
Steps:
    # Show and check the local interface and statistics
    SEND RouterNode1 "show int Serial0" dispIntf CheckIntfUp

    # Derive remote IP based on local IP found in "dispIntf"
    # (p2p, ^.3 does .1 <-> .2)
    SET SrcIPnet dispIntf(Internet address)
    SET DestIP   EXPR(ipHost(SrcIPnet ^ 0.0.0.3))

    # Ping the remote interface
    SEND RouterNode1 "ping DestIP" CheckPingSuccess
Cleanup:
```

Tcl (2/3)

```
...
if { ![regexp "(\[0-9\]+) input errors" $dispIntf - valInputErrors] } {
    log_failure "input errors value not found in '$cmd' output!"
    set ::test_result Fail
    set valInputErrors "unknown"
} elseif { $valInputErrors != 0 } {
    log_failure "input errors value is $valInputErrors but expected to be 0!"
    set ::test_result Fail
} else {
    log_info "input errors value is $valInputErrors, as expected."
}

if { ![regexp "5 minute input rate\[^\n\]*\
      (\[0-9\]+) packets/sec" $dispIntf - valInputRate] } {
    log_failure "5 minute input rate value not found in '$cmd' output!"
    set ::test_result Fail
    set valInputRate "unknown"
} elseif { $valInputRate <= 0 } {
    log_failure "5 minute input rate value is $valInputRate but expected\
        to greater than 0!"
    set ::test_result Fail
} else {
    log_info "5 minute input rate value is $valInputRate, as expected."
}
...
```

ETA

```
Test Case Identifier: etaVsTcl_1
Title: ETA vs Tcl Comparison
Keywords: demo send receive verify
Objectives:
    Compare the ease and power of ETA to traditional Tcl code.
Common:
    RESULTLIST(CheckIntfUp)
        "line protocol = up"
        "input errors = 0"
        "AFTER 5 minute input rate  ATTR packets/sec > 0"
    RESULTLIST(CheckPingSuccess)
        "Success rate = 100 percent"
Setup:
Steps:
    # Show and check the local interface and statistics
    SEND RouterNode1 "show int Serial0" dispIntf CheckIntfUp

    # Derive remote IP based on local IP found in "dispIntf"
    # (p2p, ^.3 does .1 <-> .2)
    SET SrcIPnet dispIntf(Internet address)
    SET DestIP   EXPR(ipHost(SrcIPnet ^ 0.0.0.3))

    # Ping the remote interface
    SEND RouterNode1 "ping DestIP" CheckPingSuccess
Cleanup:
```

Tcl (2/3)

```
...
if { ![regexp "(\[0-9\]+) input errors" $dispIntf - valInputErrors] } {
    log_failure "input errors value not found in '$cmd' output!"
    set ::test_result Fail
    set valInputErrors "unknown"
} elseif { $valInputErrors != 0 } {
    log_failure "input errors value is $valInputErrors but expected to be 0!"
    set ::test_result Fail
} else {
    log_info "input errors value is $valInputErrors, as expected."
}

if { ![regexp "5 minute input rate\[^\n\]*\
      (\[0-9\]+) packets/sec" $dispIntf - valInputRate] } {
    log_failure "5 minute input rate value not found in '$cmd' output!"
    set ::test_result Fail
    set valInputRate "unknown"
} elseif { $valInputRate <= 0 } {
    log_failure "5 minute input rate value is $valInputRate but expected\
        to greater than 0!"
    set ::test_result Fail
} else {
    log_info "5 minute input rate value is $valInputRate, as expected."
}
...
```

ETA

```
Test Case Identifier: etaVsTcl_1
Title: ETA vs Tcl Comparison
Keywords: demo send receive verify
Objectives:
    Compare the ease and power of ETA to traditional Tcl code.
Common:
    RESULTLIST(CheckIntfUp)
        "line protocol = up"
        "input errors = 0"
        "AFTER 5 minute input rate  ATTR packets/sec > 0"
    RESULTLIST(CheckPingSuccess)
        "Success rate = 100 percent"
Setup:
Steps:
    # Show and check the local interface and statistics
    SEND RouterNode1 "show int Serial0" dispIntf CheckIntfUp

    # Derive remote IP based on local IP found in "dispIntf"
    # (p2p, ^.3 does .1 <-> .2)
    SET SrcIPnet dispIntf(Internet address)
    SET DestIP   EXPR(ipHost(SrcIPnet ^ 0.0.0.3))

    # Ping the remote interface
    SEND RouterNode1 "ping DestIP" CheckPingSuccess
Cleanup:
```

Tcl (3/3)

```
...
if { ![regexp "Internet address is\
      (\[0-9\]+\.\[0-9\]+\.\[0-9\]+\.\[0-9\]+)" $dispIntf - valSrcIP] } {
    log_failure "Internet address value not found in '$cmd' output!"
    set ::test_result Fail
    set valSrcIP "unknown"
} else {
    set bytes [split $valSrcIP .]
    set bytes [lreplace $bytes 3 3 [expr { [lindex $bytes 3] ^ 3 }]]
    set valDestIP [join $bytes .]

    set cmd "ping $valDestIP"
    if { ![node_send_receive $RouterNode1 $cmd\r "# *$" dispPing] } {
        error "Cannot continue: Failed to send '$cmd' to node $RouterNode1!"
    }
    if { ![regexp "Success rate is 100 percent" $dispPing] } {
        log_failure "Ping failed!"
        set ::test_result Fail
    } else {
        log_info "Ping succeeded."
    }
}

log_info "test result: $::test_result"
```

ETA

```
Test Case Identifier: etaVsTcl_1
Title: ETA vs Tcl Comparison
Keywords: demo send receive verify
Objectives:
    Compare the ease and power of ETA to traditional Tcl code.
Common:
    RESULTLIST(CheckIntfUp)
        "line protocol = up"
        "input errors = 0"
        "AFTER 5 minute input rate  ATTR packets/sec > 0"
    RESULTLIST(CheckPingSuccess)
        "Success rate = 100 percent"
Setup:
Steps:
    # Show and check the local interface and statistics
    SEND RouterNode1 "show int Serial0" dispIntf CheckIntfUp

    # Derive remote IP based on local IP found in "dispIntf"
    # (p2p, ^.3 does .1 <-> .2)
    SET SrcIPnet dispIntf(Internet address)
    SET DestIP   EXPR(ipHost(SrcIPnet ^ 0.0.0.3))

    # Ping the remote interface
    SEND RouterNode1 "ping DestIP" CheckPingSuccess
Cleanup:
```

Tcl (3/3)

```
...
if { ![regexp "Internet address is\
     (\[0-9\]+\.\[0-9\]+\.\[0-9\]+\.\[0-9\]+)" $dispIntf - valSrcIP] } {
    log_failure "Internet address value not found in '$cmd' output!"
    set ::test_result Fail
    set valSrcIP "unknown"
} else {
    set bytes [split $valSrcIP .]
    set bytes [lreplace $bytes 3 3 [expr { [lindex $bytes 3] ^ 3 }]]
    set valDestIP [join $bytes .]

    set cmd "ping $valDestIP"
    if { ![node_send_receive $RouterNode1 $cmd\r "# *$" dispPing] } {
        error "Cannot continue: Failed to send '$cmd' to node $RouterNode1!"
    }
    if { ![regexp "Success rate is 100 percent" $dispPing] } {
        log_failure "Ping failed!"
        set ::test_result Fail
    } else {
        log_info "Ping succeeded."
    }
}

log_info "test result: $::test_result"
```

ETA

```
Test Case Identifier: etaVsTcl_1
Title: ETA vs Tcl Comparison
Keywords: demo send receive verify
Objectives:
    Compare the ease and power of ETA to traditional Tcl code.
Common:
    RESULTLIST(CheckIntfUp)
        "line protocol = up"
        "input errors = 0"
        "AFTER 5 minute input rate  ATTR packets/sec > 0"
    RESULTLIST(CheckPingSuccess)
        "Success rate = 100 percent"
Setup:
Steps:
    # Show and check the local interface and statistics
    SEND RouterNode1 "show int Serial0" dispIntf CheckIntfUp

    # Derive remote IP based on local IP found in "dispIntf"
    # (p2p, ^.3 does .1 <-> .2)
    SET SrcIPnet dispIntf(Internet address)
    SET DestIP   EXPR(ipHost(SrcIPnet ^ 0.0.0.3))

    # Ping the remote interface
    SEND RouterNode1 "ping DestIP" CheckPingSuccess
Cleanup:
```

Tcl (3/3)

```
...
if { ![regexp "Internet address is\
      (\[0-9\]+\.\[0-9\]+\.\[0-9\]+\.\[0-9\]+)" $dispIntf - valSrcIP] } {
    log_failure "Internet address value not found in '$cmd' output!"
    set ::test_result Fail
    set valSrcIP "unknown"
} else {
    set bytes [split $valSrcIP .]
    set bytes [lreplace $bytes 3 3 [expr { [lindex $bytes 3] ^ 3 }]]
    set valDestIP [join $bytes .]

    set cmd "ping $valDestIP"
    if { ![node_send_receive $RouterNode1 $cmd\r "# *$" dispPing] } {
        error "Cannot continue: Failed to send '$cmd' to node $RouterNode1!"
    }
    if { ![regexp "Success rate is 100 percent" $dispPing] } {
        log_failure "Ping failed!"
        set ::test_result Fail
    } else {
        log_info "Ping succeeded."
    }
}

log_info "test result: $::test_result"
```

ETA

```
Test Case Identifier: etaVsTcl_1
Title: ETA vs Tcl Comparison
Keywords: demo send receive verify
Objectives:
    Compare the ease and power of ETA to traditional Tcl code.
Common:
    RESULTLIST(CheckIntfUp)
        "line protocol = up"
        "input errors = 0"
        "AFTER 5 minute input rate  ATTR packets/sec > 0"
    RESULTLIST(CheckPingSuccess)
        "Success rate = 100 percent"
Setup:
Steps:
    # Show and check the local interface and statistics
    SEND RouterNode1 "show int Serial0" dispIntf CheckIntfUp

    # Derive remote IP based on local IP found in "dispIntf"
    # (p2p, ^.3 does .1 <-> .2)
    SET SrcIPnet dispIntf(Internet address)
    SET DestIP   EXPR(ipHost(SrcIPnet ^ 0.0.0.3))

    # Ping the remote interface
    SEND RouterNode1 "ping DestIP" CheckPingSuccess
Cleanup:
```

Tcl (3/3)

```
...
if { ![regexp "Internet address is\
      (\[0-9\]+\.\[0-9\]+\.\[0-9\]+\.\[0-9\]+)" $dispIntf - valSrcIP] } {
    log_failure "Internet address value not found in '$cmd' output!"
    set ::test_result Fail
    set valSrcIP "unknown"
} else {
    set bytes [split $valSrcIP .]
    set bytes [lreplace $bytes 3 3 [expr { [lindex $bytes 3] ^ 3 }]]
    set valDestIP [join $bytes .]

    set cmd "ping $valDestIP"
    if { ![node_send_receive $RouterNode1 $cmd\r "# *$" dispPing] } {
        error "Cannot continue: Failed to send '$cmd' to node $RouterNode1!"
    }
    if { ![regexp "Success rate is 100 percent" $dispPing] } {
        log_failure "Ping failed!"
        set ::test_result Fail
    } else {
        log_info "Ping succeeded."
    }
}

log_info "test result: $::test_result"
```

ETA

```
Test Case Identifier: etaVsTcl_1
Title: ETA vs Tcl Comparison
Keywords: demo send receive verify
Objectives:
    Compare the ease and power of ETA to traditional Tcl code.
Common:
    RESULTLIST(CheckIntfUp)
        "line protocol = up"
        "input errors = 0"
        "AFTER 5 minute input rate  ATTR packets/sec > 0"
    RESULTLIST(CheckPingSuccess)
        "Success rate = 100 percent"
Setup:
Steps:
    # Show and check the local interface and statistics
    SEND RouterNode1 "show int Serial0" dispIntf CheckIntfUp

    # Derive remote IP based on local IP found in "dispIntf"
    # (p2p, ^.3 does .1 <-> .2)
    SET SrcIPnet dispIntf(Internet address)
    SET DestIP   EXPR(ipHost(SrcIPnet ^ 0.0.0.3))

    # Ping the remote interface
    SEND RouterNode1 "ping DestIP" CheckPingSuccess
Cleanup:
```

6 verb commands!

Tcl

```
set RouterNode1 ckNSNode1
set intf $::nodes_array(RouterNode1,INTF1)

if { ![node_connect $RouterNode1] } {
    error "Cannot continue: Connection to node $RouterNode1 failed!"
}

set cmd "show int $intf"
if { ![node_send_receive $RouterNode1 $cmd\r "# "$" dispIntf] } {
    error "Cannot continue: Failed to send '$cmd' to node $RouterNode1!"
}

if { [regexp "line protocol is (\[^ \]+)" $dispIntf - valLineProto] } {
    log_failure "line protocol value not found in '$cmd' output!"
    set ::test_result Fail
    set valLineProto "unknown"
} elseif { $valLineProto != "up" } {
    log_failure "line protocol is '$valLineProto' but expected to be 'up'!"
    set ::test_result Fail
} else {
    log_info "line protocol is '$valLineProto', as expected."
}

if { [regexp "(\[0-9\]+) input errors" $dispIntf - valInputErrors] } {
    log_failure "input errors value not found in '$cmd' output!"
    set ::test_result Fail
    set valInputErrors "unknown"
} elseif { $valInputErrors != 0 } {
    log_failure "input errors value is $valInputErrors but expected to be 0!"
    set ::test_result Fail
} else {
    log_info "input errors value is $valInputErrors, as expected."
}

if { [regexp "5 minute input rate\[^\n\]*\
      (\[0-9\]+) packets/sec" $dispIntf - valInputRate] } {
    log_failure "5 minute input rate value not found in '$cmd' output!"
    set ::test_result Fail
    set valInputRate "unknown"
} elseif { $valInputRate == 0 } {
    log_failure "5 minute input rate value is $valInputRate but expected\
        to greater than 0!"
    set ::test_result Fail
} else {
    log_info "5 minute input rate value is $valInputRate, as expected."
}

if { [regexp "Internet address is\
      (\[0-9\]+\.\[0-9\]+\.\[0-9\]+\.\[0-9\]+)" $dispIntf - valSrcIP] } {
    log_failure "Internet address value not found in '$cmd' output!"
    set ::test_result Fail
    set valSrcIP "unknown"
} else {
    set bytes [split $valSrcIP .]
    set bytes [lreplace $bytes 3 3 [expr { [lindex $bytes 3] ^ 3 }]]
    set valDestIP [join $bytes .]

    set cmd "ping $valDestIP"
    if { ![node_send_receive $RouterNode1 $cmd\r "# "$" dispPing] } {
        error "Cannot continue: Failed to send '$cmd' to node $RouterNode1!"
    }
    if { ![regexp "Success rate is 100 percent" $dispPing] } {
        log_failure "Ping failed!"
        set ::test_result Fail
    } else {
        log_info "Ping succeeded."
    }
}

log_info "test result: $::test_result"
```

60 lines!

Send, Receive, and Verify!

Products

Routers Switches Hubs Gateways *Firewalls* *Servers*
Workstations *Databases* Analyzers

UUT Protocols

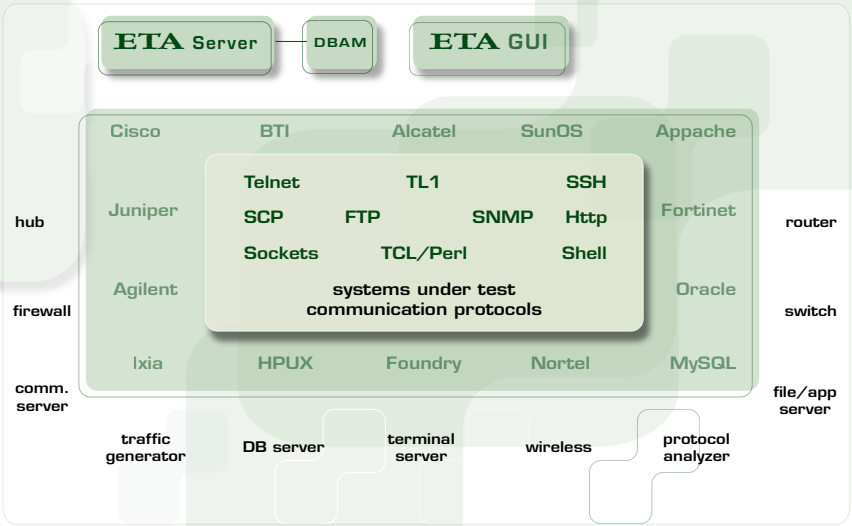
Telnet TL1 SSH SFTP FTP *SNMP* *XML* *Sockets* *SQL*
Tcl Perl Shell Files HTML *HTTP* *HTTPS*

Vendors

Cisco Juniper Nortel Alcatel Lucent F5 *IBM* Ciena MySQL
Apache Sun Linux *HP* Agilent Ixia Spirent *Microsoft*

|target environment|

ETA 2.1



ETALIQ
intelligent quality

Task #1: Execution

Learn to:

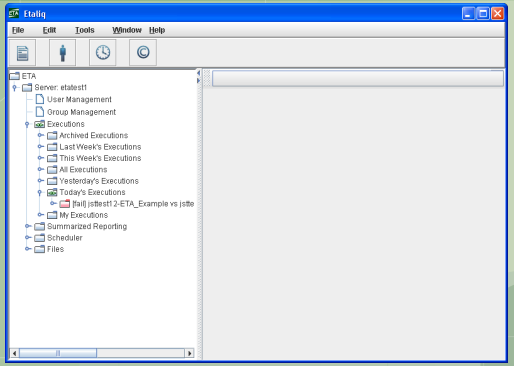
- schedule an execution
- review an execution's reports
- navigate using relative position jumping
- update the test case code

Register at `http://www.etaliq.com` to access the *Live* ETA *Demo*

ETALIQ
intelligent quality

Task #1: Execution

List of executions

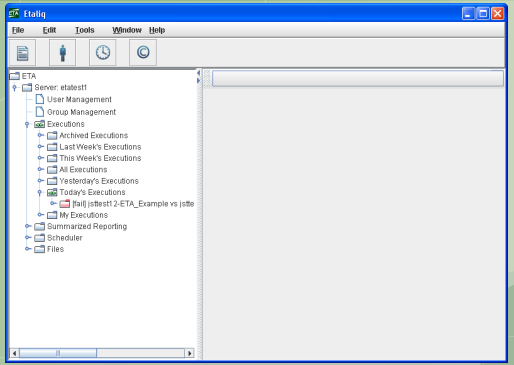


- Expand the *Executions* tree item to display execution filters

ETALIQ
intelligent quality

Task #1: Execution

List of executions

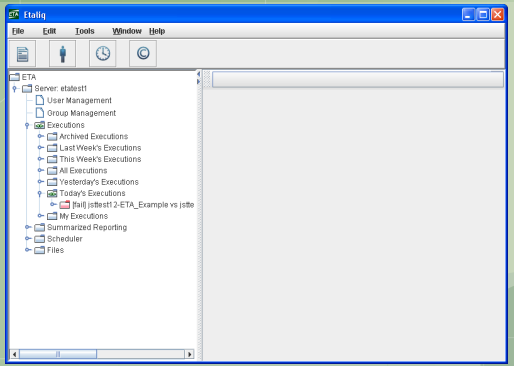


- Expand the *Executions* tree item to display execution filters
- Expand the *Today's Executions* execution filter

ETALIQ
intelligent quality

Task #1: Execution

List of executions

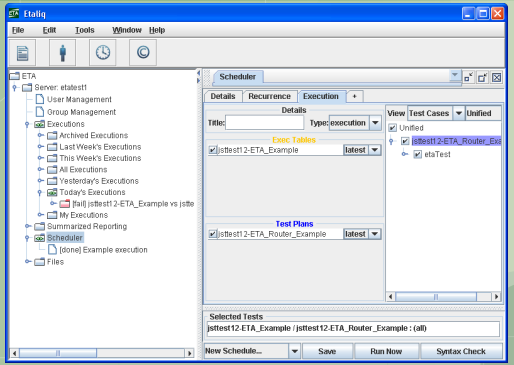


- Expand the *Executions* tree item to display execution filters
- Expand the *Today's Executions* execution filter
- Any executions just scheduled or executed today will appear under this tree item
- Several default execution filters are available
- Any number of custom execution filters can be created using the *Filter Management* tool

ETALIQ
intelligent quality

Task #1: Execution

Starting the Schedule

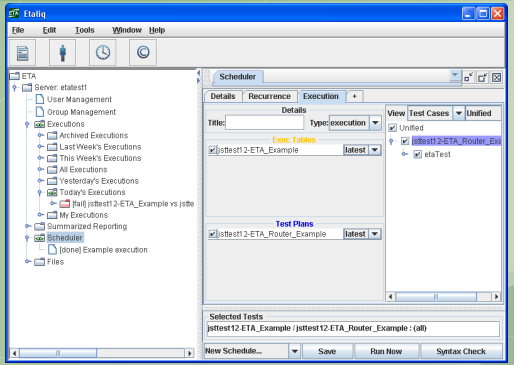


- Select the *Scheduler* link from the *Left Tree* to activate the *Scheduler* tool in the right pane

ETALIQ
intelligent quality

Task #1: Execution

Starting the Schedule

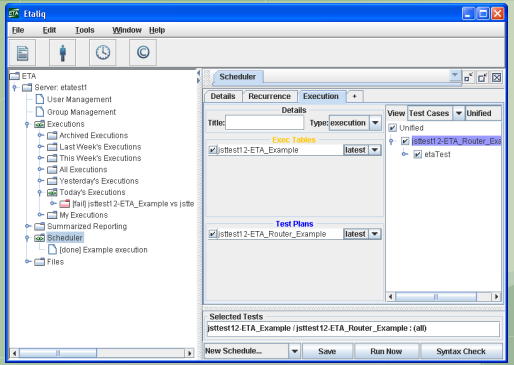


- Select the *Scheduler* link from the *Left Tree* to activate the *Scheduler* tool in the right pane
- Expand the *Scheduler* tree item to display existing schedules

ETALIQ
intelligent quality

Task #1: Execution

Starting the Schedule

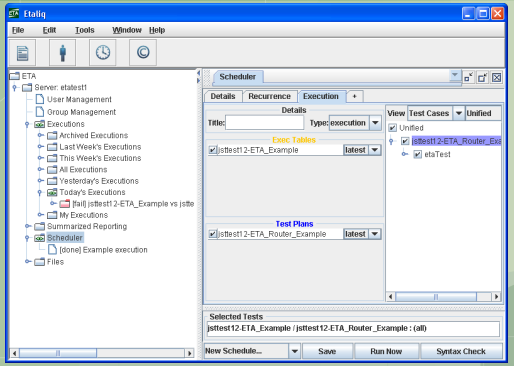


- Select the *Scheduler* link from the *Left Tree* to activate the *Scheduler* tool in the right pane
- Expand the *Scheduler* tree item to display existing schedules
- Select the *Scheduler*'s *Execution* tab

ETALIQ
intelligent quality

Task #1: Execution

Starting the Schedule

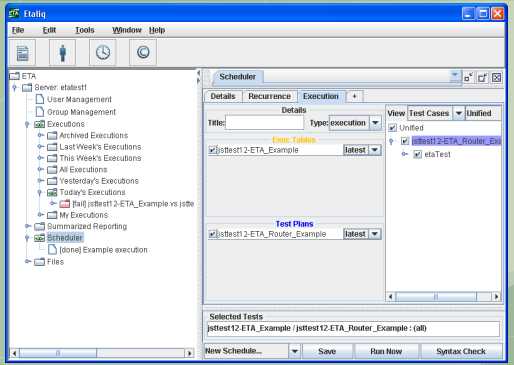


- Select the *Scheduler* link from the *Left Tree* to activate the *Scheduler* tool in the right pane
- Expand the *Scheduler* tree item to display existing schedules
- Select the *Scheduler*'s *Execution* tab
- Select the execution table named `user-ETA_Example`

ETALIQ
intelligent quality

Task #1: Execution

Starting the Schedule

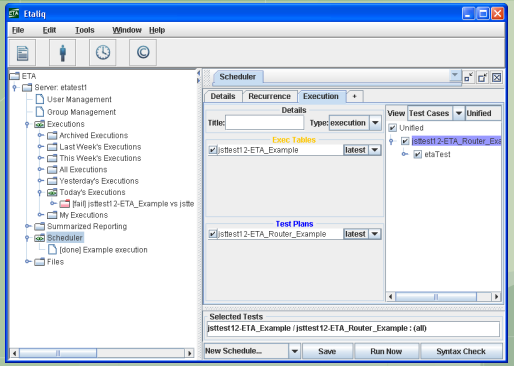


- Select the *Scheduler* link from the *Left Tree* to activate the *Scheduler* tool in the right pane
- Expand the *Scheduler* tree item to display existing schedules
- Select the *Scheduler*'s *Execution* tab
- Select the execution table named `user-ETA_Example`
- Select the test plan named `user-ETA_Router_Example`

ETALIQ
intelligent quality

Task #1: Execution

Starting the Schedule

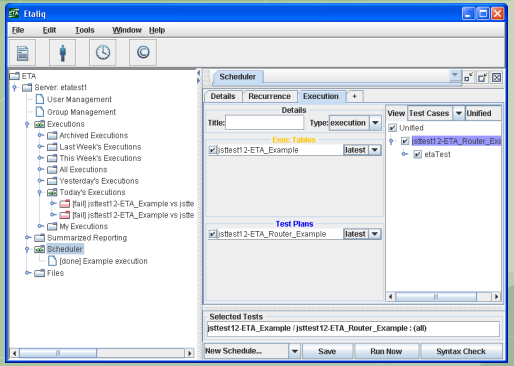


- Select the *Scheduler* link from the *Left Tree* to activate the *Scheduler* tool in the right pane
- Expand the *Scheduler* tree item to display existing schedules
- Select the *Scheduler*'s *Execution* tab
- Select the execution table named `user-ETA_Example`
- Select the test plan named `user-ETA_Router_Example`
- Click the *Run Now* button

ETALIQ
intelligent quality

Task #1: Execution

Live Summary Report

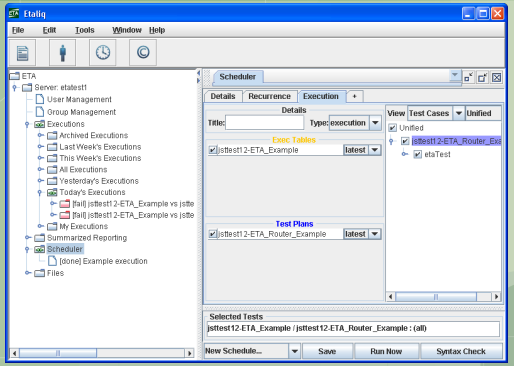


- The schedule just started appears under *Today's Executions* in the *Left Tree*

ETALIQ
intelligent quality

Task #1: Execution

Live Summary Report

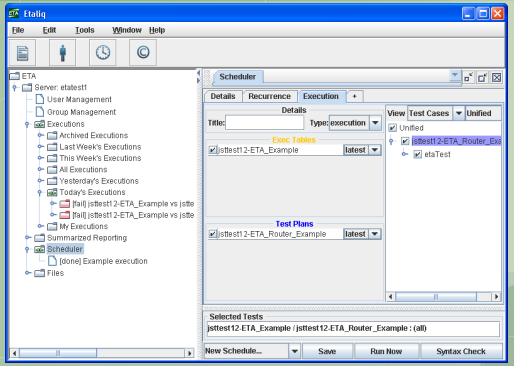


- The schedule just started appears under *Today's Executions* in the *Left Tree*
- Expand the schedule to reveal any verification and test plan executions

ETALIQ
intelligent quality

Task #1: Execution

Live Summary Report

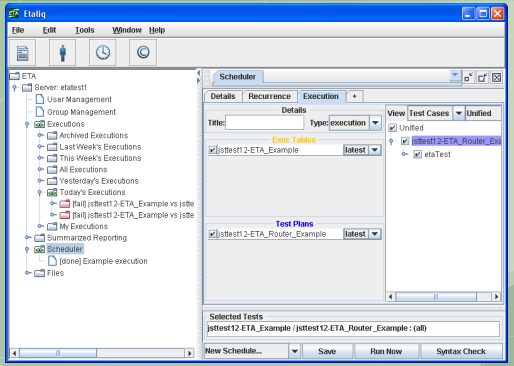


- The schedule just started appears under *Today's Executions* in the *Left Tree*
- Expand the schedule to reveal any verification and test plan executions
- Select the execution to see the live reports in the right pane

ETALIQ
intelligent quality

Task #1: Execution

Live Summary Report

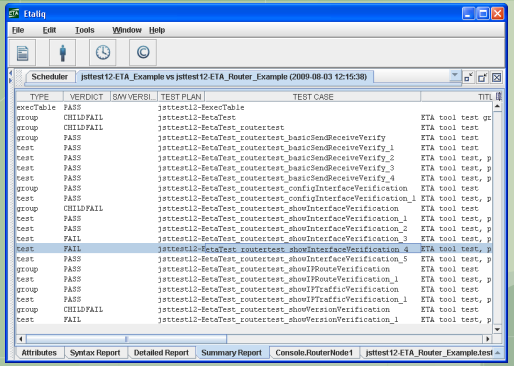


- The schedule just started appears under *Today's Executions* in the *Left Tree*
- Expand the schedule to reveal any verification and test plan executions
- Select the execution to see the live reports in the right pane
- Click the *Right* arrow on the *Resize Bar* in the middle of the screen to hide the *Left Tree* and leave more room to the *View Panel*

ETALIQ
intelligent quality

Task #1: Execution

Live Summary Report

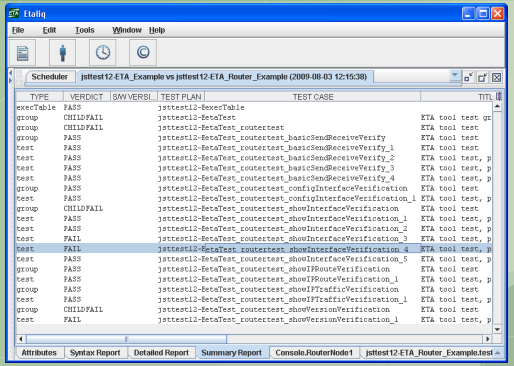


- The default selected tab is the *Summary Report*
- This tab updates live to reflect the verdict of each executed group and test

ETALIQ
intelligent quality

Task #1: Execution

Live Summary Report

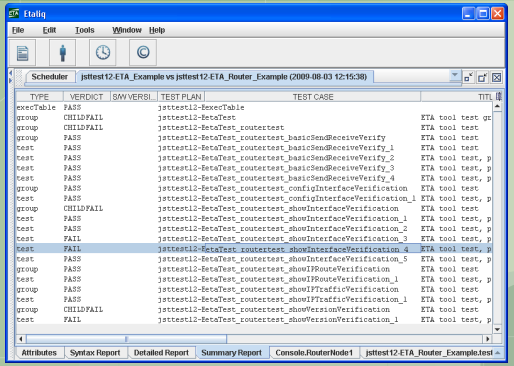


- The default selected tab is the *Summary Report*
- This tab updates live to reflect the verdict of each executed group and test
- Select this test from the table:
  ETA_RouterTest_showInterfaceVerification_4

ETALIQ
intelligent quality

Task #1: Execution

Live Summary Report

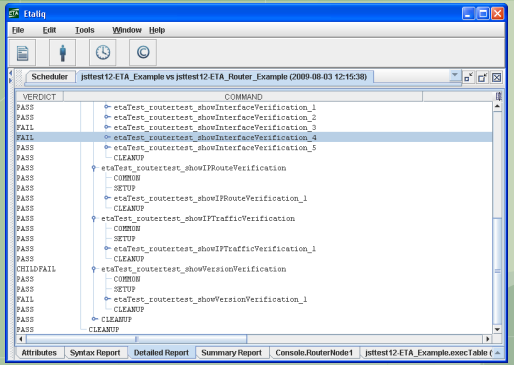


- The default selected tab is the *Summary Report*
- This tab updates live to reflect the verdict of each executed group and test
- Select this test from the table:
  ETA_RouterTest_showInterfaceVerification_4
- Select the *Detailed Report* tab to jump directly to the details of its execution

ETALIQ
intelligent quality

Task #1: Execution

Live Detailed Report

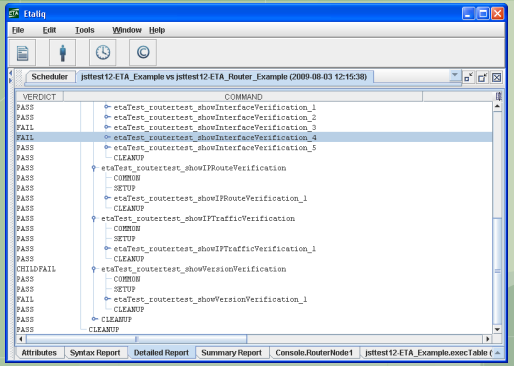


- The detailed report is a hierarchical representation of every test group, test case, section and step

ETALIQ
intelligent quality

Task #1: Execution

Live Detailed Report

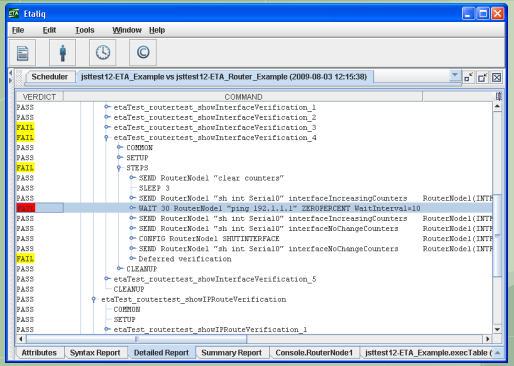


- The detailed report is a hierarchical representation of every test group, test case, section and step
- Every step executed is logged along with its verdict, details of execution and any node results

ETALIQ
intelligent quality

Task #1: Execution

Finding Failures

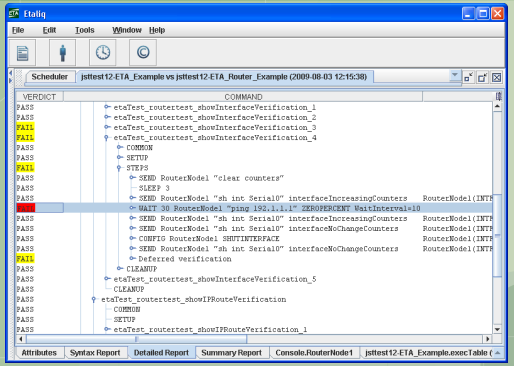


- Focus by selecting the currently highlighted line

ETALIQ
intelligent quality

Task #1: Execution

Finding Failures

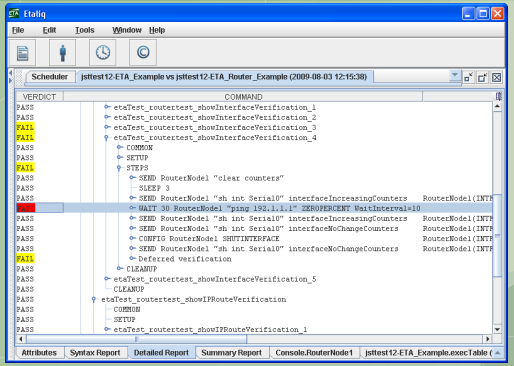


- Focus by selecting the currently highlighted line
- Press F4 to find the first failure

ETALIQ
intelligent quality

Task #1: Execution

Finding Failures

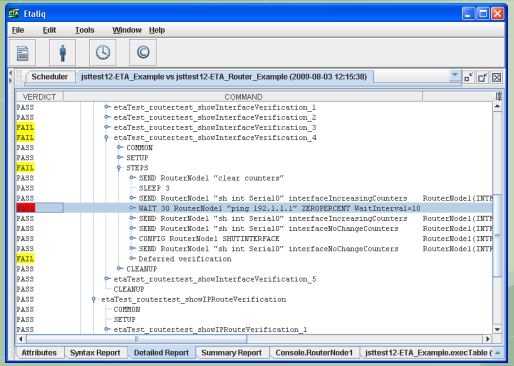


- Focus by selecting the currently highlighted line
- Press F4 to find the first failure
- Press F4 again to find the next failure

ETALIQ
intelligent quality

Task #1: Execution

Finding Failures

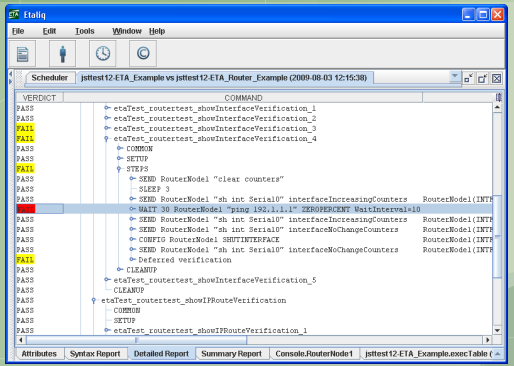


- Focus by selecting the currently highlighted line
- Press F4 to find the first failure
- Press F4 again to find the next failure
- Focus by selecting the currently highlighted line

ETALIQ
intelligent quality

Task #1: Execution

Finding Failures

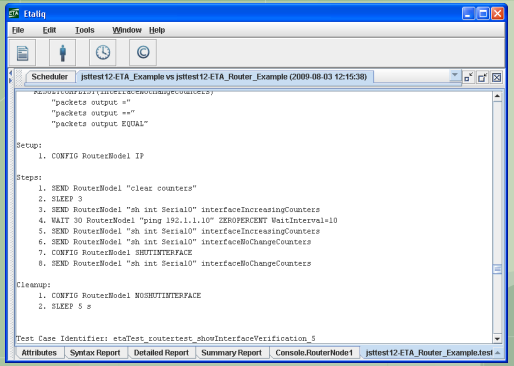


- Focus by selecting the currently highlighted line
- Press F4 to find the first failure
- Press F4 again to find the next failure
- Focus by selecting the currently highlighted line
- Select the *user-ETA_Router_Example.testPlan* tab to jump directly to the test case code (access hidden tabs using the *Tab Selector* arrow on the right side of the tab list)

ETALIQ
intelligent quality

Task #1: Execution

Updating Test Code

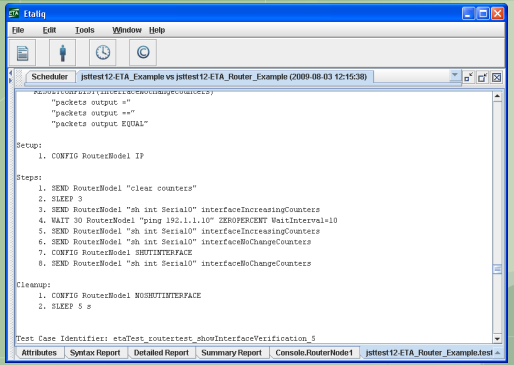


- Select the *File* → *Unlock* menu item

ETALIQ
intelligent quality

Task #1: Execution

Updating Test Code

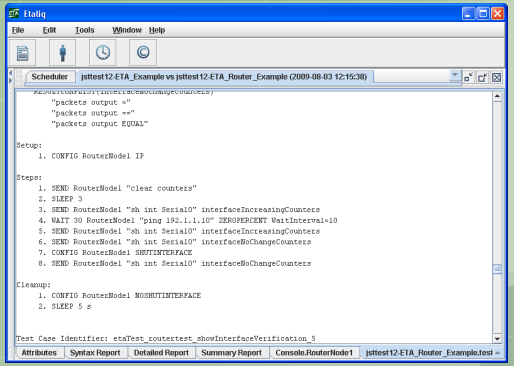


- Select the *File* → *Unlock* menu item
- Update the test case code
  (For example, change the ping destination address from `192.168.1.1` to `192.168.1.10`)

ETALIQ
intelligent quality

Task #1: Execution

Updating Test Code

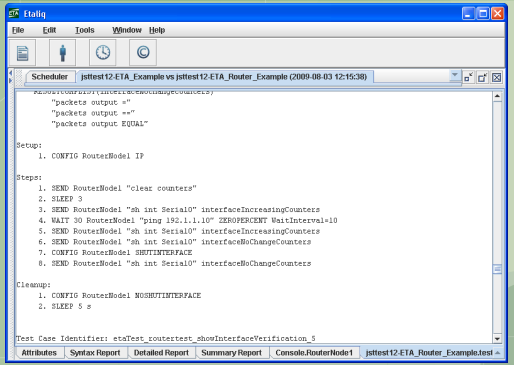


- Select the *File* → *Unlock* menu item
- Update the test case code
  (For example, change the ping destination address from `192.168.1.1` to `192.168.1.10`)
- Select *File* → *Save* to save the changes
  (A “Success” dialog will pop-up to confirm the operation)

ETALIQ
intelligent quality

Task #1: Execution

Updating Test Code



- Select the *File* → *Unlock* menu item
- Update the test case code
  (For example, change the ping destination address from `192.168.1.1` to `192.168.1.10`)
- Select *File* → *Save* to save the changes
  (A “Success” dialog will pop-up to confirm the operation)
- Your test is ready to execute again.

It can’t get any easier than this!

ETALIQ
intelligent quality

Task #2: Execution Summarization

Learn to:

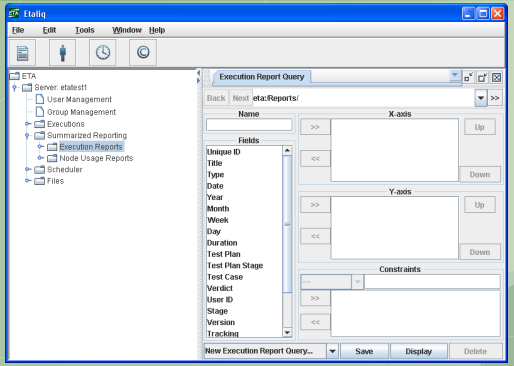
- create a summarized execution report
- drill-down to access finer details
- access details of execution reports

Register at `http://www.etaliq.com` to access the *Live* ETA *Demo*

ETALIQ
intelligent quality

Task #2: Execution Summarization

Summarized Reporting

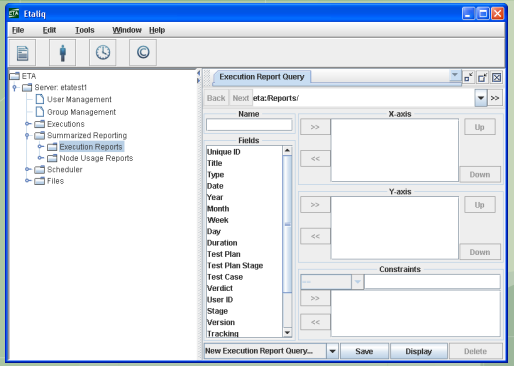


- Expand the *Summarized Reporting* tree item to display a list of report types

ETALIQ
intelligent quality

Task #2: Execution Summarization

Summarized Reporting

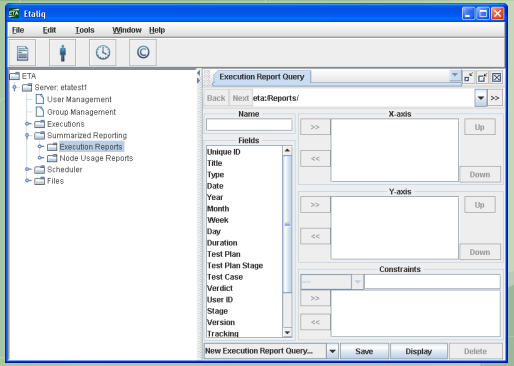


- Expand the *Summarized Reporting* tree item to display a list of report types
- *Execution Reports* rolls up the results of multiple executions to provide summarized *Execution Reports*

ETALIQ
intelligent quality

Task #2: Execution Summarization

Summarized Reporting

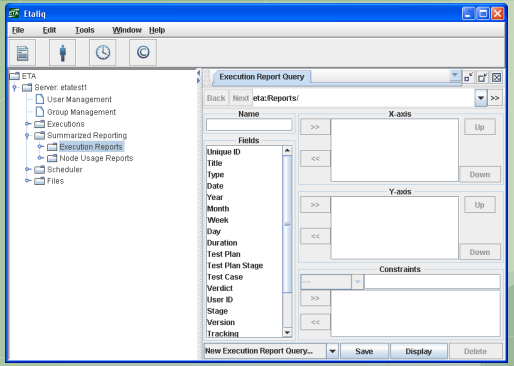


- Expand the *Summarized Reporting* tree item to display a list of report types
- *Execution Reports* rolls up the results of multiple executions to provide summarized *Execution Reports*
- *Node Usage Reports* tracks the usage of nodes by executions and manual lockouts to provide summarized *Node Usage Reports*

ETALIQ
intelligent quality

Task #2: Execution Summarization

Summarized Reporting

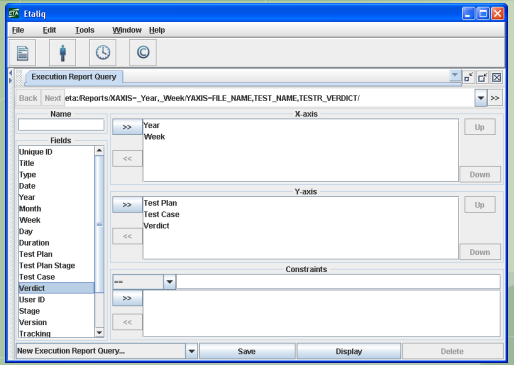


- Expand the *Summarized Reporting* tree item to display a list of report types
- *Execution Reports* rolls up the results of multiple executions to provide summarized *Execution Reports*
- *Node Usage Reports* tracks the usage of nodes by executions and manual lockouts to provide summarized *Node Usage Reports*
- Select the *Execution Reports* link to activate the *Execution Report Query*

ETALIQ
intelligent quality

Task #2: Execution Summarization

Execution Report Query

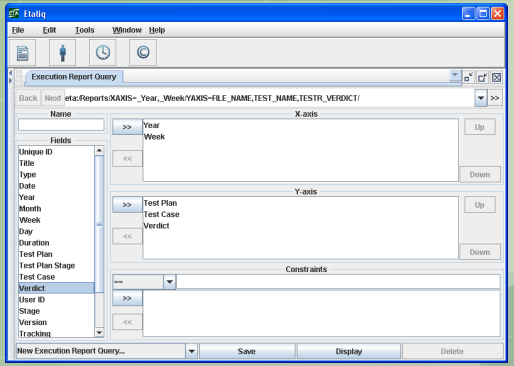


- From this screen, many combinations of fields can be selected to appear on the X- or Y-axis of the report

ETALIQ
intelligent quality

Task #2: Execution Summarization

Execution Report Query

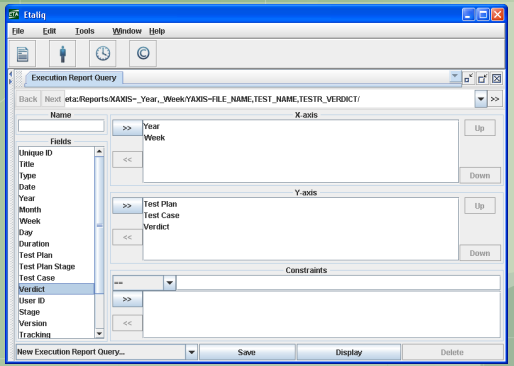


- From this screen, many combinations of fields can be selected to appear on the X- or Y-axis of the report
- For each of *Year* and *Week*, select the field and press the top ">>" button to add it to the X-axis

ETALIQ
intelligent quality

Task #2: Execution Summarization

Execution Report Query

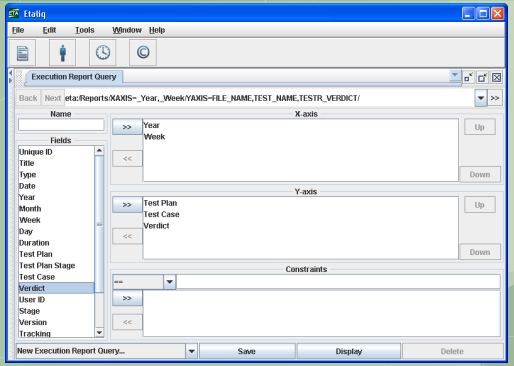


- From this screen, many combinations of fields can be selected to appear on the X- or Y-axis of the report
- For each of *Year* and *Week*, select the field and press the top “>>” button to add it to the X-axis
- For each of *Test Plan*, *Test Case* and *Verdict*, select the field and press the middle “>>” button to add it to the Y-axis

ETALIQ
intelligent quality

Task #2: Execution Summarization

Execution Report Query

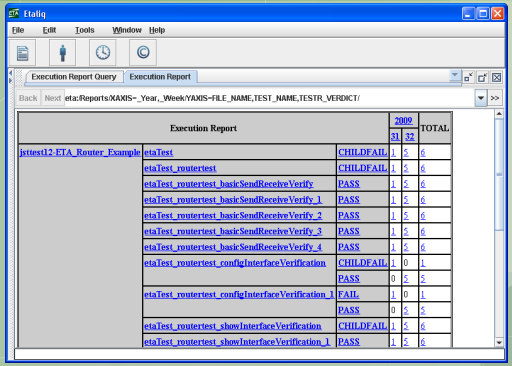


- From this screen, many combinations of fields can be selected to appear on the X- or Y-axis of the report
- For each of *Year* and *Week*, select the field and press the top “>>” button to add it to the X-axis
- For each of *Test Plan*, *Test Case* and *Verdict*, select the field and press the middle “>>” button to add it to the Y-axis
- Press the *Display* button to create the report

ETALIQ
intelligent quality

Task #2: Execution Summarization

Execution Report



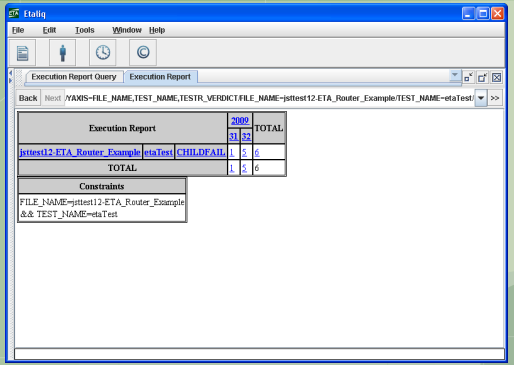
Execution Report			2009		TOTAL
			31	32	
jrttest12-ETA_Router_Example	etaTest	CHILDFAIL	1	5	6
	etaTest_routertest	CHILDFAIL	1	5	6
	etaTest_routertest_basicSendReceiveVerify	PASS	1	5	6
	etaTest_routertest_basicSendReceiveVerify_1	PASS	1	5	6
	etaTest_routertest_basicSendReceiveVerify_2	PASS	1	5	6
	etaTest_routertest_basicSendReceiveVerify_3	PASS	1	5	6
	etaTest_routertest_basicSendReceiveVerify_4	PASS	1	5	6
	etaTest_routertest_configInterfaceVerification	CHILDFAIL	1	0	1
		PASS	0	5	5
	etaTest_routertest_configInterfaceVerification_1	FAIL	1	0	1
		PASS	0	5	5
	etaTest_routertest_showInterfaceVerification	CHILDFAIL	1	5	6
	etaTest_routertest_showInterfaceVerification_1	PASS	1	5	6

- Following any link will "drill-down" to the next level of report

ETALIQ
intelligent quality

Task #2: Execution Summarization

Execution Report

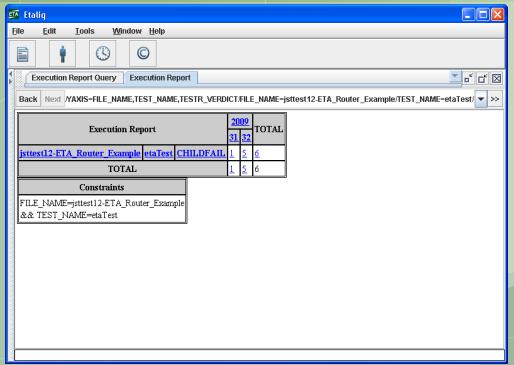


- Following any link will “drill-down” to the next level of report
- When a header link is followed, the created report will only include that row or column

ETALIQ
intelligent quality

Task #2: Execution Summarization

Execution Report

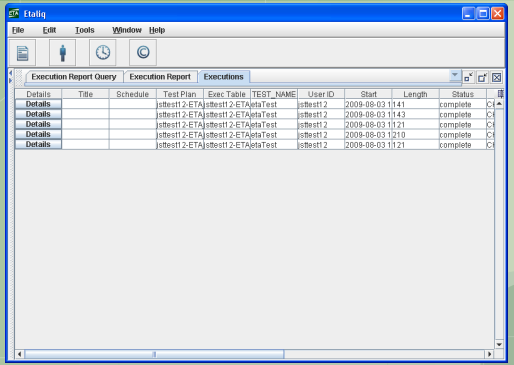


- Following any link will “drill-down” to the next level of report
- When a header link is followed, the created report will only include that row or column
- When a cell link is followed, an Executions table will be displayed of all the executions that matched that cell

ETALIQ
intelligent quality

Task #2: Execution Summarization

Execution Table

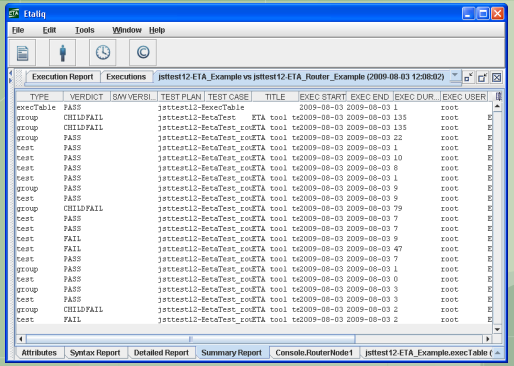


- Within the Executions table, a *Details* button appears in the first column of every individual execution.

ETALIQ
intelligent quality

Task #2: Execution Summarization

Execution Table



- Within the Executions table, a *Details* button appears in the first column of every individual execution.
- Click one to open all the report files associated with that specific execution.

All information is just a click away!

ETALIQ
intelligent quality

Case Studies

innovation in automation

innovation in automation

www.etaliq.com

ETALIQ
intelligent quality

Case Study #1

Test Automation: In-house Isn't the Answer

Who

- Telecom hardware start-up
- 50 engineers

Challenge

- Rapid black-box device testing
- Long term regression tests
- Broad coverage

Solution

- Old Linux server
- ETA Client and Server install
- A Student Engineer

Result

- Useful tests immediately
- 140 tests in less three months
- Protocol errors corrected
- Impressive customer feedback

Download the full text at `http://www.etaliq.com`

ETALIQ
intelligent quality

Case Study #2

Test Automation: Do More Testing in Less Time

Who

- Carrier class network equipment provider
- Over 10,000 engineers

Challenge

- Configure DUT in preparation for manual testing
- Currently takes each engineer 2–6 weeks

Solution

- ETA Client and Server Install
- Develop a flexible table-based configuration suite

Result

- 6 person months meet 99% of requirements
- Provided 80 additional test cases to verify stability
- Became a licenced Etaliq ETA customer
- ROI well beyond expectations

Download the full text at `http://www.etaliq.com`

ETALIQ
intelligent quality

Case Study #3

Test Automation: Do More Testing with Less Hardware

Who

- Carrier class network equipment provider
- Over 10,000 engineers

Challenge

- Determine current testbed configuration, and adapt test cases to suit
- Give summarized reports of result by combination

Solution

- ETA Client and Server Install
- 4 person months development time
- Developed a configuration reader
- Created 130 adaptable tests creating 2,200 unique results

Result

- Up to 2,500 results per day, per testbed
- ROI well beyond expectations

Download the full text at http://www.etaliq.com

ETALIQ
intelligent quality

Case Study #4

Test Automation: Solve Problems Quickly

Who

- Carrier class network equipment provider
- Over 10,000 engineers

Challenge

- Place the DUT under heavy stress
- Allow other tests in parallel

Solution

- ETA Client and Server Install
- 6 person days development time

Result

- Solution delivered
- Fast, reliable, customizable
- ROI well beyond expectations

Download the full text at `http://www.etaliq.com`

ETALIQ
intelligent quality

Conclusion

ETALIQ
intelligent quality

Innovation in Automation

Etaliq revolutionizes and accelerates automated testing by providing an integrated framework where test plans and scripts are merged into a single document using **ETA**'s patent-pending command language.

ETALIQ
intelligent quality

Manual & Automation Related Questions

What if you could

- Substantially reduce automation complexity?

ETALIQ
intelligent quality

Manual & Automation Related Questions

What if you could

- Substantially reduce automation complexity?
- Use automation to assist in manual testing?

ETALIQ
intelligent quality

Manual & Automation Related Questions

What if you could

- Substantially reduce automation complexity?
- Use automation to assist in manual testing?
- Have your Subject Matter Experts (SMEs) write the automation?

ETALIQ
intelligent quality

Manual & Automation Related Questions

What if you could

- Substantially reduce automation complexity?
- Use automation to assist in manual testing?
- Have your Subject Matter Experts (SMEs) write the automation?
- Create and verify your automation prior to feature availability?

ETALIQ
intelligent quality

Manual & Automation Related Questions

What if you could

- Substantially reduce automation complexity?
- Use automation to assist in manual testing?
- Have your Subject Matter Experts (SMEs) write the automation?
- Create and verify your automation prior to feature availability?
- Get summary pass/fail reporting real time by day/week/month?

ETALIQ
intelligent quality

Manual & Automation Related Questions

What if you could

- Substantially reduce automation complexity?
- Use automation to assist in manual testing?
- Have your Subject Matter Experts (SMEs) write the automation?
- Create and verify your automation prior to feature availability?
- Get summary pass/fail reporting real time by day/week/month?
- Create table-driven setups?

ETALIQ
intelligent quality

Manual & Automation Related Questions

What if you could

- Substantially reduce automation complexity?
- Use automation to assist in manual testing?
- Have your Subject Matter Experts (SMEs) write the automation?
- Create and verify your automation prior to feature availability?
- Get summary pass/fail reporting real time by day/week/month?
- Create table-driven setups?
- Create your automation in $1/10^{th}$ the time?

ETALIQ
intelligent quality

Conclusion

If the answer is *YES*.

ETALIQ
intelligent quality

Conclusion

If the answer is *YES*.

We can help.

ETALIQ
intelligent quality

Conclusion

If the answer is *YES*.

ETALIQ
intelligent quality

Conclusion

If the answer is *YES*.

ETALIQ
intelligent quality

innovation in automation