



# La Simulation matériel et logiciel dans un système de test automatique



Victor Fernandes  
Marvin Test Solutions  
European Manager



## Aircraft Armament Equipment

- Bomb Racks
- Ejector Racks
- Missile Rail Launchers
- Multiple Carriage Systems
- Pylons



## Test & Support Systems

- Armament and munition Test Sets
- SMS Test Sets
- Flightline, I-level, and depot-level testers
- Production test equipment



## Land Systems

- Mission Critical Equipment Trailers and Shelters
- Vehicle Environmental Systems
- Auxiliary Power Units
- Weapons Stations



## Tactical Vehicles

- Light Strike Armored or Unarmored
- Rescue and Personnel Recovery
- Reconnaissance
- Anti-tank



## Machining Tooling Prototype

- Armored Personnel Carriers
- MRAP
- Abrams
- Bradley
- FMTV
- JLTV
- Stryker

**Total Solutions & Support Capabilities including repair, overhaul, upgrades, maintenance & training**

# Les clients

## Military

U.S. Armed Forces and our allies

## Aerospace

U.S. and Foreign Aerospace Firms

## Manufacturing

Industrial and Semiconductor



## ■ Solutions de Test Flightline

MTS-3060 SmartCan™



MT1888



MT3045 IRIS



AN/TMS-205



- F-16
- F-15
- F-35
- F-22
- A-10
- TA-50
- AH-64
- AH-1
- UH-60
- Tigre

## ■ Solutions de Test Niveau intermédiaire

MTS-916



MTS-206



MTS-209



MTS-235



- F-16
- F-15
- F-35
- Supporte autres avions et munitions

# Nos Solutions (cont.)

## ■ Solutions de Test Dépôt

TS-376



TS-217



TS-201



- F-16
- F-15
- F-35
- Supporte autres avions et munitions

## ■ Solutions de Test pour l'industrie

GBATS



GENASYS



TS-900



- Avionique, Défense , Semi-conducteurs et autres applications de test fonctionnel

# Full Spectrum Product Portfolio

Supporting our customers with innovative products & systems

Hardware Building Blocks



Software Building Blocks

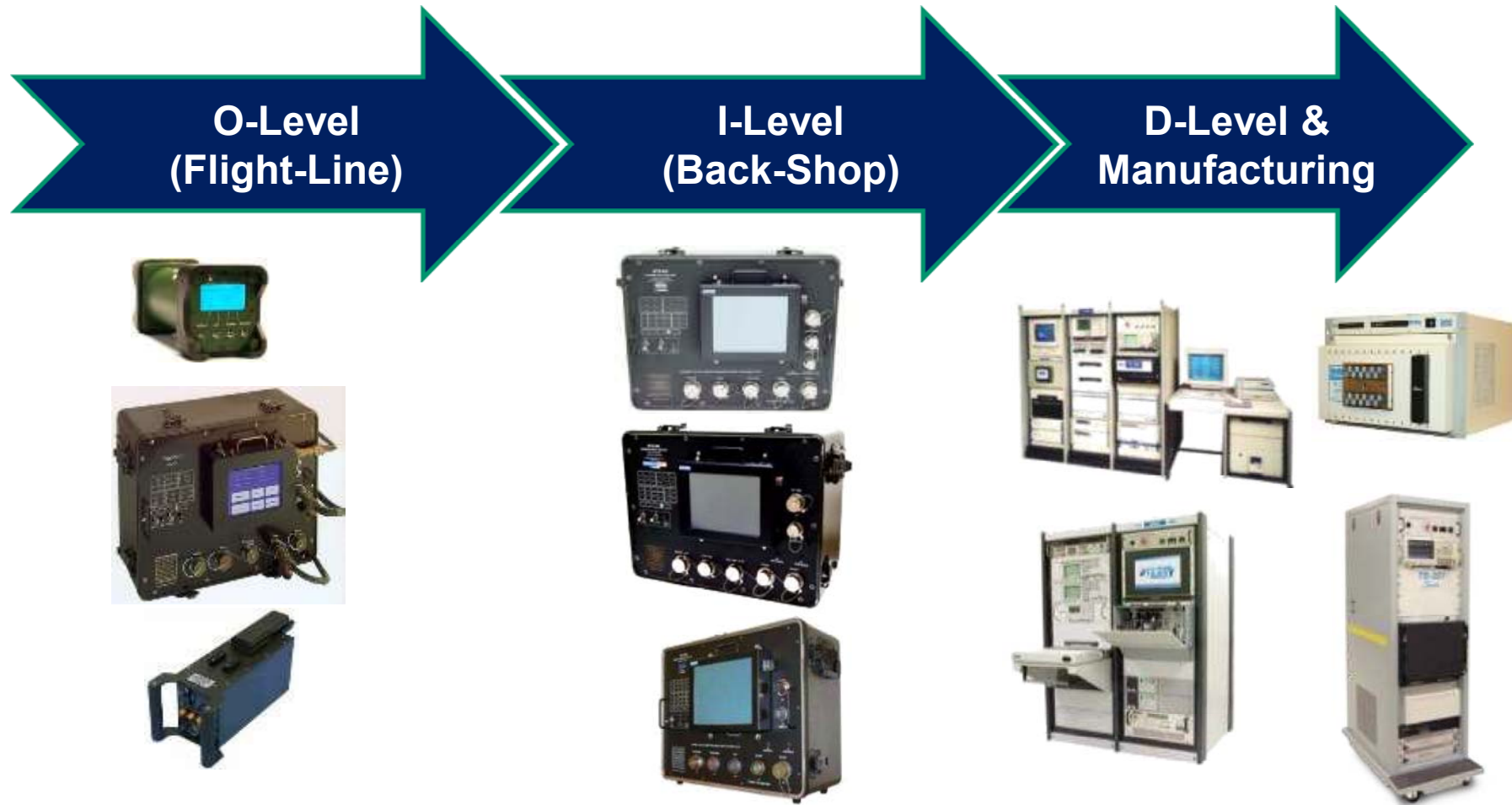


Preconfigured Systems & Subsystems

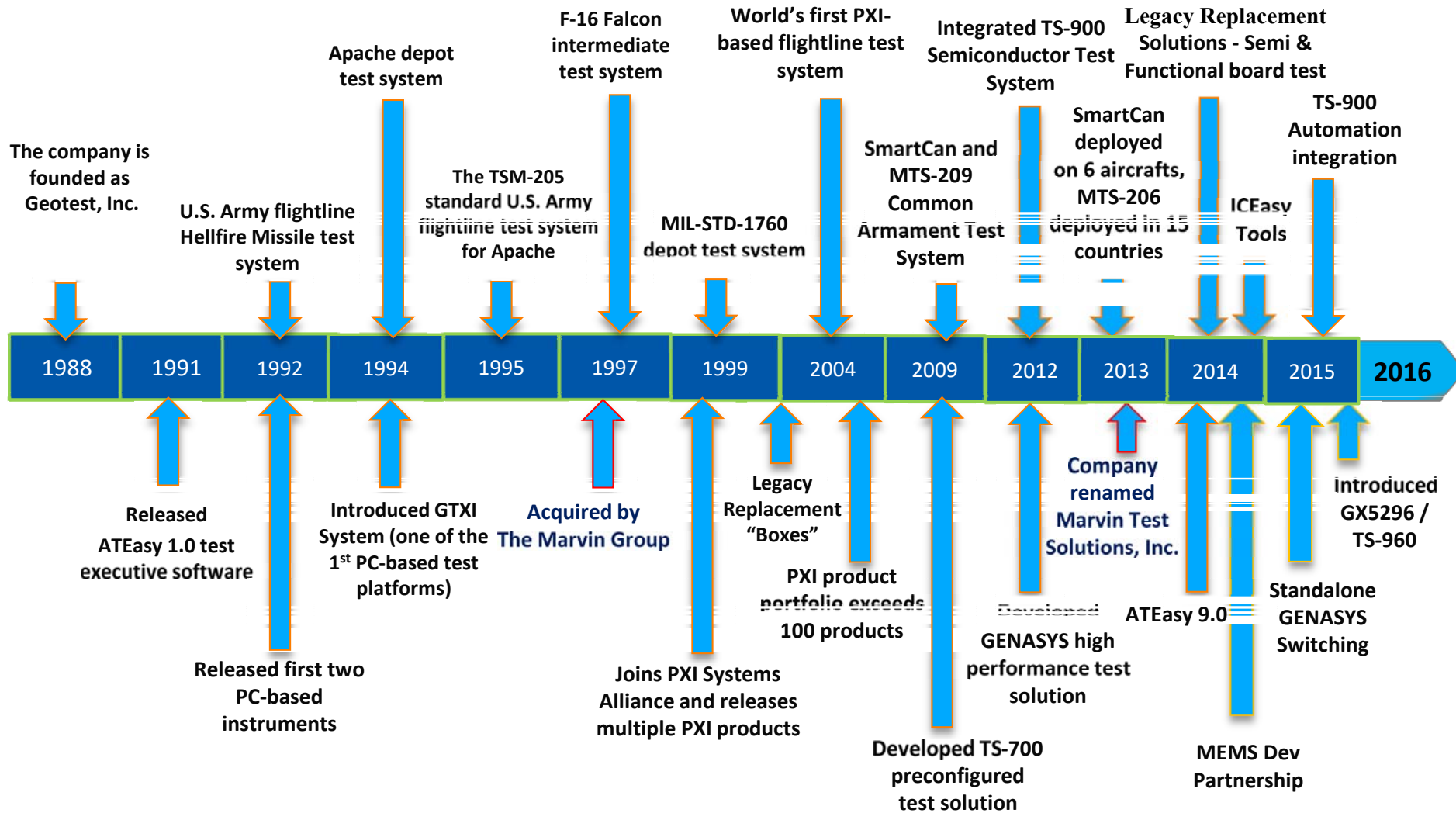


# Vertical Integration Offers a Range of Test Solutions

Supporting our customers with innovative Turn-Key test solutions



# Over 25 Years of Innovative Test Systems





# PXI Products



# “Integrated” PXI Test Systems

## TS-700 Series

Bench-top PXI-based testers for digital, avionics and mixed-signal test applications



## TS-323 GENASYS

High performance mixed signal functional test system for production and depot applications



## MTS-207 Platform

Ultra-rugged PXI platform for demanding field and flightline applications

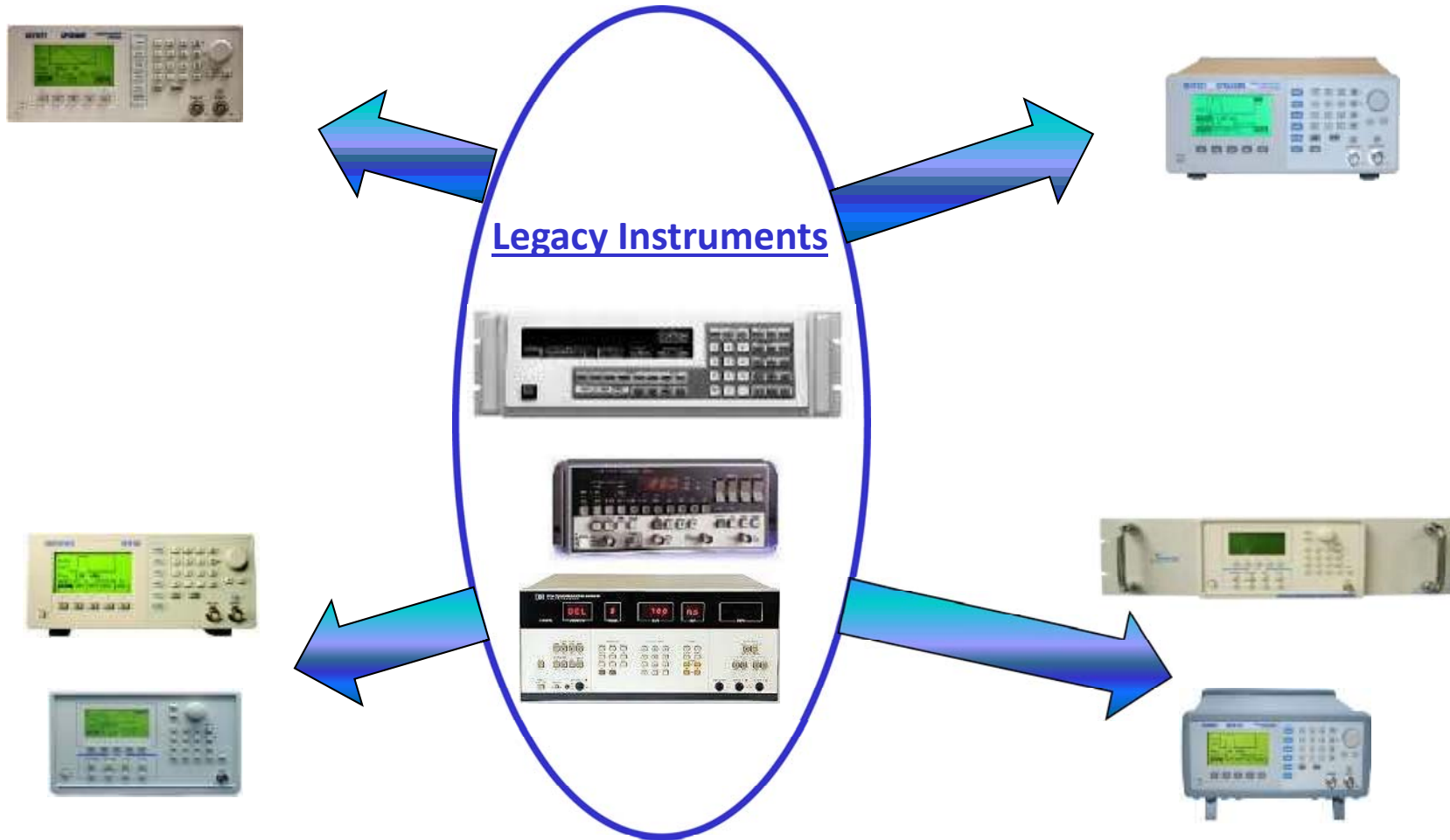


## TS-900 Series

High-performance semiconductor test system



# Legacy Replacement Products



# Software Tools Make A.T.E. Easy



An analog waveform development tool



A digital vector editor for all MTS' high-performance digital I/O products

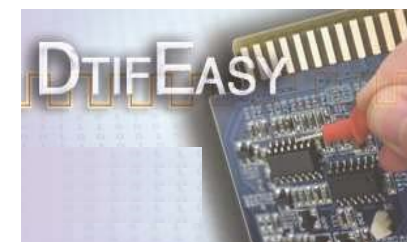


An integrated, open-architecture Test Executive and Development Suite for test applications deployed since 1991

Calibration and Verification software



A LASAR post-processor test and troubleshooting tool



# La Simulation matériel et logiciel dans un système de test automatique

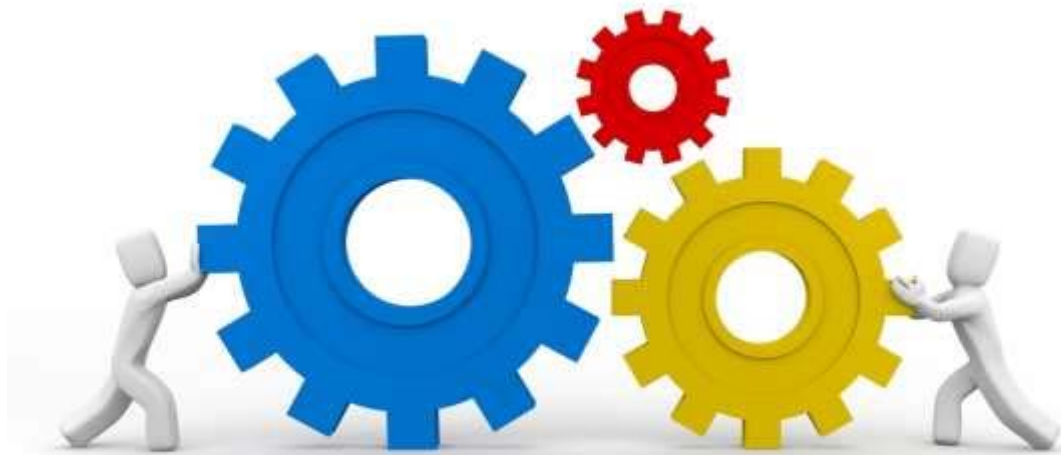
La simulation nécessite l'exécution de l'application avec un sous-ensemble de l'instrumentation ou sans instrumentation et avec ou sans l'unité sous test (UUT)

La solution présentée expose en détail la mise en œuvre d'un système de simulation qui est créé par l'amélioration du moteur d'exécution de l'application qui fait partie d'un environnement logiciel de développement d'applications de test automatique ATEasy



# What is Test System Simulation?

Simulation is the imitation of the operation of a test system over time, including the system instruments, the test system test application and the UUT.



# Why do we need to simulate?

- Develop, Debug and Run, Test your application on your desktop with no hardware (Instruments or UUT)
  - System time, Better resource utilization
  - Engineering time – faster development and debugging cycles
  - Short time to market
  
- Code and application robustness
  - Create test/use cases, debug code and create more robust app
  - Trace and analyze performance

- Run your application with or without instruments
- Run your application without UUT
- Run your application with or without hardware interface (i.e. GPIB, VXI, LXI, USB, PXI, RS232, etc)
- Redirect Procedures to Simulation Code
- Minimal Code changes because of simulation
- Access and Perform Everything:
  - Get/Set Parameters/Return Value, Call Stack, variables, tests, etc.
  - Execute code



# What Currently Available?

- IVI Simulation
  - Pretty limited in Scope – only Driver
  - No UUT, software, UUT simulation
  - Only IVI Driver, Driver Vendor must supply Simulation driver to support it
  
- Other...
  - Custom implementation ...

**..... AND.....**

# ATEasy

- ❖ *ATEasy* is a full test development solution that includes a test executive and software application development environment, all in one application suite
- ❖ *ATEasy* is a development environment that combines the structure of **ATLAS**, the ease of Microsoft **Visual Basic** with the flexibility of Microsoft **Visual C++** into one easy-to-use integrated package



## 1. Framework driven:

- Streamline, Placeholder for everything
- Defined process for editing, running, debugging

## 2. A Test Engineer's Language:

- Test Technicians, Programmers and EE
- Instrument and Interface Interchangeability
- Self Documented, TRD oriented and easy to read (Commands)

## 3. Open Architecture

- Integrates with software standards and tools (DLL, DDE, Com/ActiveX, .NET, C, Function Panel drivers/LabWindows CVi, IVI & VISA, LabView , HTML, Source Control, ATML...)
- Supports for all standard control interfaces: (Serial, GPIB, VXI, USB, PC/PXI, TCP/IP, LXI...)

## 4. Backwards compatibility

- Any ATEasy version can open and use v1.0 files – programs, systems, and drivers
- Expandable File Format for drivers, system, programs; project format has not changed since v4.0

## 5. RAD (Rapid Application Development) tool

- Dolt!, TestIt!, TaskIt!, FormIt!, LoopIt!, ProgramIt!
- Shortens development cycle for editing, compiling running, debugging

## 6. Complete, integrated environment

- Test Executive integrated into the development environment
- Minimal external tools required, simplifies setup and deployment

## 7. Built-in Application Builder generates Royalty-Free run-time executables (.EXE files) and libraries (.DLL)

# ATEasy's Key Product Features (cont'd)

8. **Test Executive, Profile** offers full control of test execution, sequencing, test conditions, and data logging
9. **Integral Fault Analysis**
10. **Multiple UUT Testing** (Parallel or Sequential)
11. **Users & Groups based privileges** - Can be customized without writing additional code, menus, toolbar, options, debugging level
12. **Visual Basic-like forms**, menus and controls.
13. Connectivity with many **Source Controls Providers** and built in configuration management tools
14. **Simulation** – ATEasy offers full support for simulation: Instrument (no hardware), Software, UUT, and more

# ATEasy TRD Structure

The image displays the ATEasy TRD Structure interface, which is divided into several panes. On the left, the 'Test Properties' pane lists test tasks and their details. In the center, the 'TRD Structure Tests/Tasks' pane shows a hierarchical tree of test components. On the right, the 'Test Code' pane displays the underlying code for a specific test task.

**Test Properties:**

- 1. Power Supply Test:** Validate the operation of each power module prior to... For each test, verify the Go/NoGo status of the test and, where applicable, n
- 1.1. Shorts Test:** Verifies that each UUT power supply input is not shorte
- 1.1.1. 3.3V Inputs:** Pass Criteria: 33 Ohms <= Measured Value
- 1.1.2. 5.0V Inputs:** Pass Criteria: 33 Ohms <= Measured Value
- 1.2. 3.3V Power Tests:** Verify 3.3V regulation under various loads
- 1.2.1. No Load:** Program the power supply to 3.3V and measure into... Pass criteria: 3.2V <= Measured Value <= 3.4V
- 1.2.2. Minimum Load:** Program the power supply to 3.3V and measur... 100mA load applied. Pass criteria: 3.15V <= Measured Value <= 3.4V
- 1.2.3. Nominal Load:** Program the power supply to 3.3V and measure load applied. Pass criteria: 3.10V <= Measured Value <= 3.4V
- 1.2.4. Maximum Load:** Program the power supply to 3.3V and measu... load applied. Pass criteria: 3.10V <= Measured Value <= 3.4V
- 1.2.5. Over-Current:** Program the power supply to 3.3V and measure load applied. Pass criteria: Power supply over-current indicator set
- 1.3. 5.0V Power Tests:** Verify 5.0V regulation under various loads
- 1.3.1. No Load:** Program the power supply to 5.0V and measure into... Pass criteria: 4.9V <= Measured Value <= 5.25V
- 1.3.2. Minimum Load:** Program the power supply to 5.0V and measur... 100mA load applied. Pass criteria: 4.85V <= Measured Value <= 5.25V

**TRD Structure Tests/Tasks:**

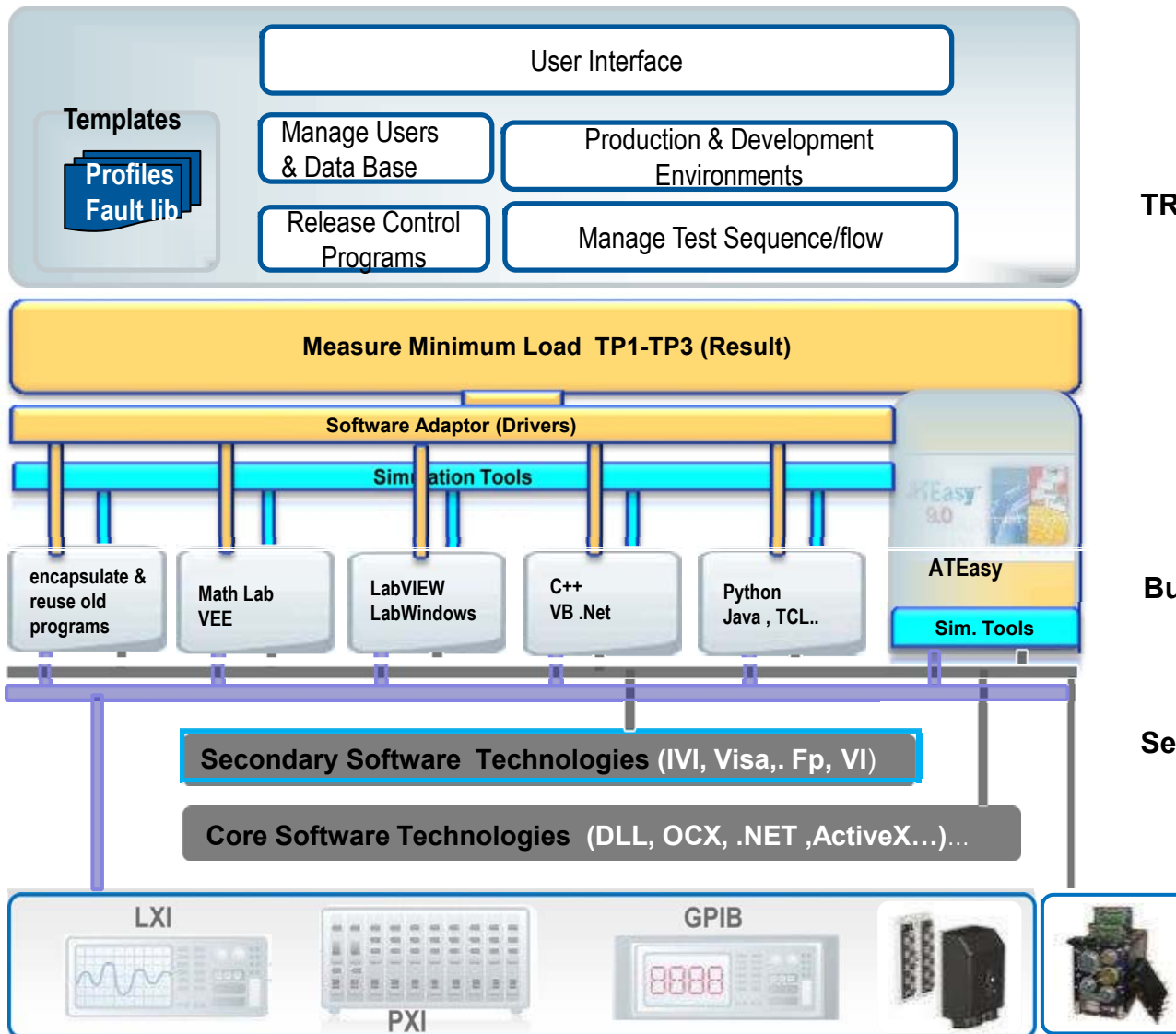
- Program
  - Tests
    - Power Supply
      - Shorts
        - 3.3V Inputs
        - 5.0V Inputs
      - 3.3V Power
        - No Load
        - 100mA Load
        - 2.5A Load
        - 5.0A Load
        - Over Current
      - 5.0V Power
        - No Load
        - 100mA Load
        - 2.5A Load
        - 5.0A Load
        - Over Current
      - Power-On
        - 3.3V Voltage
        - 3.3V Current
        - 5.0V Voltage
        - 5.0V Current
- Forms
- Commands
- Events
- Procedures
- Variables
- Types
- Libraries

**Test Code:**

```
Task 1.1 : Shorts
Test 1.1.1 : 3.3V Inputs (Pin=, Unit=, Type=MinMax, Min=33, Max=)

_3_3V_Short (m_Handle, m_Error, m_Handle, TestResult, m_Error)
```

# Incorporation of multiple Software technologies



**Test Executive**

**TRD /Signal description System commands**

```

1. Power Supply Test - Verify the operation of each power supply prior to connecting to the UUT.
   1.1. Setup Test - Load the test script and verify that it is correct.
   1.2. Run Test - Execute the test script and verify that it is correct.
   1.3. Pass/Fail - 23 (Pass) or 24 (Fail)
2. Load Regulation Test - Verify the operation of each power supply under load.
   2.1. Setup Test - Load the test script and verify that it is correct.
   2.2. Run Test - Execute the test script and verify that it is correct.
   2.3. Pass/Fail - 25 (Pass) or 26 (Fail)
3. Voltage Regulation Test - Verify the operation of each power supply under load.
   3.1. Setup Test - Load the test script and verify that it is correct.
   3.2. Run Test - Execute the test script and verify that it is correct.
   3.3. Pass/Fail - 27 (Pass) or 28 (Fail)
4. Current Regulation Test - Verify the operation of each power supply under load.
   4.1. Setup Test - Load the test script and verify that it is correct.
   4.2. Run Test - Execute the test script and verify that it is correct.
   4.3. Pass/Fail - 29 (Pass) or 30 (Fail)
5. Output Voltage Test - Verify the operation of each power supply under load.
   5.1. Setup Test - Load the test script and verify that it is correct.
   5.2. Run Test - Execute the test script and verify that it is correct.
   5.3. Pass/Fail - 31 (Pass) or 32 (Fail)
6. Output Current Test - Verify the operation of each power supply under load.
   6.1. Setup Test - Load the test script and verify that it is correct.
   6.2. Run Test - Execute the test script and verify that it is correct.
   6.3. Pass/Fail - 33 (Pass) or 34 (Fail)
7. Output Power Test - Verify the operation of each power supply under load.
   7.1. Setup Test - Load the test script and verify that it is correct.
   7.2. Run Test - Execute the test script and verify that it is correct.
   7.3. Pass/Fail - 35 (Pass) or 36 (Fail)
8. Output Efficiency Test - Verify the operation of each power supply under load.
   8.1. Setup Test - Load the test script and verify that it is correct.
   8.2. Run Test - Execute the test script and verify that it is correct.
   8.3. Pass/Fail - 37 (Pass) or 38 (Fail)
9. Output Ripple Test - Verify the operation of each power supply under load.
   9.1. Setup Test - Load the test script and verify that it is correct.
   9.2. Run Test - Execute the test script and verify that it is correct.
   9.3. Pass/Fail - 39 (Pass) or 40 (Fail)
10. Output Noise Test - Verify the operation of each power supply under load.
    10.1. Setup Test - Load the test script and verify that it is correct.
    10.2. Run Test - Execute the test script and verify that it is correct.
    10.3. Pass/Fail - 41 (Pass) or 42 (Fail)
    
```

**Development Tools  
Bus Trace & debug / simulation  
Program documentation**

**Secondary Software Technologies**

**Core Software Technologies**

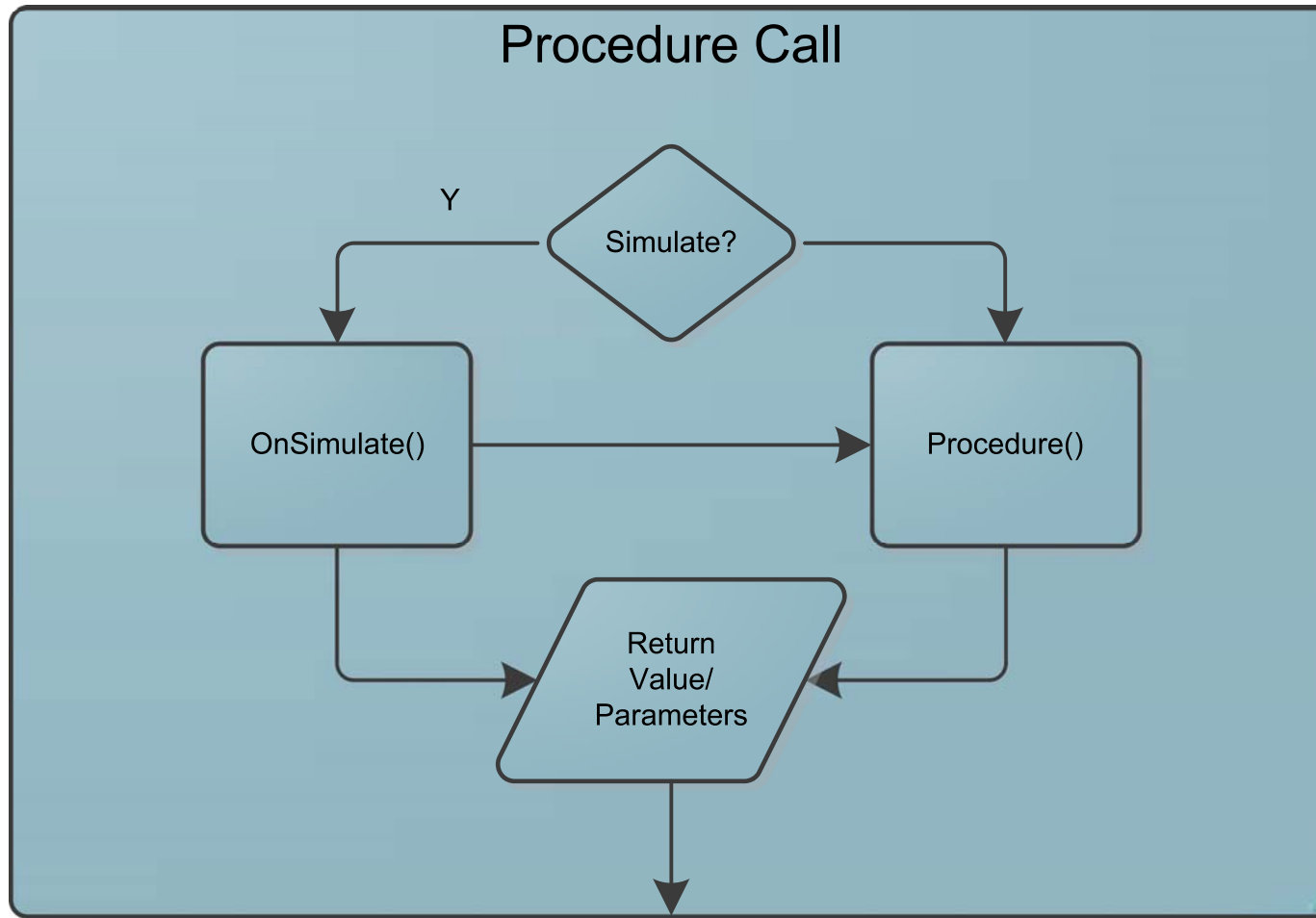
**UUT**

# How we Did It? (ATEasy Implementation)

- Set Simulation mode:
  - `App.Simulate=True`, Command line,
- Select Procedure to Simulate:
  - `SetSimulateProcedures()`:
    - module, sub-module, wildcard, individual
    - Procedure, IOTable, DLL
  - `GetSimulateParameters()`, `GetCaller()`, Test
- Event occurs when simulated procedure is called
  - `OnSimulate(sModule, proc, penReturn)`: Any



# Simulation Procedure Call



# Simulation in Action - Setup

The screenshot shows a software interface for simulation setup. On the left, a menu is open with the 'Simulate' option highlighted. The main window, titled 'Simulation.drv', displays a tree view on the left with categories like Forms, Commands, Events, Procedures, IOTables, Variables, Types, and Libraries. The central area shows a code editor with the following content:

```
Events : OnInit(): Void Public
Occurs when the driver is started.
Name | Type | Description
endif
endif
! Select what to simulate
! code to handle these reside in OnSimulate of this driver
if App.Simulate
! simulate all the DMM procedures and IO Tables
assert(SetSimulateProcedures("DMM.IOTables")) ! all DMM driver IO Tabl
assert(SetSimulateProcedures("DMM.Procedures")) ! all DMM driver Procedu
assert(SetSimulateProcedures("DMM.SetRange")) ! one DMM procedure just
! the following 3 functions achive the same thing
assert(SetSimulateProcedures("Language.User32.Procedures.GetSystemMetrics"))
assert(SetSimulateProcedures("Language.GetSystemMetrics"))
assert(SetSimulateProcedures("Language.User32.GetSystemMetrics"))
```

# Simulation In Action - Implementation

```
! ATEasy Simulation Example
! Take a look at the SIM driver un
! When running this example in sim
! other DMM errors ignored
DMM Set Function VDC()
DMM Measure (TestResult)
```

```
sModule=obModule.name
select sModule
case "DMM"
  sProc=procCallee
  select sProc
  case "DMM.Measure"
    p=GetSimulateParameter("pdResult")
    p=6.0
```

```
Events : OnSimulate(obModule, procCallee, penSimulateStatus): Variant Public
Occurs when a simulated procedure or IOTable is call
Name / Type
obModule Val Object
procCallee Val Procedure
penSimulateStatus Var Internal.enumSimulateStatus
dMultiplier Double
i Long
sModule=obModule.name
select sModule
case "DMM"
  sProc=procCallee
  select sProc
  case "DMM.Measure"
    p=GetSimulateParameter("pdResult")
    p=6.0
  case "DMM.GetSystemIdentification"
    p=GetSimulateParameter("psID")
    p="34401A"
  endselect
  trace sProc ! trace all DMM calls
  penSimulateStatus=enSimulateDone
case "Language"
  sProc=procCallee
  select sProc
  case "Language.User32.GetSystemMetrics"
    ! language test 7.1
    if Test.Id="User32"
      p=GetSimulateParameter("smIndex")
      lIndex=p
      if lIndex=0 or lIndex=1
        vrReturn=1024
      endif
    endif
    trace sProc ! trace specific calls
    penSimulateStatus=enSimulateDone
  endselect
return vrReturn
```

# Simulation In Action – Test Executive

The screenshot displays the 'Simulation - DMMSimulation Program' window. The interface includes a menu bar (Program, View, Run, Conditions, Log, Tools, Help) and a toolbar with various control buttons like Start, Reset, Abort, Pause, and Continuous. The main area is divided into a left-hand tree view and a right-hand summary panel.

**Tree View:**

- PRO #1 - DMMSimulat...
- DMMSimulation
  - 1. DMM Tests
    - 1.1. Measure VDC

**Summary Panel:**

Company : Marvin Test Solutions  
 User name : Ronniye  
 Project : Simulation  
 Version : 1 (Fri Aug 30 13 14:41:15)

**Program : DMMSimulation [SIMULATION]**

UUT :  
 Version : 1 (Mon Sep 09 13 10:30:05)  
 Serial # :  
 Start time : 9/13/2013 10:49:21 AM

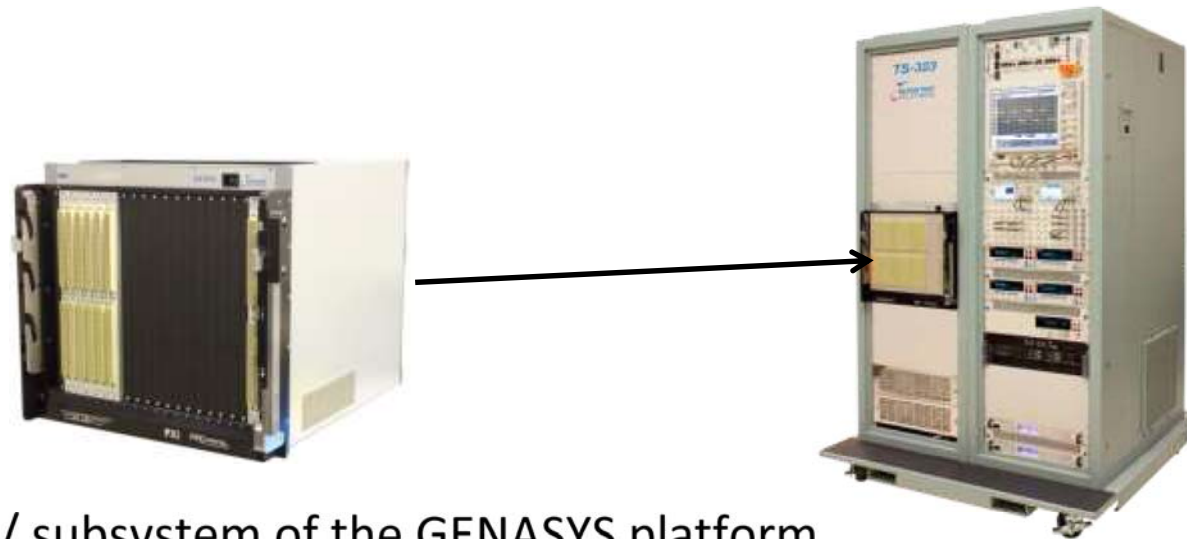
**Task 1 : DMM Tests**

#	Test Name	Pin	Unit	Min	Result	Max	Status
001	Measure VDC	P1-11	Volt	+5.9500	+6.0000	+6.0500	Pass

Stop Time : 9/13/2013 10:49:21 AM  
 Elapsed Time : 0.00 minutes  
**UUT Status : Pass [SIMULATION]**  
 Signature : \_\_\_\_\_

**Status Bar:** Skips to next task if exists | SIMULATION | Pass | READY | 00:00:00

# GENASYS Switching Subsystem



- Key component / subsystem of the GENASYS platform
- Innovative switching subsystem provides a hybrid-pin and “any resource to any pin” architecture
- Scalable system supports a minimum of 128 multiplexed hybrid pins, can be configured to support over 4500 interface pins
- Direct connect implementation interfaces all switching cards directly to the receiver interface, eliminating thousands of wires

# GENASYS Switching Subsystem

- GX7016 switching chassis with integrated MacPanel Scout receiver
  - 20 slot PXI chassis
  - Scout receiver & switch routing infrastructure
- Accommodates up to 18 “direct connect” switch cards
  - High density connection interface with no cabling
  - Internal, 16 wire analog bus with switch fabric to minimize stubbing
  - Compatible with a wide range of digital and analog resources



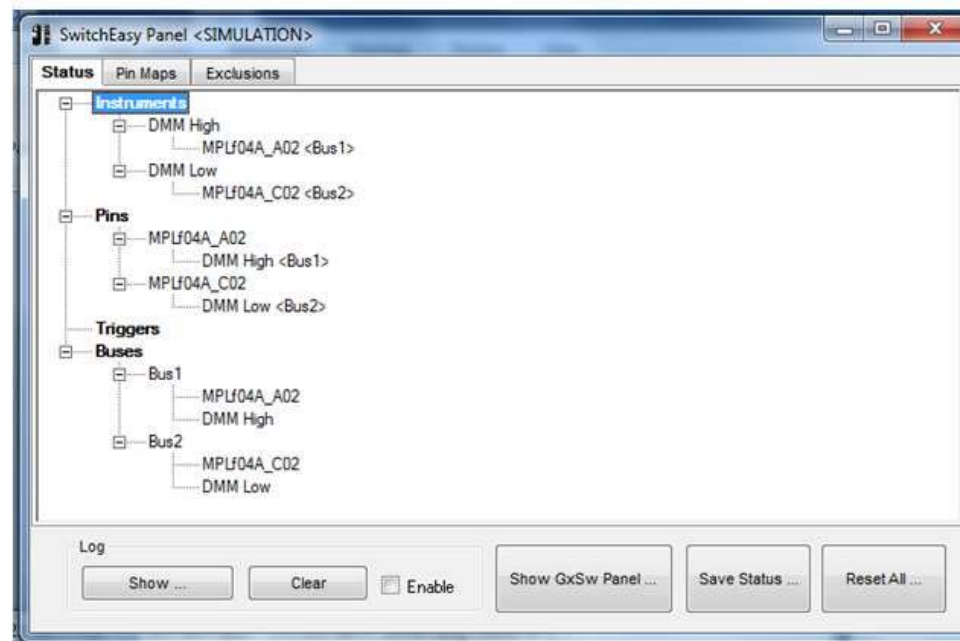
GX7016 Switching Chassis



GENASYS Switch Card

# GENASYS Switching Software - SwitchEasy

- Signal routing management tool
  - End to end signal routing with deterministic switching and switch path management
  - Simplifies programming with Resource-to-Receiver Pin and Resource-To-UUT Pin commands
- Seamless integration with ATEasy test development / test executive software
- Reports total number of relay operations for all switching cards ( values are stored in non-volatile memory on each card)



- Simulation is a powerful testing, debugging and analysis tool
- Simulation increase productivity of test engineers and system resource
- Need for a Run-Time access to implement simulation
- Simple design encourage usage and improve quality