



PRODUCTIVITY
ENHANCEMENT
SYSTEMS

IEEE P1641 Implementation Architecture

Prototype Demonstration



Objective

- Demonstrate the prototype of an implementation architecture for signals defined by IEEE P1641 (Standard for Signal and Test Definition)
 - Integrates products from Racal Instruments and TYX Corporation
 - Implements IEEE P1641, ATML and IVI Signal Interface emerging standards
 - Enables use of IEEE P1641 signals in high-complexity, ATE-independent TPSs



History

- Racal and TYX participate in the development of IEEE P1641
 - Racal and TYX applications support / will support the standard
- Integration prototype first demonstrated in Jan 2002; improvements:
 - “In-place” integration of software applications
 - Improves user experience and development productivity
 - Transfer of signal descriptions in XML format
 - Open architecture, ATML-compliant
 - Addition of automatic resource allocation services and IVI Signal Drivers
 - Instrument independence



Standards

- IEEE P1641
 - Basic Signal Components (BSC) – building blocks supporting the formal definition of signals
 - Test Signal Framework (TSF) – set of standard signal definitions (similar to ATLAS noun definitions)
- ATML
 - Signal descriptions
 - Capability descriptions
- IVI Signal Interface
 - COM API for signal-oriented control of instruments & signal-oriented description of instrument capabilities



Products

■ **Racal** *newWave*

- Visual design of P1641 signals
- Graphical signal simulation
- Signal synthesis (ARBs, DACs, synthetic instruments, ...)
- Export of signal descriptions (XML, ...)
- Integration with third-party applications via ActiveX

■ **TYX TestBase**

- Visual design of test strategies
- Execution of test procedures developed with various programming/test languages
- Storage of test results (XML, ...)
- Integration with third-party applications (ex. *newWave*)

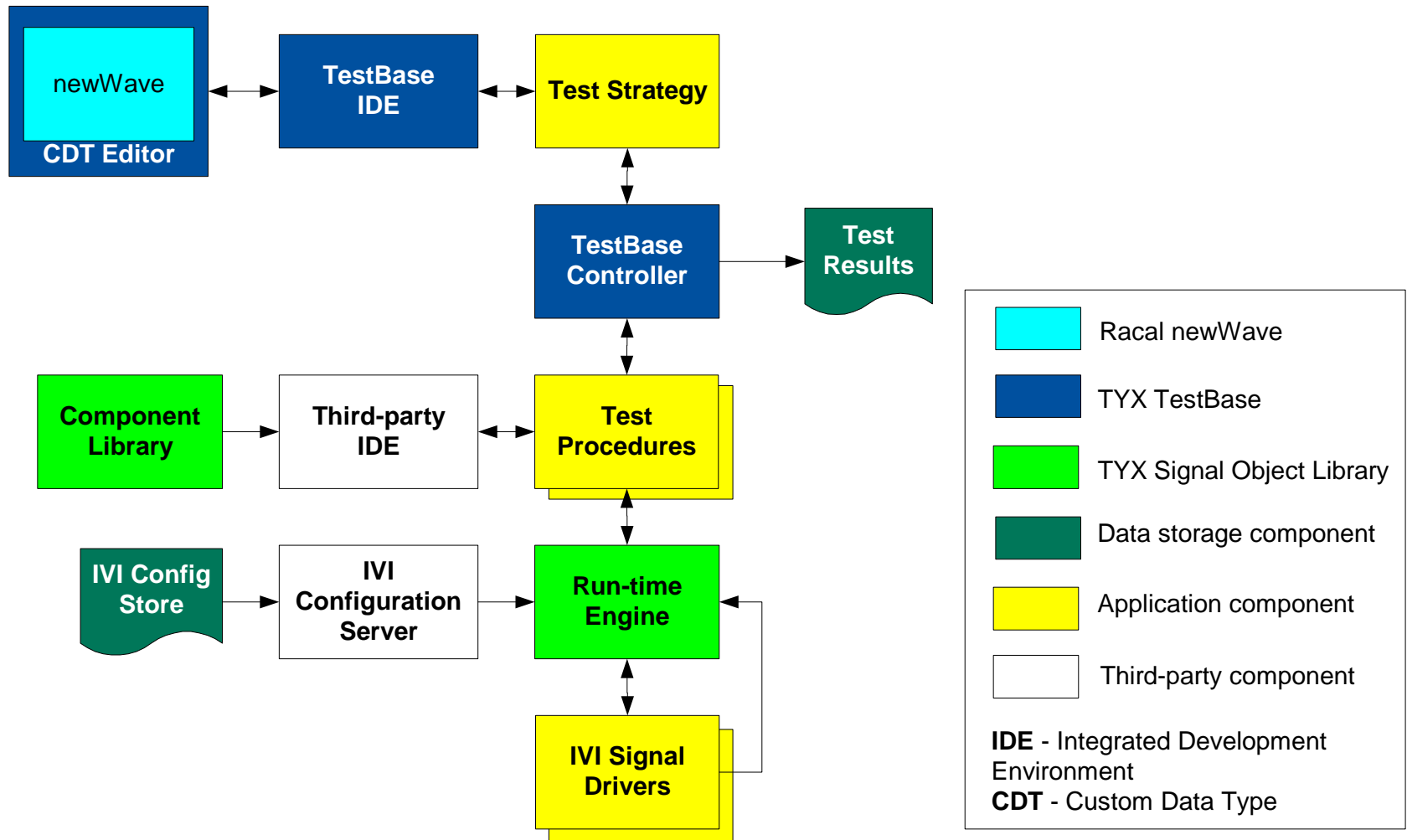


Products

- **TYX Signal Object Library** (prototype)
 - Component library implementing IEEE P1641 signals
 - Usable from COM-compatible programming/test languages
 - Run-time Engine
 - Automatic resource allocation
 - Switch path calculation
 - Signal-oriented control of instruments and switches via IVI Signal Drivers



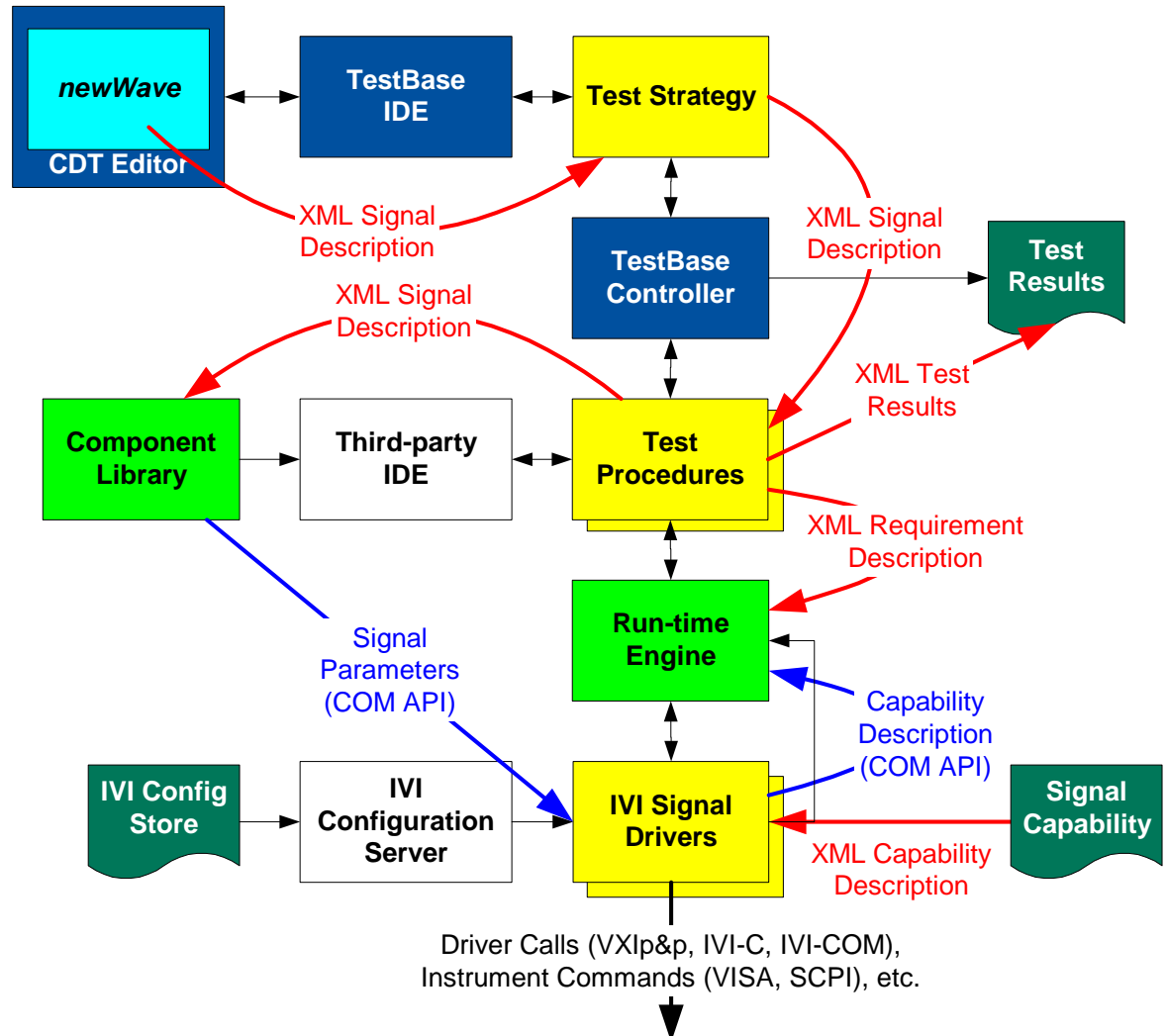
Software Architecture





Software Architecture ...

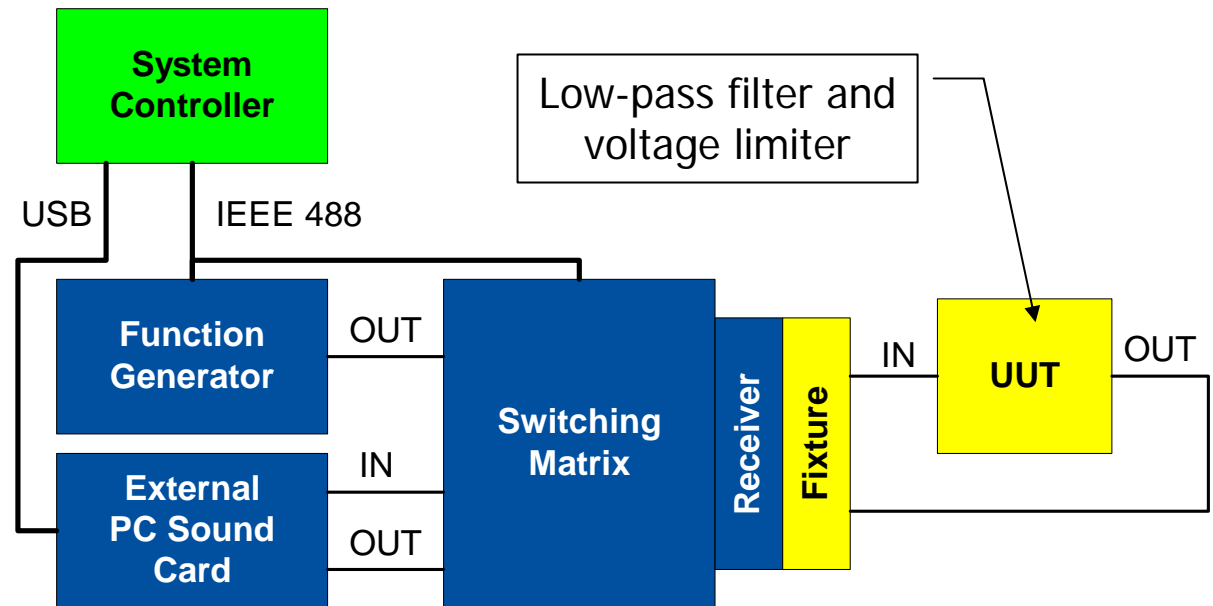
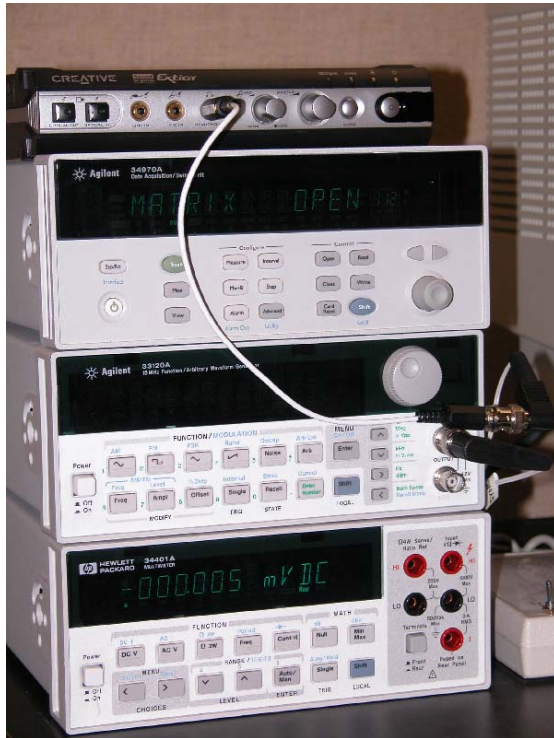
Data flows





Demonstration

- Hardware configuration





Demonstration...

- *newWave* integrated with **TestBase** through a Custom Data Type Editor

The screenshot shows the TestBase IDE interface. The main workspace displays a test flow diagram with a 'START' block, followed by an 'AC Signal Test (SignalResponse)' block, and then an 'All Signal Test (SignalResponse)' block. A red arrow points from the 'AC Signal Test' block to the 'Input Parameter Values' table below.

Parameter	Data Type	Unit	Source	Value
Meas_HI	string		Immediate	OUT
Meas_LO	string		Immediate	OUT_GND
Stim_HI	string		Immediate	IN
Stim_LO	string		Immediate	IN_GND
StimulusSignal	Signal.newWave		Immediate	newWave diagram...

Below the table, there are sections for 'Props', 'IPVs', 'OPVs', and 'Output'. The 'Output' section shows 'Terminating diagnosis', 'Outcome: OK', and 'Pass: False'. The status bar at the bottom indicates 'Ready' and the date/time '11/16/2003 6:18 PM'.

The screenshot shows the 'newWave Signal' window. The main area displays an 'AC SIGNAL (V)' waveform with parameters $\phi = 0$ and $f = 1000$. A red arrow points from the 'StimulusSignal' value in the TestBase IDE to this window. To the right of the waveform is a 'Shows' button and a small green waveform icon. A 'Palette Bar' on the right side contains the following items: 'AudioSource', 'SignalFunction', 'Viewers', and 'signal'. The status bar at the bottom shows 'All' and 'Event Functor'.



Demonstration...

- Graphical signal design and simulation in *newWave*

The screenshot displays the **newWave Signal** software interface. The main workspace shows a graphical signal design with two sinusoidal waveforms. The top waveform is labeled "Carrier" with parameters: $\text{amp} = 2.5$, $f = 5000$, and $\phi = 0$. The bottom waveform is labeled "In" with parameters: $\text{amp} = 2.0$, $f = 500$, and $\phi = 0$. These are connected to an "AM" block with $\text{modIdx} = 1.0$. A "Shows" block is connected to the output of the AM block. A red arrow points from the "Shows" block to the "Schemas:Views:Shows AM modIndex=1.0 SinusoidalVoltage 2.0V@500" window. This window displays the simulation results, showing a green waveform on a black background. The waveform is a modulated signal. The window includes a "Summary" tab and various controls: Channel 1, Gain, Offset, 5 V/Div, 0.000V, Trigger, Fall, 0.000V, Timebase, 500 uS/Div, Freeze, Sampling, 160000 Samples/Sec, and Start/Stop.

Schemas:Views:Shows AM modIndex=1.0 SinusoidalVoltage 2.0V@500

Summary View VBS XML Signal Streaming

Channel: 1

Gain: 5 V/Div

Offset: 0.000V

Trigger: Trigger

Fall: Fall

0.000V

Timebase: 500 uS/Div

Freeze: Freeze

Sampling: 160000 Samples/Sec

Start/Stop



Demonstration...

- XML generation from *newWave*

The screenshot shows the **newWave Signal** application window. The main workspace contains a block diagram with two input sinusoids and an AM modulator block. The top sinusoid is labeled "Carrier" with parameters: `Sinusoidal amp=2.5 f=5000 $\phi=0$` . The bottom sinusoid is labeled "In" with parameters: `Sinusoidal amp=2.0 f=500 $\phi=0$` . The AM modulator block is labeled "AM modIndex=1.0" and has a "Shows" button next to it. A red arrow points from the "Shows" button to a secondary window titled "Schemas: Users:Shows AM modIndex=1.0 SinusoidalVoltage 2.0V@500". This window has tabs for "Summary", "VBS", "XML", and "Signal Streaming". The "XML" tab is active, displaying the following XML code:

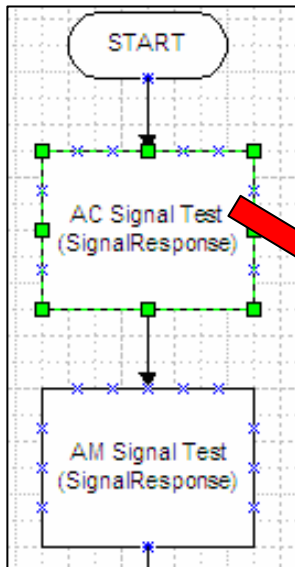
```
<?xml version="1.0" ?> <!--encoding="UTF-8"-->
<!-- created with newWave [http://www.racalstruments.com] by Chris Gorrige [Racal In
<!--W3C Schema generated by newWave [http://www.racalstruments.com] -->
<Signal name="" Out="AM24"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="C:\Program Files\Racal Instrume
    <Sinusoid name="SinusoidalVoltage148" amplitude="2.5" frequency="5000" />
    <Sinusoid name="SinusoidalVoltage149" amplitude="2.0" frequency="500" />
    <AM name="AM24" modIndex="1.0" Carrier="SinusoidalVoltage148" In="Sinus
</Signal>
```

Below the XML code is an "Edit" section with the text "No Errors".



Demonstration...

- Signal-based, UUT-oriented test strategy design in **TestBase**



Input Parameter Values

Action : AC Signal Test

Parameter	DataType	Unit	Source	Value
▶ Meas_HI	string		Immediate	OUT
Meas_LO	string		Immediate	OUT_GND
Stim_HI	string		Immediate	IN
Stim_LO	string		Immediate	IN_GND
StimulusSignal	Signal.newWave		Immediate	newWave diagram...

Props IPVs OPVs

UUT pins

signal
description
(newWave)



Demonstration...

- Signal-based, UUT-oriented test procedure development with the **Signal Object Library**

```
' Initialize Resource Controller
```

```
Dim rc As SIGLIBLib.ResController
```

```
Set rc = New SIGLIBLib.ResController
```

```
rc.Initialize strConfigStore, strFixture, bSimulate,
```

... resource allocation

```
' allocate signals
```

```
Dim inputSignal As Object
```

```
Set inputSignal = rc.GetSignalXML(Source, m_strXML)
```

XML requirements

```
Dim outputSignal As SIGLIBLib.Sensor_Waveform
```

```
Dim objReqParams As New ReqParameters
```

```
Set outputSignal = rc.GetSignal(SENSOR, "WAVEFORM", objReqParams)
```

```
' setup stimulus signal
```

```
inputSignal.SetupXML m_strXML
```

XML signal description

```
' connect stimulus
```

```
inputSignal.Connect m_strStim_HI, m_strStim_LO
```

signal operations

UUT pins



Demonstration...

```
' setup sensor signal
Dim dblSampleTime As Double
dblSampleTime = 4# * (1 / 1000#)
outputSignal.Setup dblSampleTime, IviFloat_NaN

' measure response
outputSignal.Connect m_strMeas_HI, m_strMeas_LO
outputSignal.Arm
Call outputSignal.Fetch(varSamples)
arrSamples = varSamples
outputSignal.Disconnect

' display response
Waveform2.ChartData = arrSamples
Waveform2.Layout

' disconnect stimulus
inputSignal.Disconnect

' reset signals
inputSignal.Reset
outputSignal.Reset
```

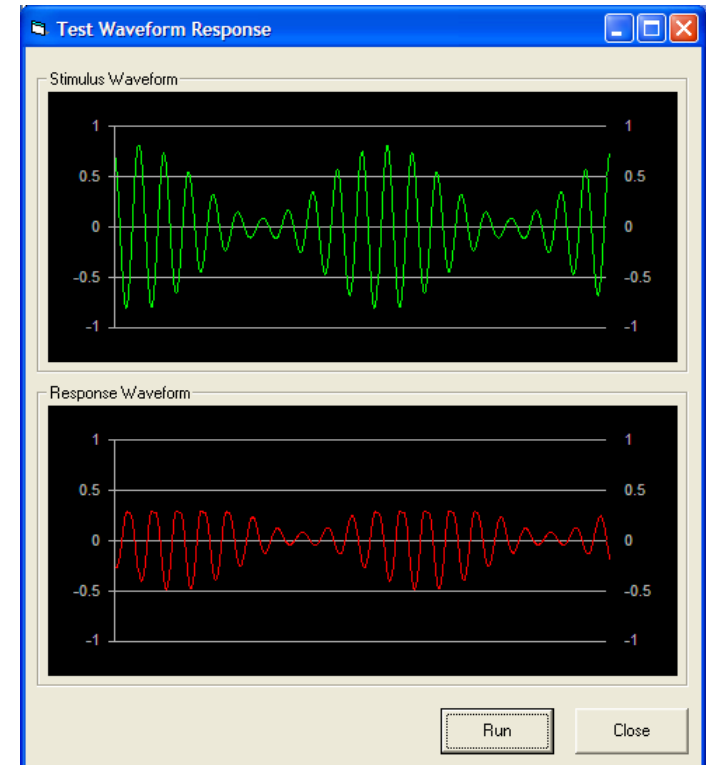
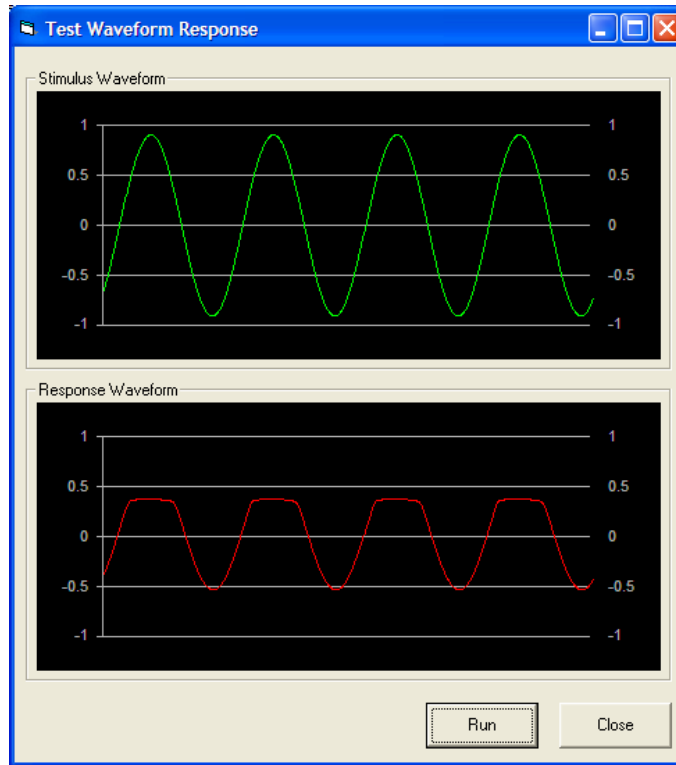
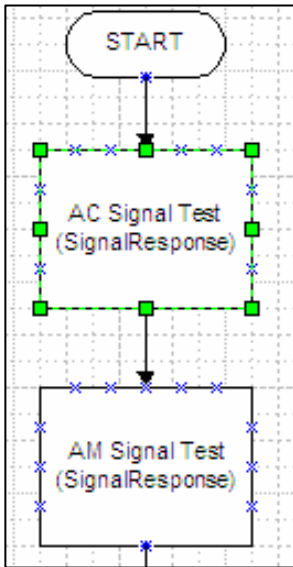


Demonstration...

- Test strategy execution in **TestBase**

- AC Signal Response

- AM Signal Response



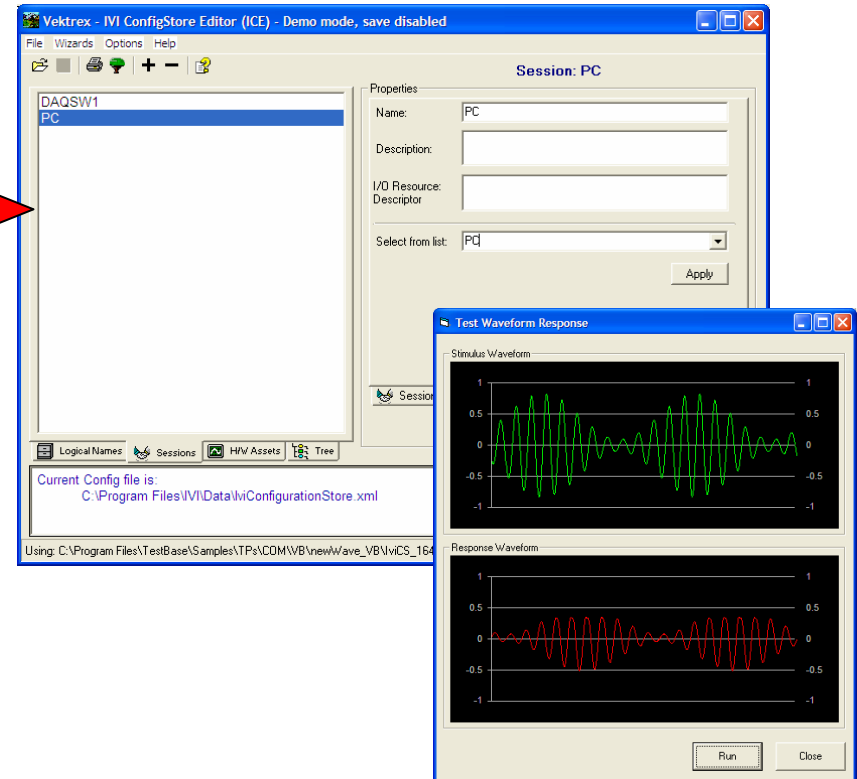
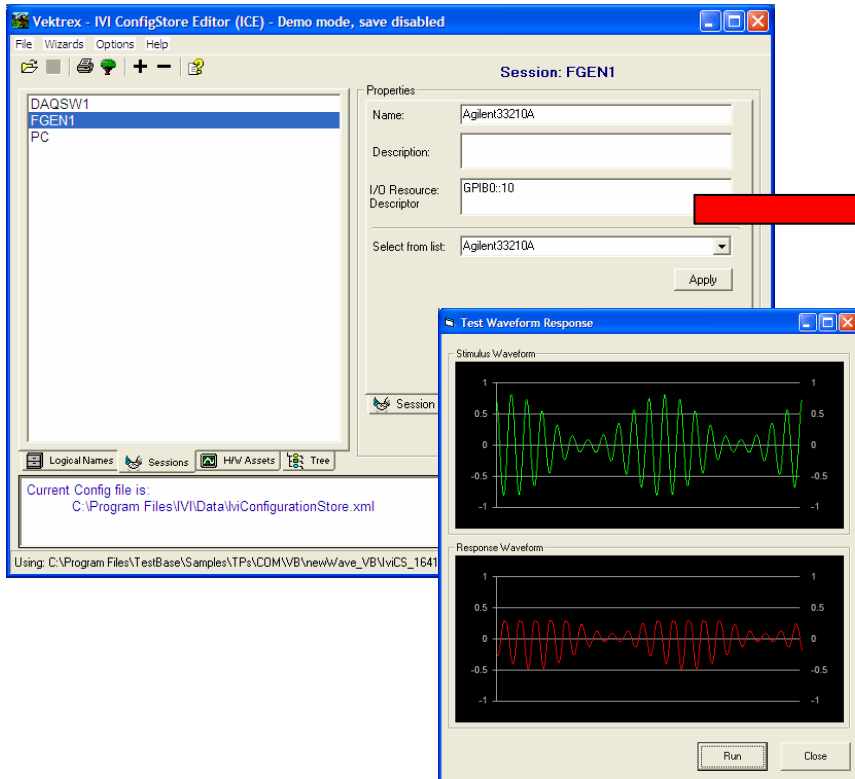


Demonstration...

■ Instrument interchangeability

- AM Signal Source - Agilent 33210A
- Waveform Sensor - PC sound card

- AM Signal Source - PC sound card
- Waveform Sensor - PC sound card





Conclusion

- Functional contribution to the integrated solution:
 - *newWave*: supports graphical signal design & simulation
 - Provides IEEE P1641 compliance
 - **TestBase**: enables utilization of signals in high-complexity TPSs
 - Supports implementation of signal measurements, fault isolation, UUT control, operator interface, etc.
 - **Signal Object Library** and **IVI Signal Drivers**: provide ATE independence
 - “Guaranteeable” performance
 - Supports both traditional and synthetic instrumentation



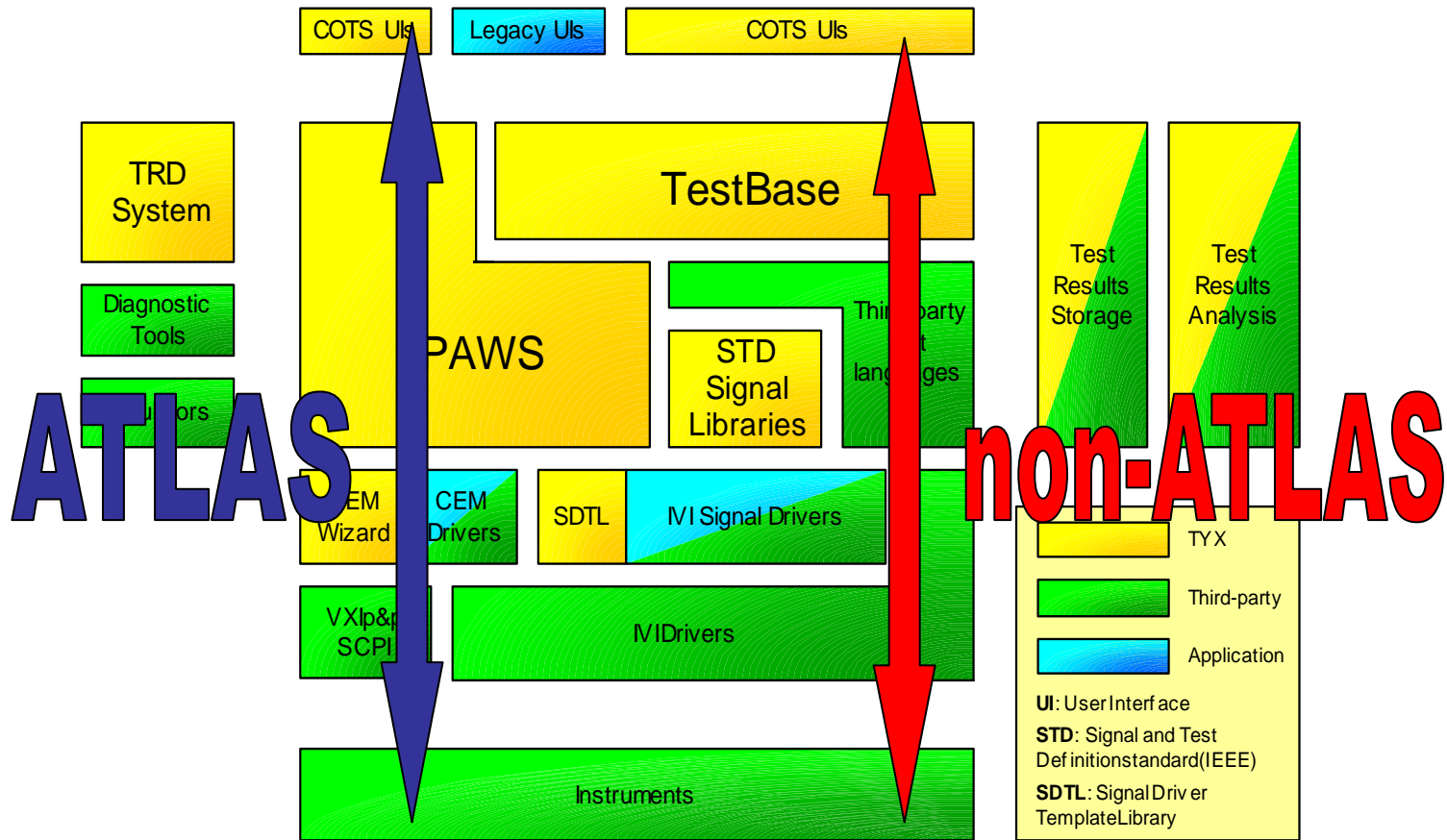
Conclusion...

- Benefits
 - **Visual design & simulation**
 - For signal descriptions & test strategies
 - Simplifies development, no programming expertise required
 - **Signal-based, UUT-oriented test specification**
 - At test strategy & test procedure level
 - Provides instrument independence
 - Simplifies development
 - Favors code reuse
 - **Support for synthetic instrumentation**
 - Signal-oriented control is the most effective approach
 - **Open architecture** – functionality can be extended by:
 - Import of diagnostic strategies in standard formats (DiagML)
 - Storage of test results in standard formats (ATML)
 - Integration with third-party programming languages and test environments
 - Development of custom production interfaces, etc.
 - **Standards compliance** - IEEE P1641, ATML, IVI



Conclusion...

- TYX Product Line





Conclusion...

■ **The next-generation TPS**

- Makes use of test executives and various programming languages & test environments
 - Enables parallel development
 - Favors code reuse
 - Offers better support for diagnostics
- ATE-independent
 - Signal-based, UUT-oriented
- Supported by software architectures where:
 - Multiple applications interoperate seamlessly
 - Enabled by component interfacing technologies (COM, Web services, ...)
 - Running on interconnected computers
 - Using XML for data exchange
- Can be implemented with existing technologies and existing products