Test Software Platform - Introduction

Product Overview and Demo

Neil Baliga
Agenda

Problem
Solution
Features
Demo
Key Questions
Discussion
Problem
Device complexity is **Increasing**
Test and data volumes are **Growing**
Product cycles are **Shrinking**
Test system costs are **Escalating**
Maintenance costs are **Climbing**
Test organizations are being **Squeezed**
Customer's Problem

Customers are struggling with homegrown test systems

- excessive code effort
- hard to analyze data
- manual labor
- delayed schedules
- missed product failures

Organizational pressures ensure this problem remains

- Color of money – Capital, Overhead, Program
- Mission of test orgs is to ship products, not build frameworks
Customer Pain Points

Test Manager
- Faces delays due to lack of tools to analyze better.
- Lacks visibility into “live” production metrics.
- Incurs high SW development costs and technical risk

System Engineer
- Too much time trying to troubleshoot and sell-off test data.
- Has to write one-off scripts to analyze and troubleshoot data.
- Hard to define and iterate sequences, limits, and parameters.

Test Developer
- Needlessly builds test “infrastructure” software.
- Struggles to adapt code for changing requirements.

Test Operator
- Lacks tools to do first level troubleshooting.
- Unnecessary manual work leads to errors.
**Common Causes**

**Technology Outgrowth**
- Still using systems that have been outgrown by newer complexity and test volume.
- Requires more manual labor to adapt new needs into older systems.
- Repetitive + arduous work = costly errors + delays

**Hardware Coupling**
- Tight coupling between hardware and test code
- Hardware obsolescence or substitution requires costly development and validation effort.
- Systems are subject to iteration due to driver updates.
- Barriers to using better or cheaper instruments.

**Tedious Data Analysis**
- Hard to analyze data in files
- Lacking flexibility with rigid file formats, data structures, or databases.
- Requires additional software or processing for reports and sell-off – custom SW, tools, Excel scripts
- Time consuming troubleshooting, down-time

**Duplication**
- In house systems each have their own “frameworks” – incompatible code across projects
- Inability to re-use means more duplicate work
- Duplicate work = $$ and Schedule Risk
- Configuration management, support, maintenance problems
Hidden Costs of Homegrown Test Software

- Test Executive
- Drivers
- Customer Test Code
- Data Storage
- Analytical Tools
- Reports
- Monitoring/Logging
- Virtual Instruments
Solution
Solution

Verifide is a complete, scalable, and modular software platform for use across the entire testing lifecycle

**TEST PLATFORM** – Scalable software foundation of your test systems from bench test to enterprise volume – will not be outgrown

**PRODUCTS** – Generic design that can handle testing for any product type – avoid duplication.

**DATA** – Single data solution for all your test systems – better and faster data analysis

**EQUIPMENT** – Modular and scalable services architecture to decouple hardware interfaces.

**DEVELOPMENT** – Develop in the language of proficiency - LabVIEW, .NET, Python, Matlab etc.
## Solution

### Test Automation

- **Services Framework** – Modular and scalable services framework for hosting instrument drivers, tests, or other code
- **Monitoring and safety** – Framework and tools for logging and data monitoring, with automatic event and alarm handling
- **Development** – Rich API, supports LabVIEW™, .NET, Python and more
- **Reqs and Specs** – Efficient, table based test sequences, limits, and parameters - supports Excel™

### Test Data Analysis

- **Data Storage** – Dynamic database technology that eliminates need for data management code.
- **Data Analysis** – Rich and interactive tools to visualize, analyze, and sell-off test data.
- **Metrics and Yield** – Tools to view metrics, data distributions, and perform post-test limit check and yield analysis.
- **Reports** – Report generation with no coding
## Benefits

<table>
<thead>
<tr>
<th>BENEFITS</th>
<th>RESULT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IMMEDIATE:</strong></td>
<td>Build test system software quicker.</td>
</tr>
<tr>
<td><strong>SHORT TERM:</strong></td>
<td>Better data analysis and disposition.</td>
</tr>
<tr>
<td><strong>FUTURE:</strong></td>
<td>Standardize across projects for reuse. Easier maintenance with modular architecture.</td>
</tr>
</tbody>
</table>
Compatibility

DEVELOPMENT ENVIRONMENTS

TEST EXECUTIVES

Develop in your environment of choice, use Verifide API for data storage, logging, or other services.

Use existing front end applications to execute tests in production or end-of-line scenarios.
Boeing has been an adopter since 2006.

We have amassed a base of **over 500 users** at Boeing who have used our software. Test related roles range from Test Developers, Systems Engineers, Test Operators and Production Managers.

The software was adopted as a platform for new systems including:

- Satellite RF Payload Test Systems
- In-Orbit Test Systems
- Calibration Systems
- Flight Electronics Unit Test Systems
- PCB/Card Test Systems
- Command and Control Systems
Features
Test data is different; typical IT data solutions do not suffice:

- File based systems are flexible and simple but do not provide management, search, querying, or analytics.
- Databases allow analysis and querying but are not flexible, require database administrator skills and effort, and can perform poorly.

Verifide’s platform is powered by our **Dynamic Database** technology for an out-of-the-box data solution:

- Flexibility of flat files but with the power of a database
- The ability to store rich data like tables, arrays, and images.
- The power to search, access, and analyze test data locally or from a remote computer
- Requires NO database design or administration because the database is built dynamically as data is stored to it.
Organize your data. Make it easier to find and analyze with hierarchies.

Dynamic Database

No database administrator needed. Database structures are built on-the-fly as tests store data.

Database designs, builds, and tunes itself – automatically, as data is stored!
Unified interface and sets of tools to analyze data across all your systems and data formats.

Store data using the API or import from your own flat files etc. using our example code.
Data Analysis
Bring visibility to your data

PROBLEM:

Without proper tools, test data is hard to make sense of:

• Trying to get high-level visibility into test data is difficult
• Mining data for trend analysis requires lots of manual effort
• Tools like Excel provide basic charts but no interactivity or ability to drill down.
• Reporting tools are expensive, not flexible, and often require coding.

SOLUTION:

System engineers and test engineers can use our suite of interactive tools to:
• Visualize data and identify trends
• Drill down
• Compare/overlay test data.
• Generate reports from tests with no code

Get access from anywhere with our web-based application as well as our desktop client.
Data Analysis Tools

Report generation with no coding. Design your templates and the software creates the report automatically.

Drill down into Raw data for a test.
Use 2D trending to overlay plot data to detect excursions in multi-dimensional data.

Extract trend data across test runs to correlate data and spot problems.

DATA TRENDING

VERIFIED
Limit check data AFTER testing for accurate yield. Even apply limits to data from years ago.

Load one or multiple limit sets to test hypothetical scenarios.
Managers and System Engineers do not have access to high level information from test systems:

- Hard to identify problematic tests and wasted test time
- Can’t predict future costs without knowing current performance
- No visibility into test system utilization, data distribution etc.
- Hard to know variations and distributions of measured data – results in failures in the field, inefficient processes.
- Limits are hard to nail down – results in retests and unnecessary diagnostics and troubleshooting

**SOLUTION:**

Verifide’s data management allows you to slice and analyze both process-level and measurement data using the same tool.

- Create pivot analysis tables and metrics on selected data sets
- Visualize data as histograms, distribution plots, pie charts etc.
- Drag and drop fields to create custom slices of data
- Post process test data for limits – do what-if scenarios to fine tune limits and process controls.
Summary Reports

Create your own summary reports and view them from the Dashboard

Data Distribution Plots

Yield Analysis

Pareto Reports
Statistical Process Control

Easily create statistical process analysis reports on any data set with a single button click.
PROBLEM:

Most test organizations incur high start-up costs when building or iterating systems.

- New system requirements spin off long infrastructure projects
- Language restrictions make infrastructure hard to implement
- Negligible reuse of tests, drivers, and “infrastructure” from other systems.
- Unstable code at bottom layers of the architecture results in unstable systems

SOLUTION:

Verifide provides powerful support for the developer with infrastructure to make your life easy:

- Use LabVIEW, MATLAB, ATEasy, VEE, C#, .NET, or Python
- Build distributed systems using our services framework and messaging infrastructure
- Developer-focused and full IDE support
- Full API for logging, monitoring, alarms, events, sequencing, and data management
Flexible Development

Use sequencers like TestStand or use Verifide’s own table-driven sequencer.

Write tests in your language of choice & start with sample code and how-to videos.
Easily troubleshoot with manual command execution.

Manage your test system assets.
The testing process subjects your device to a range of conditions in order to evaluate its performance. But many of these conditions can also damage your device if not careful.

- Device safety is dependent on operator intervention
- Systems lack tools to monitor for anomalous conditions
- Lack of traceability and activity records result in longer troubleshooting and higher down-time

**PROBLEM:**

**SOLUTION:**

Verifide provides a complete framework for the test runtime:

- Automatic alarms on key information with timeouts
- Monitoring of system parameters (temperature, current etc)
- System logging for traceability with log data mining utilities
- Automatic safety shutdown under unsafe conditions
- Big-Data stream handling (telemetry etc)
Automate safety with our events and alarm framework.

Be alerted on critical data limits with parallel monitoring.

Strip Charts and Event Handling
Troubleshoot faster with log snapshots

### Activity Logs

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Source</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>EventServer</td>
<td>Xml comments file loaded from C:\Source\Release\binaries\</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>EventServer</td>
<td>ADDED Instrument Successfully</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>EventServer</td>
<td>DONE Loading Instruments</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>SystemController</td>
<td>status=system.state Clear</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>SystemController</td>
<td>Running (1 of 1 Loaded, 0 Failed)</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>Server2.awa:9176</td>
<td>Loading data monitor server &lt;DataMonitor&gt;</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>SystemController</td>
<td>Running (1 of 1 Loaded, 0 Failed)</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>SystemController</td>
<td>demo_sequence Running</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>SystemController</td>
<td>Running (1 of 1 Loaded, 0 Failed)</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>SystemController</td>
<td>EVENT ID=1 CHART1 Warning HIGH limit failed. Data value 50.25 &gt; 55.0</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>SystemController</td>
<td>EVENT ID=2 CHART1 Critical HIGH limit failed. Data value 67.24 &gt; 75.0</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>SystemController</td>
<td>EVENT ID=3 CHART1 Damage HIGH limit failed. Data value 70.5 &gt; 80.0</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>SystemController</td>
<td>EVENT EXPIRED: ID=1 CHART1 Damage HIGH limit failed. Data value 112.200 &gt; 98.0</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>SystemController</td>
<td>TRIGGERED STATE CHANGE: ID=1 CHART1 Damage HIGH limit failed. Data value 115.29 &gt; 98.0</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>SystemController</td>
<td>ALARM: Urgent Priority Event Was Not Accepted Within 3 Seconds. Event ID=1</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>SystemController</td>
<td>SETTING STATE TO: Stop, User=EventServer</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>DriverServer</td>
<td>DriverServer State Set to Stop</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>EventServer</td>
<td>NOTIFIED AND SET SERVER: DriverServer to Stop state successfully</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>EventServer</td>
<td>NOTIFIED SERVER: TextServer of Stop state successfully</td>
</tr>
<tr>
<td>Feb 29 16</td>
<td>21:44:67.104</td>
<td>EventServer</td>
<td>NOTIFIED SERVER: StationServer of Stop state successfully</td>
</tr>
</tbody>
</table>
Quickly identify issues with filtering and log mining utilities.
Devices are getting more complex and test volumes are increasing. At the same time, modern time to market expectations are subjecting test schedules to more pressure.

• Existing test executives are being stretched beyond their capabilities and resulting in excessive labor.
• Setting up sequences, parameters, and limits is complex and very hard to iterate.
• System engineers rely on test organization to translate and implement parameters and limits.

PROBLEM:

SOLUTION:

Our table-driven test approach cuts development and setup effort and allows reuse:

• Use tables for sequences, parameters, and limits – also create and edit in Excel.
• System engineers can define and release updates themselves.
• Parameter resolution technology eliminates repetitive data entry.
Table-Based Test Sequencer

Better test planning with hierarchical sequence grouping

Easy to follow, row and column based test sequencing
Parameter Resolution

- Reuse parameter setups with modular files
- Reduce data entry with matching criteria
- Easily add/remove input parameters with table columns
For live demonstration videos and to learn more, visit us at:

www.verifide.com

Reach out to us:

Verifide Technologies, Inc.
6050 Hellyer Ave, #150
San Jose, CA 95138

1.408.834.8713
hello@verifide.com
Key Questions
QUESTION: I’m already using LabVIEW/TestStand/VEE/ATEasy. Why do I need your product?

ANSWER: Chances are you’re getting by, but having delays and difficulties with managing data and iterations in design or requirements. You’re also likely performing lots of manual labor to setup systems, analyze data, and troubleshooting. Verifide provides you with the platform you need to solve this, and you don’t have to stop using LabVIEW/TestStand/VEE/ATEasy.

QUESTION: We do our own test software development, why should I use your tools?

ANSWER: No organization has the time and budget to write a full set of tools for data storage, data analysis, logging, etc. Those that try, have to write and maintain an expensive set of code. Those that do without, are missing the ability to deeply diagnose, disposition, and report on testing. With Verifide, you still write your own test code to test your system, of course.
I already have a bunch of data that I want to analyze. Can I still use your tools?

Yes, we provide a programmatic interface with sample code to be able to import data into our Dynamic Database. The parsing of your data is the main effort, importing the data into our database is the easy part.

You can then gradually migrate existing test systems to store data directly into our database without needing a separate import step.
**Key Questions**

**QUESTION:** Am I going to be forced to come to you every time I need to change something?

**ANSWER:** No. Our modular services framework allows you to plug-and-play your own system specific functionality as components. You retain all intellectual property and control of your test code and any other components such as DUT interface libraries.

**QUESTION:** I think this will take too much effort on our part to integrate/use this product.

**ANSWER:** We are not asking anyone to throw away their investments in existing systems. There are immediate, medium, and long term benefits to the product. You can chose to use some portions of the product without others. Using just our data management, for example, will give you the highest ROI with least effort since it does not affect your existing hardware, test code, or sequences.
Q: How does this talk to the hardware?
A: You write wrappers for existing interfaces and plug them into our services framework.

Q: What’s different from TestStand?
A: You can use test stand with our platform. Our platform goes beyond a test executive and has tools for data analysis, metrics, logging, strip charting.

Q: How is the licensing handled?
A: Flexible options: perpetual, subscription, floating, and trial.

Q: Am I locked into your data format?
A: No. You can freely use formats like ASCII, Excel, XML, JSON.

Q: Can I only use some pieces of this platform?
A: Sure, you can use your existing front end and just use the data storage and analytical tools.
Q: Do I need to install software to gain access to analysis and metrics?
A: Verifide additionally sells a web dashboard and analysis tool which can do 95%+ of what the installed desktop client can.

Q: I don’t want to learn a special language.
A: Verifide allows you to develop with LabView, VEE, ATEasy, Python, C++, C#, or any .Net language.

Q: We don’t need all these features.
A: Verifide is modular and we have licenses structured for the modules you need.