Automation Hooks Architecture Trade Study for Flexible Test Orchestration

Chatwin A. Lansdowne, John R. Maclean, *Members, IEEE*
Frank J. Graffagnino, Patrick A. McCartney

Comment [A1]: Members IEEE?

***Abstract*—We describe the conclusions of a technology and communities survey supported by concurrent and follow-on proof-of-concept prototyping to evaluate feasibility of defining a durable, versatile, reliable, visible software interface to support strategic modularization of test software development. The objective is that test sets and support software with diverse origins, ages, and abilities can be reliably integrated into test configurations that assemble and tear down and reassemble with scalable complexity in order to conduct both parametric tests and monitored trial runs. The resulting approach is based on integration of three recognized technologies that are currently gaining acceptance within the test industry and when combined provide a simple, open and scalable test orchestration architecture that addresses the objectives of the Automation Hooks task. The technologies are automated discovery using multicast DNS Zero Configuration Networking (zeroconf), commanding and data retrieval using resource-oriented Restful Web Services, and XML data transfer formats based on Automatic Test Markup Language (ATML). This open-source standards-based approach provides direct integration with existing commercial off-the-shelf (COTS) analysis software tools.**

***Index Terms*— Software standards, Test equipment, Test facilities, Testing, Software management, Software reusability**

I. Introduction

NASA's Constellation Program identified an opportunity to reduce out-year operating costs for system and sub-system integration test operations through automation-assisted test choreography and data orchestration. There are complimentary opportunities to improve scientific research and engineering development workflows.

Essentially, the opportunity is that even for run-once and investigative testing, COTS and even custom hardware is configured and monitored by users through keyboard-and-mouse software packages. If data from these ~~disparate~~heterogeneous modules could be harvested through a robust, open standard based infrastructure, the data products could be formed more quickly, accurately, and thoroughly, and results correlated more powerfully—by comparison with, say, having users type data from screens into spreadsheets or collect and transfer data files in an ad-hoc fashion.

The application scenarios anticipated are not high volume or highly repetitive. Automating development of the test procedures themselves from requirements is not a significant area of interest. Much of the potential for saving is related to discovery of module data requirements and aggregation of test data in integrated scenarios containing a changing assortment of highly complex and coupled modules.

There are many technical challenges to address, but first one must confront the organizational challenge in that the software modules available at a system integration test, for example, are of diverse origins and developed on heterogeneous platforms.

Having accomplished the aggregation of coincident observations, one can further imagine storing and restoring the configuration of the test bed using read/write interfaces, and ultimately it could be possible to repeat test sequences and overlay data.

Quiescent, continuous, and event-driven test cycles are anticipated. Scripted flows are presumed built on less structured fault-isolation or experimentation test flows.

The concept of operations imposes some constraints required to enable data correlation. These include time synchronization mechanisms and resources, data indexing, labeling data with metadata, and encouraging the use of widely understood self-describing data formats.

II. Criteria and Field of Choices

A concept of operations was proposed, and then distilled down to a set of "guiding principles" which could be used for evaluating different approaches. These principles included:

Non-proprietary, with multiple vendors. A proprietary or single-vendor interface could not achieve universal penetration into varied developments and could present a single-point-of-failure risk to the Program.

Widespread usage, with active user communities. Our intention was not to reinvent the interface and associated toolsets, but rather to find and adopt (adapt) already widely supported technologies.

Supported in the Test industry. Interfaces with existing support in NASA, DoD, and consumer communities and test COTS products were given affirmative weight.

Multiple sources of ready development tools. Software

Manuscript received June 1, 2010. This work was performed in NASA Johnson Space Center's Avionics Systems Division in support of the Constellation Program.

C. A. Lansdowne is with the National Aeronautics and Space Administration, Houston, TX 77058 USA (phone: 281-483-1265; fax: 281-483-6297; e-mail: chatwin.lansdowne@ nasa.gov).

J. R. Maclean is with METECS, Houston, TX 77058 USA (phone 281-483-3265; e-mail: john.r.maclean@nasa.gov).

interfaces supported by a family of open source tools provide rapid deployment.

Language and OS independent. Interfaces that are tied to specific operating systems or development environments only solve part of the problem, and are vulnerable to accelerated obsolescence.

Having described what we were seeking, we surveyed test communities at NASA, DoD, and in industry, and also considered plug-and-play consumer interfaces. We considered that our software elements could use simulation interfaces, or instrumentation interfaces, or web services interfaces. Fundamentally the difference between requirements for a test software interface and a simulation software interface is that the modules do not need to interact.

III. Summary of Study Observations

A. Existing End-to-End Infrastructures

Several existing end-to-end simulation and test infrastructures were investigated in an attempt to find an out-of-the box capability that could be used to meet the trade study criteria.

High Level Architecture (HLA) has been used in the Constellation program as an architecture for distributed dynamics simulations. It was examined briefly but it was quickly decided that the overhead associated with its run time infrastructure and simulation federate organization made it unattractive as a test orchestration infrastructure.

The Test and Training Enabling Architecture (TENA) is a DoD initiative aimed at distributed simulation and test applications. It is geared to supporting test ranges and facilities. TENA seemed to require middleware that appeared to be single-source. In addition, it is based on CORBA, an object-based messaging protocol that has been declining in popularity because of its complexity and historical difficulty penetrating firewalls. Interest in the wider community has shifted from CORBA and its competitor DCOM to Web Services which are discussed later.

B. Established Test Software Interfaces

A promising early candidate was the LXI interface, and ultimately we adopted several features of this interface. The interface was discarded because tools for development of LXI hosts were not readily available.

Investigation of the DoD Automatic Test Systems (ATS) Open Systems approach lead to interest in the Automatic Test Markup Language as a data format. Many of the approaches of the ATS Open Systems approach were compatible with our “guiding principles”. Virtual Instrument Software Architecture (VISA) and Interchangeable Virtual Instruments (IVI) technologies were determined to be too low-level for our goal and available drivers appeared to be limited to the Windows OS. In addition, these technologies did not appear to be widely used outside the Automatic Test Equipment industry.

NASA’s Constellation program was also developing an interface for avionics test orchestration, Software and Avionics Test Orchestration Command and Messaging (SATOCM). We did exchange observations with this group, and although there remain differences in emphasis both teams believed it would be possible to achieve convergence. This interface was designed to simplify test script-writing using Python. Our study was not able to adopt it because its current incarnation contains a complex command set delivered by a custom binary messaging mechanism.. ~~was rejected as isby our study because it violates many of our guiding principles.~~

C. Discovery Protocols

Universal Plug and Play (UPnP) was evaluated against Zeroconf. Both were strong candidates, but we perceive that UPnP is fading, and Zeroconf has existing heritage in the test community through its use in LXI.

D. Messaging Protocols

Several message oriented protocols and middleware APIs at several different levels of complexity were considered. Some, like Java Message Service (JMS) were not language neutral. Some like Advanced Message Queuing Protocol (AMQP) introduced complexity by solving problems we did not have. The Simple Object Access Protocol (SOAP) web services protocol was chosen for initial prototyping because it satisfied our evaluation criteria and fit well with ATML which was also of interest. There is a wide variety of tools and implementations available including many open-source packages. It is also widely used and accepted in many industries.

A functional prototype was implemented using SOAP. Many parts of the SOAP implementation, however, were found to be complex in the face of limited prototyping resources. For example, Web Services Description Language (WSDL) files were found to be complex to create and maintain. Different implementations of SOAP were found to be incompatible without detailed attention to configurations and options. No insurmountable problems were encountered but eventually a decision was made to prototype an alternative resource oriented or Representational State Transfer (RESTful) style of web services. The level of simplification, elegance and increased ease of implementation was so striking that ultimately when faced with building a prototype using limited resources we opted for the RESTful approach. This choice affected not just the messaging protocol but the overall architecture and division of responsibilities between test orchestration software and individual test set interfaces.

Many of RESTful features such as Uniform Interface, stateless server restrictions, and cacheable responses contributed to robustness and enhanced visibility of the test protocol. Also, the perspective on commanding test equipment changed from remote-procedure-call (RPC) based to resource based which was found to result in gains in elegance and simplicity.

E. Command Sets

Test execution interfaces have a long history of using verb-based command sets, including HP BASIC, ATLAS, and SCPI. NASA’s SATOCM command set was intended to simplify script-writing, and initially we planned to implement a subset of SATOCM commands.

The RESTful style architecture primarily uses a small subset of standard HTTP commands such as GET, PUT, POST, and

DELETE directly. The richness of the interface is then captured as resources that are manipulated using these standard HTTP commands. This approach replaces the requirement to create a traditional RPC-based set of commands with the requirement to design appropriate resources to represent required test concepts. In prototyping, the resource-based approach was found to result in a simpler and more transparent infrastructure.

The command and error message sets already provided by HTTP ~~are compact, powerful, and~~ understood by a large collection of off-the-shelf software. The command set is compact and powerful, and the error message set is rich. Security and data compression solutions are innate.

F. Data Interface Protocols

We evaluated the architecture of having software modules write directly to a designated database interface without an intermediary. Scalability and robustness were identified obstacles. To make a successful interface, a completed software module must be able to create its own tables and write data to them without further changes to the platform to accommodate different database vendors or other changes in database technology. JDBC was entertained as meeting this objective, but limits the usability of the module by requiring each software module to interface with Java. An ODBC driver approach was evaluated but required specialized software to be installed and maintained on each client. An ODBCbridge driver approach eliminates the client software issue, but introduces an issue with proprietary software and a sole-source provider. It was determined that this type of SQL-oriented middleware merely transfers the maintenance problem to another vendor who must then be required.

The solution that worked best in prototyping and met the goals of the study was to use the resource oriented interface to serve data. An unexpected side benefit was that data resources could be accessed by web-ready off-the-shelf software. For example, prototypes built on modular open source software have demonstrated that this interface is already natively accessible to web browsers and to Excel.

Data log requests are submitted by the orchestrator and each testset module makes locally buffered data accessible through a resource interface for that data log request. This approach allows data to be logged with arbitrary resolution. Tight coupling mechanisms such as data pushes, reflective memory, or messaging middleware are unnecessary because ~~and~~ alignment occurs after the observations are aggregated.

G. Data Formats

Software written by hardware engineers often will write data using comma separated value (CSV) or tab separated value (TSV) formats. These formats are easy to generate, widely supported by tools, and are decades old. There are, however, many format variations including how commas are handled within a data file. This can be particularly troublesome for countries that use commas as the decimal separator. In addition there are no recommendable approaches for incorporating meta-data. There are also some operating system differences.

Binary formats are very system-dependent, although they can be supplemented with descriptive XML metadata files.

The SQL statement format was also considered, but the availability of XML-enabled databases diminishes the appeal of this option. It was further identified that different database vendors have different interpretations of the SQL standard.

XML allows sufficient metadata to be included so that database tables can be automatically created, standardizes the ~~data~~date-time format, and allows further information like theory of operation (help-text) for a parameter to be captured. ATML provides an XML language that standardizes information exchange for many kinds of data and meta-data we are interested in capturing, and is also becoming represented in test industry products. An alternative schema, NASA Exploration Information Ontology Model (NExIOM), was discarded only because it has a limited following. The authors hope that NASA can participate in the further development of ATML.

IV. Conclusion

Combining RESTful principles with Zeroconf and ATML formed a powerful, versatile, rugged interface that met all of the study criteria. The combination was found to provide a simple, elegant, and easy to use infrastructure for test orchestration. Prototypes have already demonstrated connectivity with LabVIEW, NASA's Trick Simulation Development Environment, and the Engineering DOUG Graphics for Exploration (EDGE) software used for 3D graphics rendering in Constellation training and test facilities. Prototypes have been hosted in various distributions of the Linux operating system and in Windows XP and Vista. Distributed and co-hosted topologies have been demonstrated, and multiple copies of modules are distinguishable. The interface has been demonstrated with both simulations and hardware and has been used to orchestrate a distributed Orion abort-to-orbit test scenario using JSCs Avionics Integration Enivironment (AIE) facility and the Reconfigurable Cockpit Simulation Facility. It is being integrated with test hardware in the Kedallion avionics facility and the Electronics Systems Test Lab (ESTL) at Johnson Space Center. We currently rate this interface as Technology Readiness Level 4.

Acknowledgment

This endeavor required focusing many kinds of nonintersecting experts on a multi-faceted problem. We would like to thank Thomas Brain, Tom Smith and Chris Winton for their contributions to the software prototypes. Joan Zucha and Adam Schlesinger were sources of guidance and evaluation.

References

[1] ZeroConf Reference
[2] ATML Reference
[3] REST Phd Reference
[4] REST web services book reference

~~[1] G. O. Young, "Synthetic structure of industrial plastics (Book style with paper title and editor)," in *Plastics*, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64.~~

Formatted: Indent: Left: 0.25", No bullets or numbering

[2] W. K. Chen, *Linear Networks and Systems* (Book style). Belmont, CA: Wadsworth, 1993, pp. 123–135.
[3] H. Poor, *An Introduction to Signal Detection and Estimation*. New York: Springer-Verlag, 1985, ch. 4.
[4] B. Smith, "An approach to graphs of linear forms (Unpublished work style)," unpublished.
[5] E. H. Miller, "A note on reflector arrays (Periodical style—Accepted for publication)," *IEEE Trans. Antennas Propagat.*, to be published.
[6] J. Wang, "Fundamentals of erbium-doped fiber amplifiers arrays (Periodical style—Submitted for publication)," *IEEE J. Quantum Electron.*, submitted for publication.
[7] C. J. Kaufman, Rocky Mountain Research Lab., Boulder, CO, private communication, May 1995.
[8] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interfaces (Translation Journals style)," *IEEE Transl. J. Magn.Jpn.*, vol. 2, Aug. 1987, pp. 740–741 [*Dig. 9th Annu. Conf. Magnetics* Japan, 1982, p. 301].
[9] M. Young, *The Techincal Writers Handbook*. Mill Valley, CA: University Science, 1989.
[10] J. U. Duncombe, "Infrared navigation—Part I: An assessment of feasibility (Periodical style)," *IEEE Trans. Electron Devices*, vol. ED-11, pp. 34–39, Jan. 1959.
[11] S. Chen, B. Mulgrew, and P. M. Grant, "A clustering technique for digital communications channel equalization using radial basis function networks," *IEEE Trans. Neural Networks*, vol. 4, pp. 570–578, Jul. 1993.
[12] R. W. Lucky, "Automatic equalization for digital communication," *Bell Syst. Tech. J.*, vol. 44, no. 4, pp. 547–588, Apr. 1965.
[13] S. P. Bingulac, "On the compatibility of adaptive controllers (Published Conference Proceedings style)," in *Proc. 4th Annu. Allerton Conf. Circuits and Systems Theory*, New York, 1994, pp. 8–16.
[14] G. R. Faulhaber, "Design of service systems with priority reservation," in *Conf. Rec. 1995 IEEE Int. Conf. Communications*, pp. 3–8.
[15] W. D. Doyle, "Magnetization reversal in films with biaxial anisotropy," in *1987 Proc. INTERMAG Conf.*, pp. 2.2-1–2.2-6.
[16] G. W. Juette and L. E. Zeffanella, "Radio noise currents n short sections on bundle conductors (Presented Conference Paper style)," presented at the IEEE Summer power Meeting, Dallas, TX, Jun. 22–27, 1990, Paper 90 SM 690-0 PWRS.
[17] J. G. Kreifeldt, "An analysis of surface-detected EMG as an amplitude-modulated noise," presented at the 1989 Int. Conf. Medicine and Biological Engineering, Chicago, IL.
[18] J. Williams, "Narrow-band analyzer (Thesis or Dissertation style)," Ph.D. dissertation, Dept. Elect. Eng., Harvard Univ., Cambridge, MA, 1993.
[19] N. Kawasaki, "Parametric study of thermal and chemical nonequilibrium nozzle flow," M.S. thesis, Dept. Electron. Eng., Osaka Univ., Osaka, Japan, 1993.
[20] J. P. Wilkinson, "Nonlinear resonant circuit devices (Patent style)," U.S. Patent 3 624 12, July 16, 1990.
[21] *IEEE Criteria for Class IE Electric Systems* (Standards style), IEEE Standard 308, 1969.
[22] *Letter Symbols for Quantities*, ANSI Standard Y10.5-1968.
[23] R. E. Haskell and C. T. Case, "Transient signal propagation in lossless isotropic plasmas (Report style)," USAF Cambridge Res. Lab., Cambridge, MA Rep. ARCRL-66-234 (II), 1994, vol. 2.
[24] E. E. Reber, R. L. Michell, and C. J. Carter, "Oxygen absorption in the Earth's atmosphere," Aerospace Corp., Los Angeles, CA, Tech. Rep. TR-0200 (420-46)-3, Nov. 1988.
[25] (Handbook style) *Transmission Systems for Communications*, 3rd ed., Western Electric Co., Winston-Salem, NC, 1985, pp. 44–60.
[26] *Motorola Semiconductor Data Manual*, Motorola Semiconductor Products Inc., Phoenix, AZ, 1989.
[27] (Basic Book/Monograph Online Sources) J. K. Author. (year, month, day). *Title* (edition) [Type of medium]. Volume (issue). Available: http://www.(URL)
[28] J. Jones. (1991, May 10). Networks (2nd ed.) [Online]. Available: http://www.atm.com
[29] (Journal Online Sources style) K. Author. (year, month). Title. *Journal* [Type of medium]. Volume(issue), paging if given. Available: http://www.(URL)
[30] R. J. Vidmar. (1992, August). On the use of atmospheric plasmas as electromagnetic reflectors. *IEEE Trans. Plasma Sci.* [Online]. *21(3)*. pp. 876–880. Available: http://www.halcyon.com/pub/journals/21ps03-vidmar

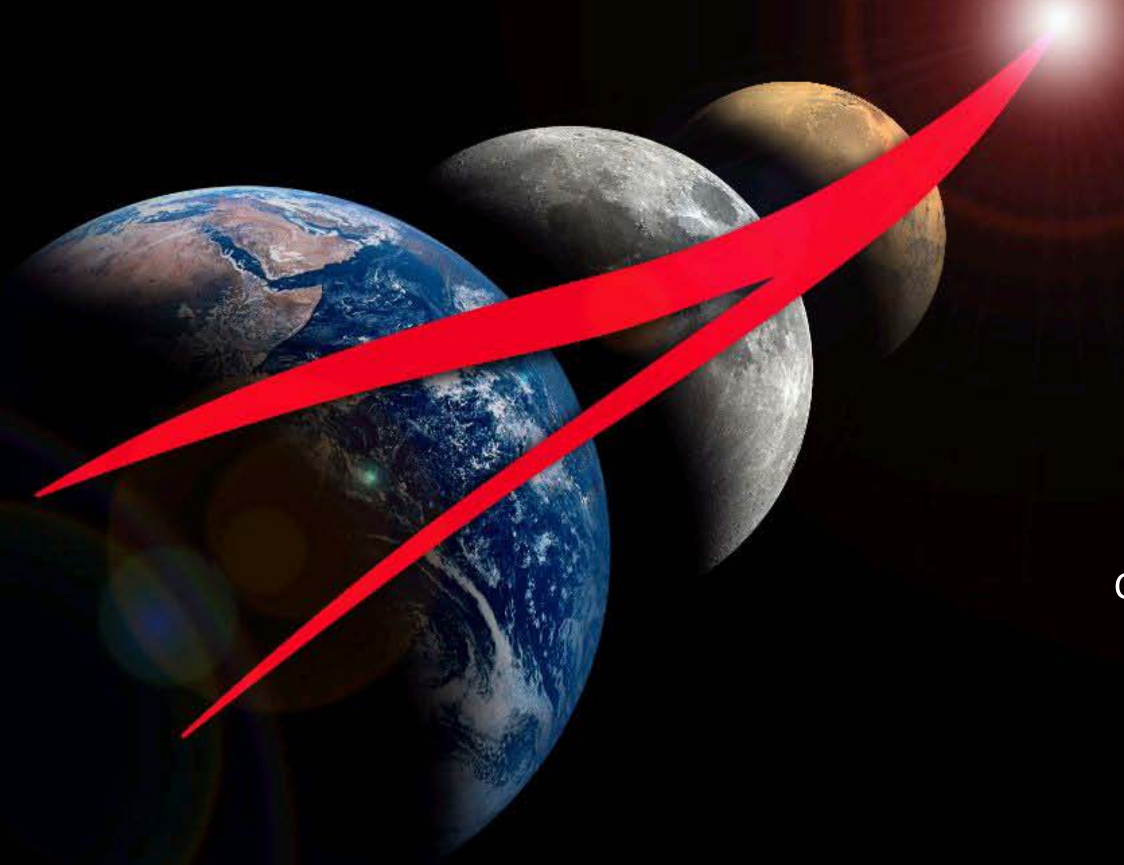
Chatwin A. Lansdowne (M'87) received a MS in electrical engineering from The Wichita State University, Wichita, Kansas, USA, 1991.
He is presently a Test Director in the Electronic Systems Test Laboratory at the NASA Johnson Space Center in Houston, Texas USA.

Comment [A2]: Biographies? Optional; think these are more common in journal articles

Formatted: Normal

Automation Hooks Architecture

Trade Study for Flexible Test Orchestration



Chatwin Lansdowne

John MacLean

Frank Graffagnino

Patrick McCartney

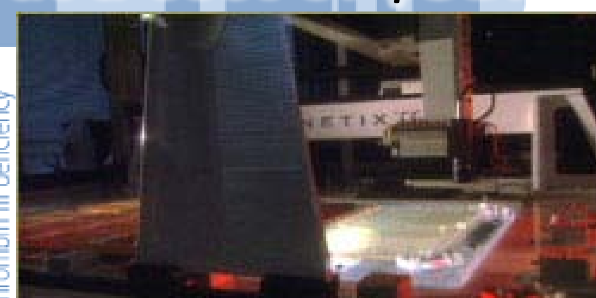
9/13/2010

CONSTELLATION

Software and Avionics Integration Office (SAVIO)

The Craig Venter Story

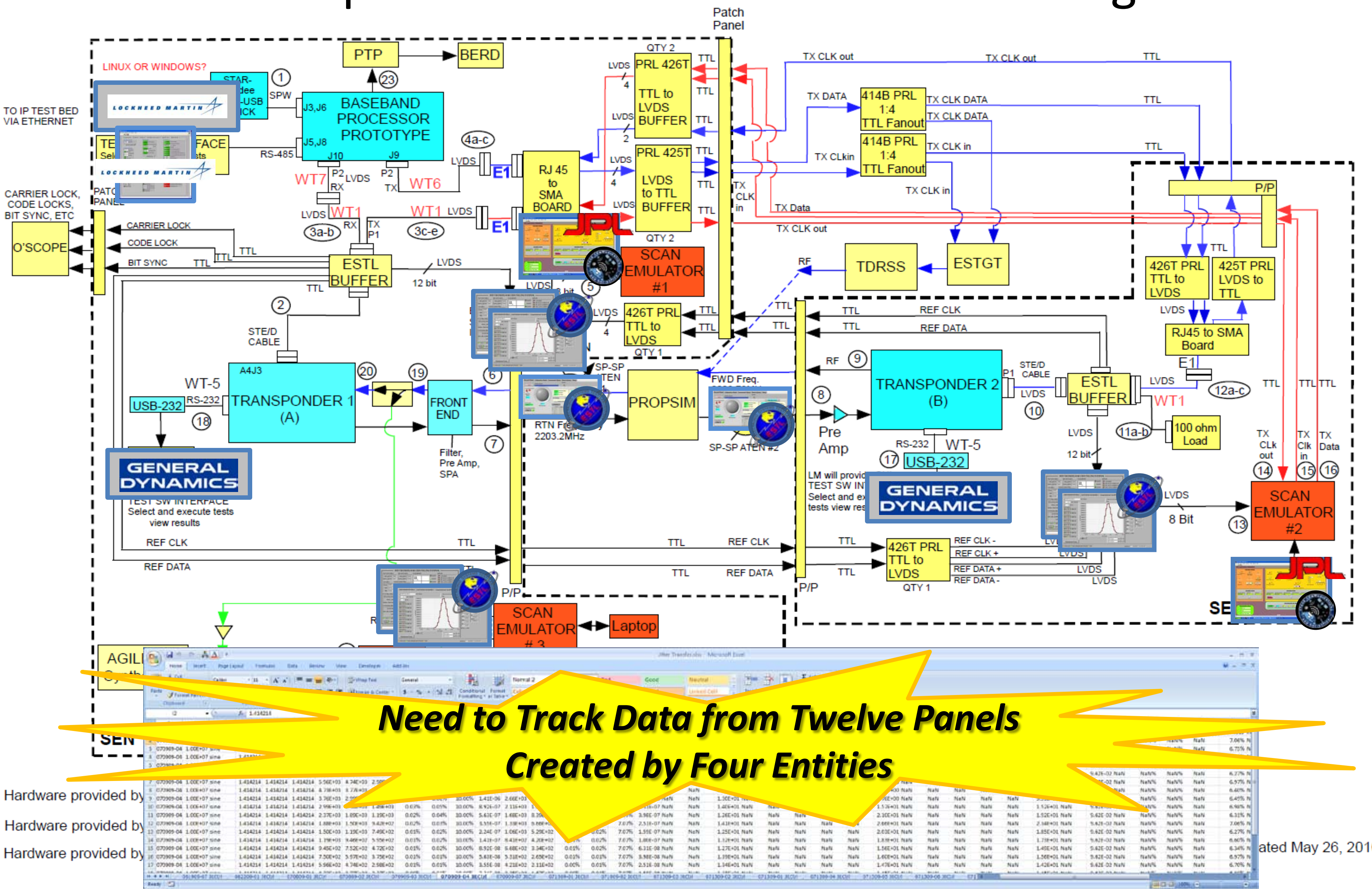
- NIH began the Human Genome Project in 1993 and expected to finish in 2003 (10 years) at a cost of $300B.
- Dr. Venter left NIH in 1998, founded Celera Genomics, and completed sequencing of the human genome in 2000 (2 years) at a cost of $3B.



Don’t Rush to Directly Solve the Problem…
Create a Machine to Solve the Problem

ESTL Use-Case Example

Orion Transponder – Baseband Processor Integration



Hardware provided by

Hardware provided by

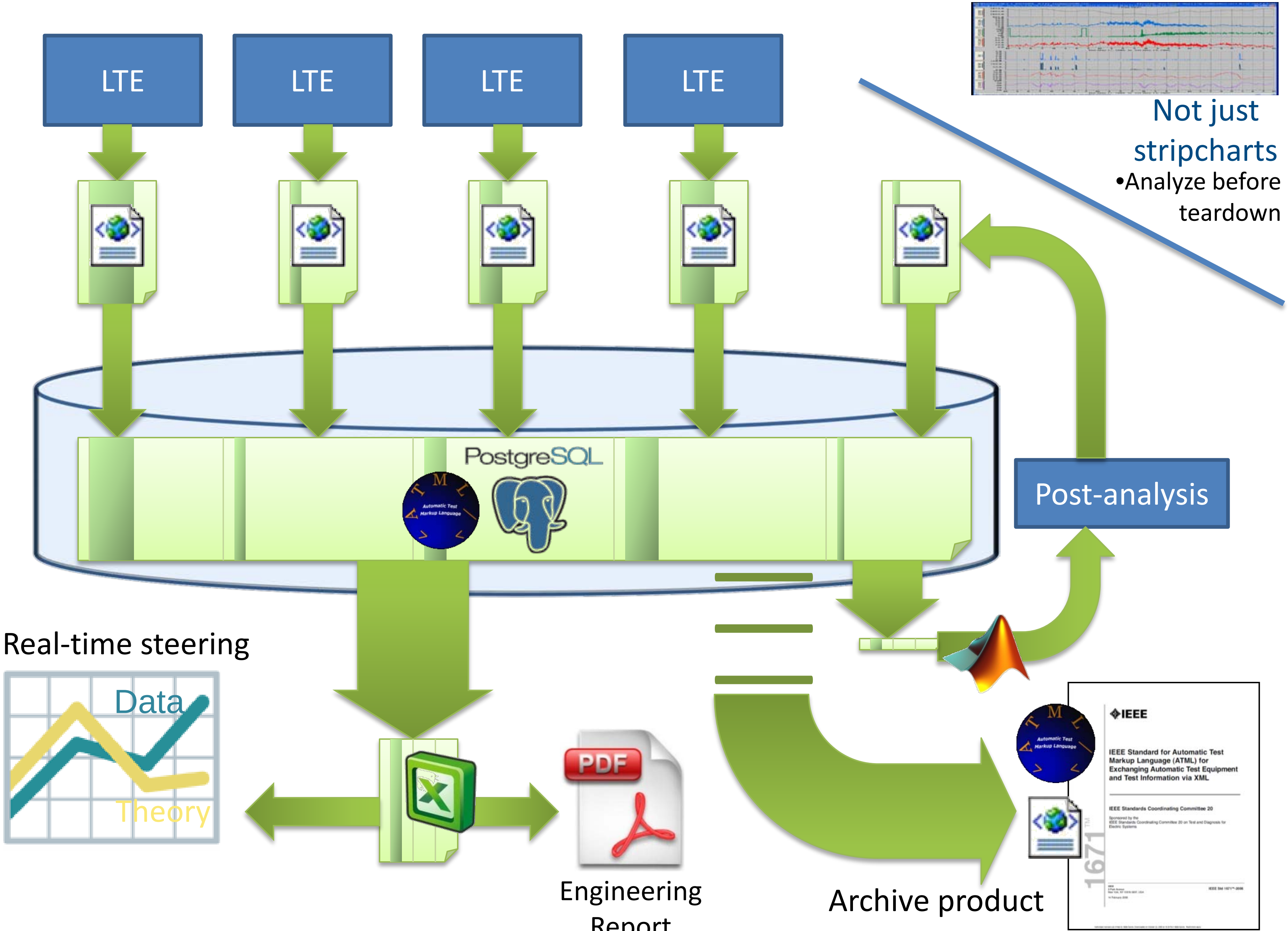
Hardware provided by

Adding Automation without Adding Infrastructure

IT support scales ***UP****,* but can IT support scale DOWN?

IT infrastructure can scale ***UP****,* but can IT infrastructure scale DOWN?

Notional AHA Prototype Architecture: Data Product Flow



Basic Functional Requirements

- Discovery
  - TFDM chooses LTE's required for test configuration
  - LTE resources/command set available to TFDM upon discovery
- Initialize, configure or return LTE's to known configurations
- Provide LTE status to TFDM directly from LTE or through a database
- Middleware to buffer LTE's from changes in database vendors/versions
- Support parametric as well as time-based test objectives
- Don't preclude
  - multiple tests on the same network
  - other test equipment on same network
  - ability of LTE's to join and leave test config as required by test conductor
  - Multiple databases

Technology Survey and Trade Study

- Surveyed NASA, Test COTS, DoD, and Consumer communities for viable approaches
- Down-selected based on "guiding principles"

Test Technology Survey

- Performed a survey (web, interviews, tour) to identify methods/protocols being used for discovery and test control in relevant test communities.
- Looked at two NASA examples
  - DSIL
  - ISIL
- Looked at two DoD examples
  - TENA
  - DoD ATS
- Looked at Commercial Products
  - NI Labview and Test Stand Test Software
  - IXIA IP Test Automation

What did the AHA study conclude?

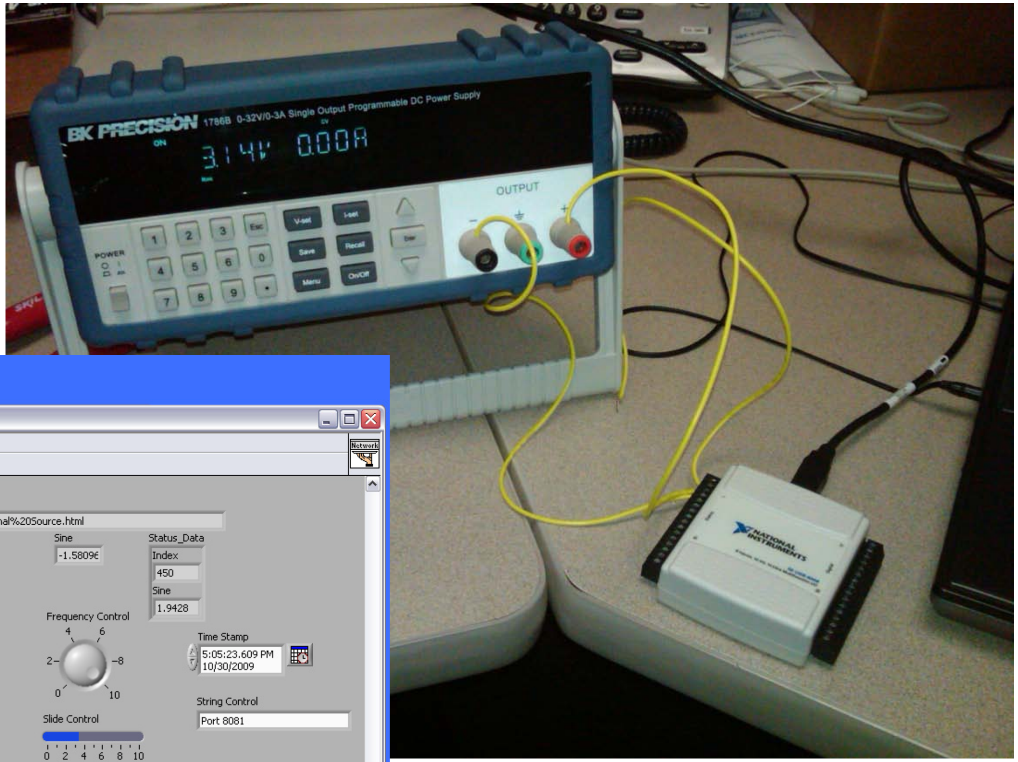
- **Discovery**: advertised resources using Zeroconf (DNS-SD, mDNS, link-local)
- **Data Representation**: XML and some ATML
- **Control/Status**: REST architecture over HTTP

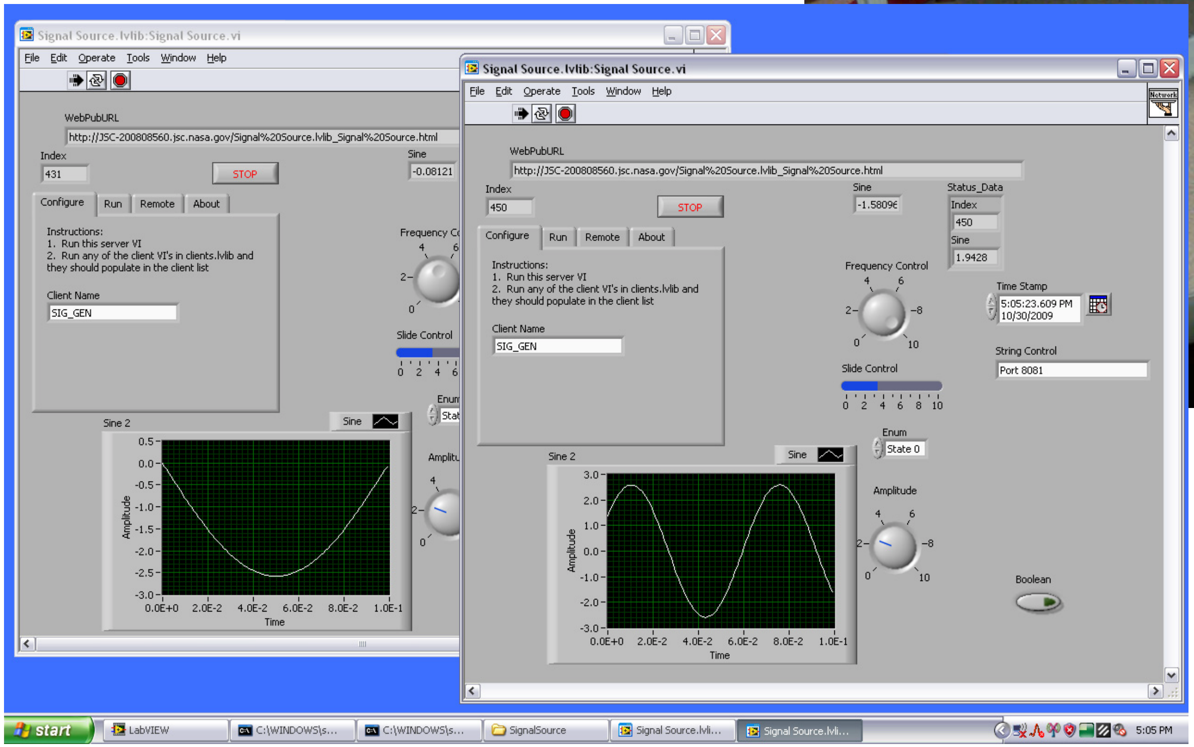
Discovery

- Assembling the automation can’t take extra time, on top of assembling the testbed, so simple plug-and-play discovery is highly desirable.
- Zeroconf has heritage in LXI, LabVIEW, and consumer products.
- Modules don’t need to be responsible for tunneling discovery through firewalls, this can be handled using proxy method or unicast.

Dynamic Port Assignments

- **Plug-and-Play discovery** enables dynamic IP address and Port assignments.





- If a computer goes down, give it to a tech and use a different one!

The XML File Format

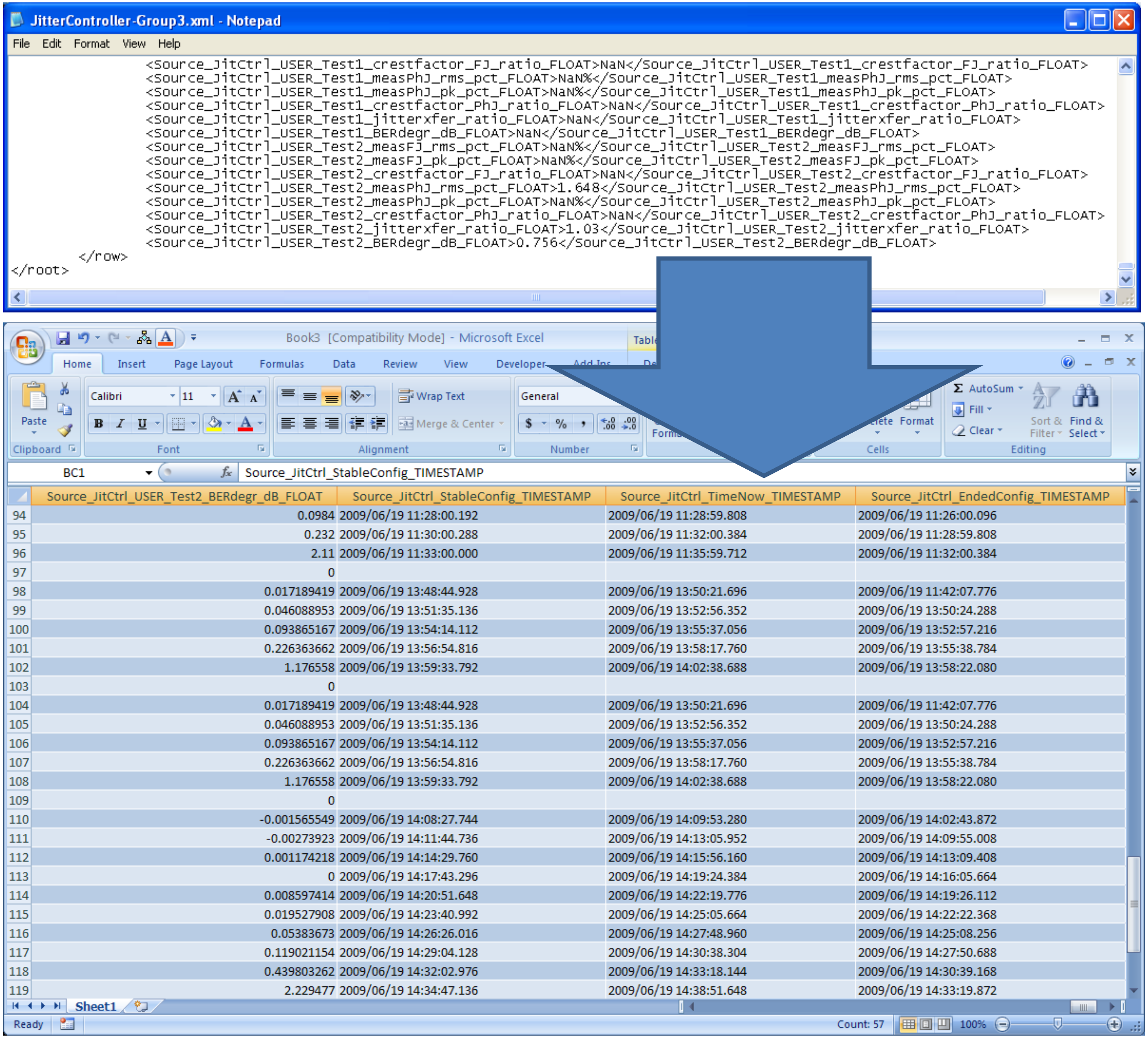
The Good

- Provides a method for thorough description: good naming, units, representation
- Record presentation format is defined separately (XSL style sheet)
- Provides many inherently standard data representations, including timestamps
- Provides a method for representing hierarchical data
- The file format is standardized
- Adoption is spreading

The Bad

- Data cannot be **appended** to an XML file, the entire file should be parsed and rewritten and the data must be **inserted**.
- “Free” tools now are crude
  - MS XML Notepad, Excel
- Good tools now are not free; JSC/EA doesn’t have them
  - XML Fox $280/seat, $1200/corp
  - Data Direct’s Stylus Studio ~$25k
- It’s not as “human readable” as you may have heard

Opening an XML file in Excel



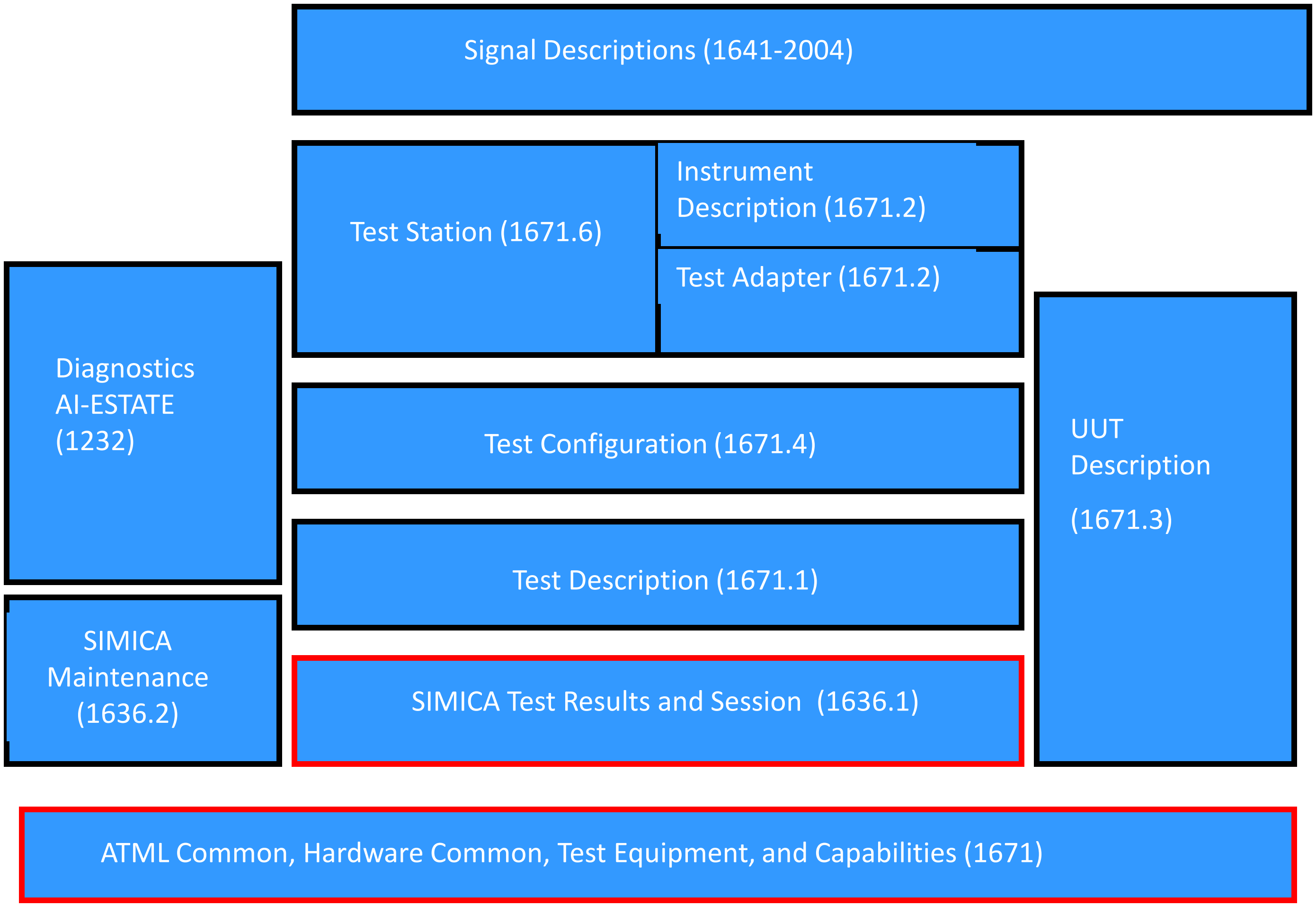

```
        <Source_JitCtrl_USER_Test1_crestfactor_FJ_ratio_FLOAT>NaN</Source_JitCtrl_USER_Test1_crestfactor_FJ_ratio_FLOAT>
        <Source_JitCtrl_USER_Test1_measPhJ_rms_pct_FLOAT>NaN%</Source_JitCtrl_USER_Test1_measPhJ_rms_pct_FLOAT>
        <Source_JitCtrl_USER_Test1_measPhJ_pk_pct_FLOAT>NaN%</Source_JitCtrl_USER_Test1_measPhJ_pk_pct_FLOAT>
        <Source_JitCtrl_USER_Test1_crestfactor_PhJ_ratio_FLOAT>NaN</Source_JitCtrl_USER_Test1_crestfactor_PhJ_ratio_FLOAT>
        <Source_JitCtrl_USER_Test1_jitterxfer_ratio_FLOAT>NaN</Source_JitCtrl_USER_Test1_jitterxfer_ratio_FLOAT>
        <Source_JitCtrl_USER_Test1_BERdegr_dB_FLOAT>NaN</Source_JitCtrl_USER_Test1_BERdegr_dB_FLOAT>
        <Source_JitCtrl_USER_Test2_measFJ_rms_pct_FLOAT>NaN%</Source_JitCtrl_USER_Test2_measFJ_rms_pct_FLOAT>
        <Source_JitCtrl_USER_Test2_measFJ_pk_pct_FLOAT>NaN%</Source_JitCtrl_USER_Test2_measFJ_pk_pct_FLOAT>
        <Source_JitCtrl_USER_Test2_crestfactor_FJ_ratio_FLOAT>NaN</Source_JitCtrl_USER_Test2_crestfactor_FJ_ratio_FLOAT>
        <Source_JitCtrl_USER_Test2_measPhJ_rms_pct_FLOAT>1.648</Source_JitCtrl_USER_Test2_measPhJ_rms_pct_FLOAT>
        <Source_JitCtrl_USER_Test2_measPhJ_pk_pct_FLOAT>NaN%</Source_JitCtrl_USER_Test2_measPhJ_pk_pct_FLOAT>
        <Source_JitCtrl_USER_Test2_crestfactor_PhJ_ratio_FLOAT>NaN</Source_JitCtrl_USER_Test2_crestfactor_PhJ_ratio_FLOAT>
        <Source_JitCtrl_USER_Test2_jitterxfer_ratio_FLOAT>1.03</Source_JitCtrl_USER_Test2_jitterxfer_ratio_FLOAT>
        <Source_JitCtrl_USER_Test2_BERdegr_dB_FLOAT>0.756</Source_JitCtrl_USER_Test2_BERdegr_dB_FLOAT>
    </row>
</root>
```

	Source_JitCtrl_USER_Test2_BERdegr_dB_FLOAT	Source_JitCtrl_StableConfig_TIMESTAMP	Source_JitCtrl_TimeNow_TIMESTAMP	Source_JitCtrl_EndedConfig_TIMESTAMP
94	0.0984	2009/06/19 11:28:00.192	2009/06/19 11:28:59.808	2009/06/19 11:26:00.096
95	0.232	2009/06/19 11:30:00.288	2009/06/19 11:32:00.384	2009/06/19 11:28:59.808
96	2.11	2009/06/19 11:33:00.000	2009/06/19 11:35:59.712	2009/06/19 11:32:00.384
97	0			
98	0.017189419	2009/06/19 13:48:44.928	2009/06/19 13:50:21.696	2009/06/19 11:42:07.776
99	0.046088953	2009/06/19 13:51:35.136	2009/06/19 13:52:56.352	2009/06/19 13:50:24.288
100	0.093865167	2009/06/19 13:54:14.112	2009/06/19 13:55:37.056	2009/06/19 13:52:57.216
101	0.226363662	2009/06/19 13:56:54.816	2009/06/19 13:58:17.760	2009/06/19 13:55:38.784
102	1.176558	2009/06/19 13:59:33.792	2009/06/19 14:02:38.688	2009/06/19 13:58:22.080
103	0			
104	0.017189419	2009/06/19 13:48:44.928	2009/06/19 13:50:21.696	2009/06/19 11:42:07.776
105	0.046088953	2009/06/19 13:51:35.136	2009/06/19 13:52:56.352	2009/06/19 13:50:24.288
106	0.093865167	2009/06/19 13:54:14.112	2009/06/19 13:55:37.056	2009/06/19 13:52:57.216
107	0.226363662	2009/06/19 13:56:54.816	2009/06/19 13:58:17.760	2009/06/19 13:55:38.784
108	1.176558	2009/06/19 13:59:33.792	2009/06/19 14:02:38.688	2009/06/19 13:58:22.080
109	0			
110	-0.001565549	2009/06/19 14:08:27.744	2009/06/19 14:09:53.280	2009/06/19 14:02:43.872
111	-0.00273923	2009/06/19 14:11:44.736	2009/06/19 14:13:05.952	2009/06/19 14:09:55.008
112	0.001174218	2009/06/19 14:14:29.760	2009/06/19 14:15:56.160	2009/06/19 14:13:09.408
113	0	2009/06/19 14:17:43.296	2009/06/19 14:19:24.384	2009/06/19 14:16:05.664
114	0.008597414	2009/06/19 14:20:51.648	2009/06/19 14:22:19.776	2009/06/19 14:19:26.112
115	0.019527908	2009/06/19 14:23:40.992	2009/06/19 14:25:05.664	2009/06/19 14:22:22.368
116	0.05383673	2009/06/19 14:26:26.016	2009/06/19 14:27:48.960	2009/06/19 14:25:08.256
117	0.119021154	2009/06/19 14:29:04.128	2009/06/19 14:30:38.304	2009/06/19 14:27:50.688
118	0.439803262	2009/06/19 14:32:02.976	2009/06/19 14:33:18.144	2009/06/19 14:30:39.168
119	2.229477	2009/06/19 14:34:47.136	2009/06/19 14:38:51.648	2009/06/19 14:33:19.872

- Excel 2007 opens the XML file.
- DateTimes display correctly and can be also used directly in calculations.
- Columns are reordered, not clear what the order is.
- Table sort operations actually execute as column sort.
- Graphing data in the table is an indirect, text-based operation.

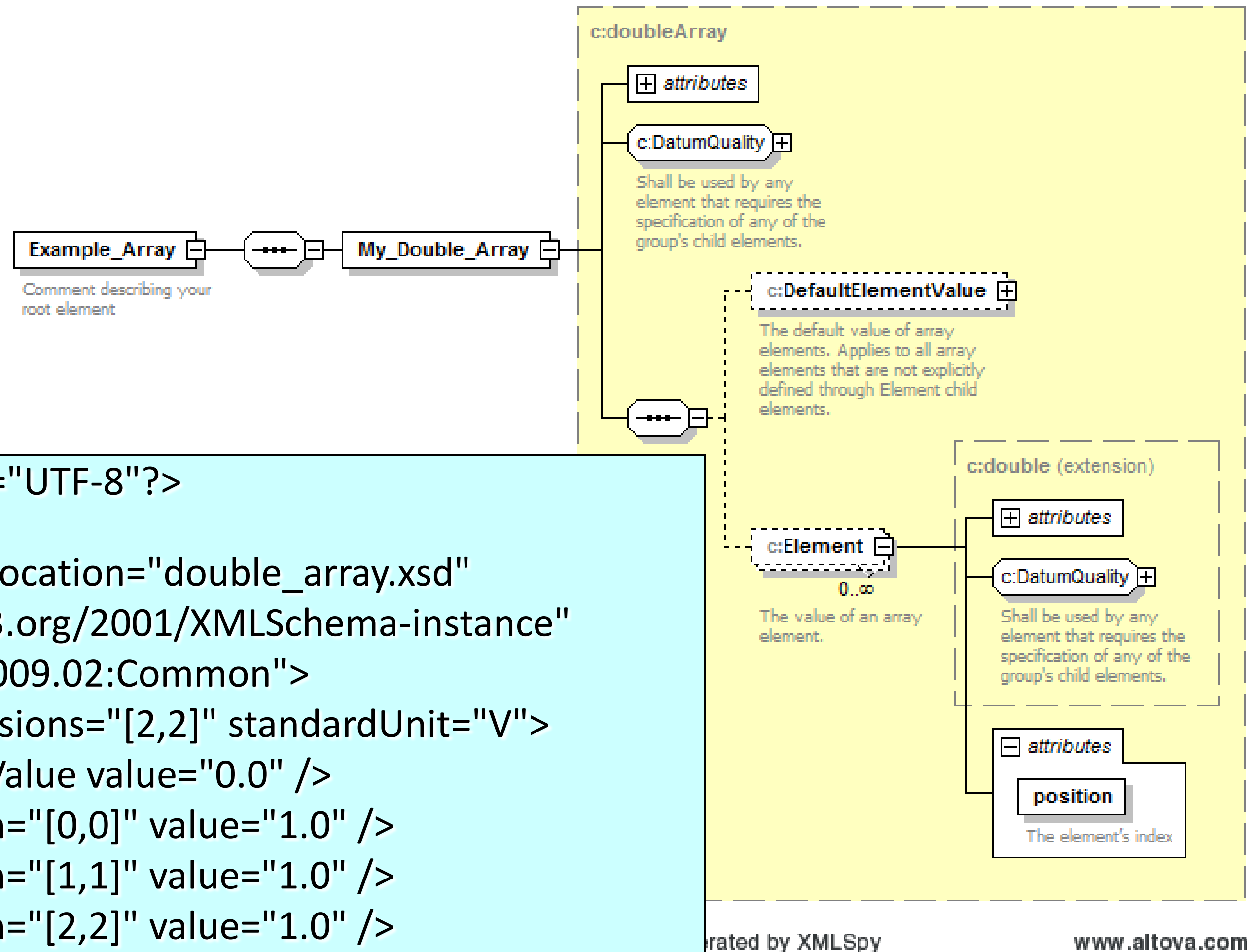
http://grouper.ieee.org/groups/scc20/tii/atml_status.htm

ATML Overview and Architecture IEEE Std 1671-2006



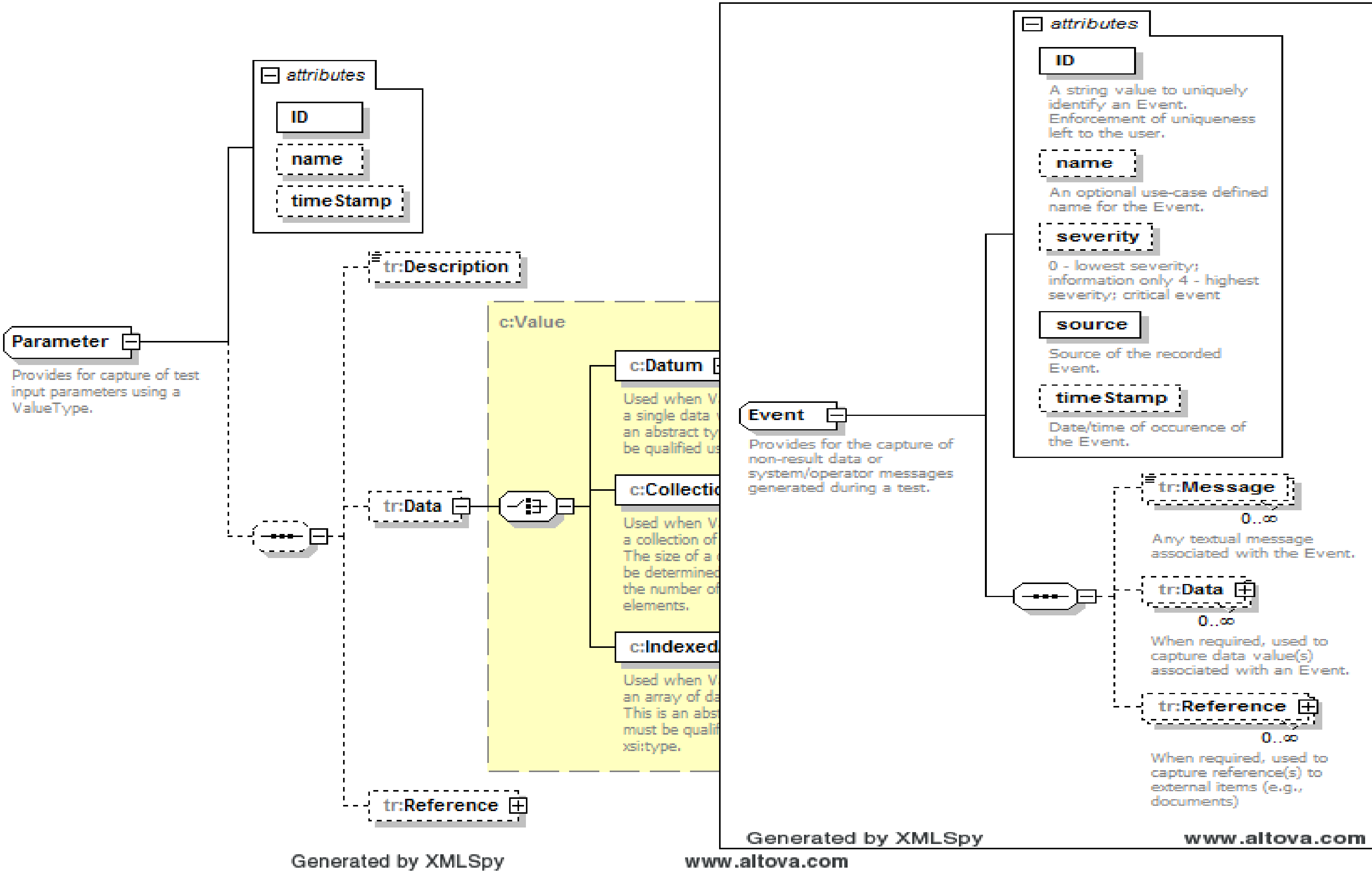
Common Data Types

- Integer
- Unsigned Integer
- Hex
- Octal
- String
- Binary
- Boolean
- Complex
- dateTime
- Double




```xml
<?xml version="1.0" encoding="UTF-8"?>
<Example_Array
    xsi:noNamespaceSchemaLocation="double_array.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:c="urn:IEEE-1671:2009.02:Common">
    <My_Double_Array dimensions="[2,2]" standardUnit="V">
        <c:DefaultElementValue value="0.0" />
        <c:Element position="[0,0]" value="1.0" />
        <c:Element position="[1,1]" value="1.0" />
        <c:Element position="[2,2]" value="1.0" />
    </My_Double_Array>
</Example_Array>
```

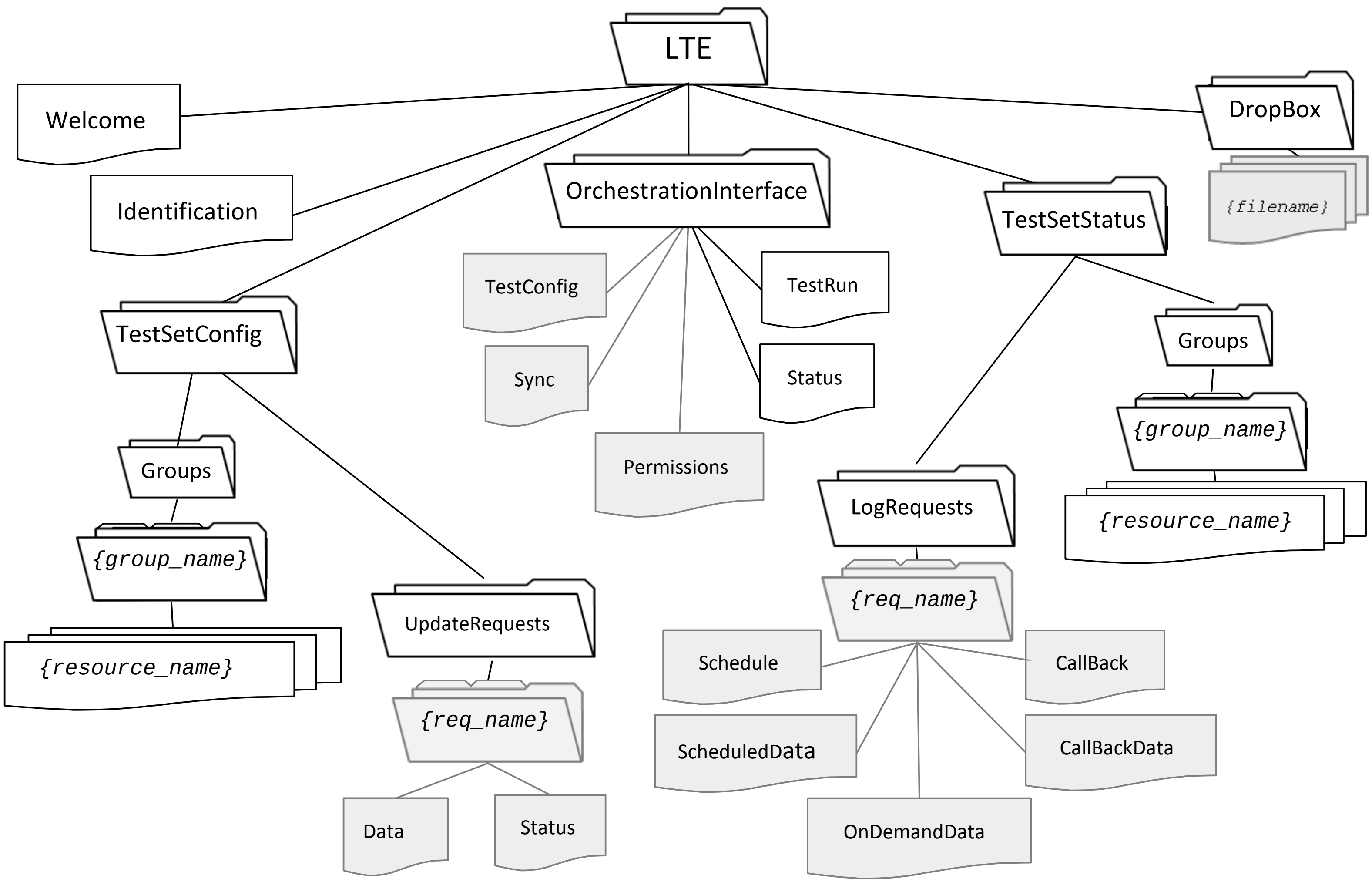
TestResults Data Types



The AHA Protocol

RESTful architecture is **simple**

- The state of the LTE is made available to LTE clients through the LTE resource tree
- It is manipulated through requests to the resource tree
- The protocol defines the minimum resource tree and standardizes optional branches



A Revolutionary New Idea!

HP BASIC

SCPI

ATLAS

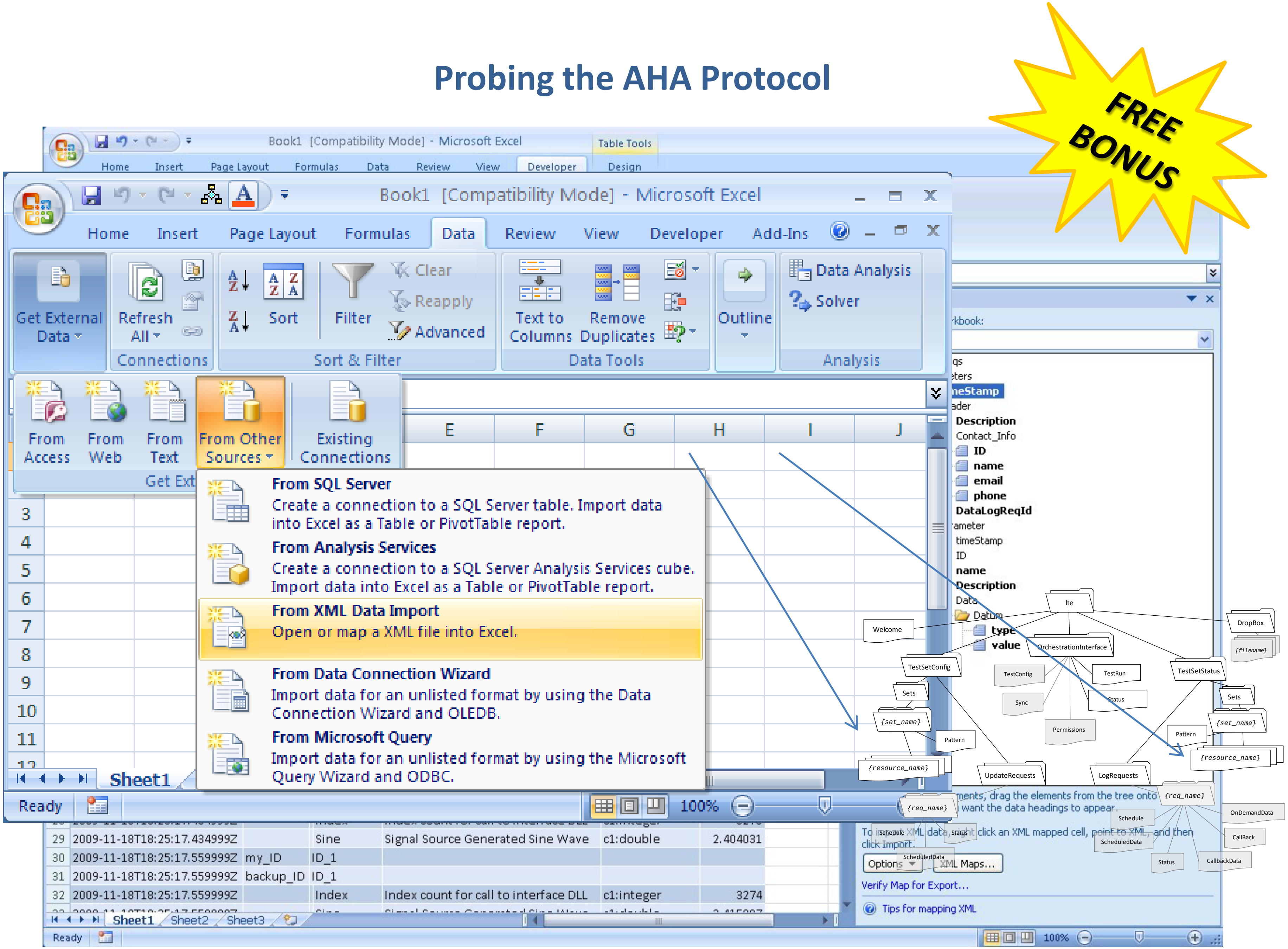
SATOCM

Verb Based

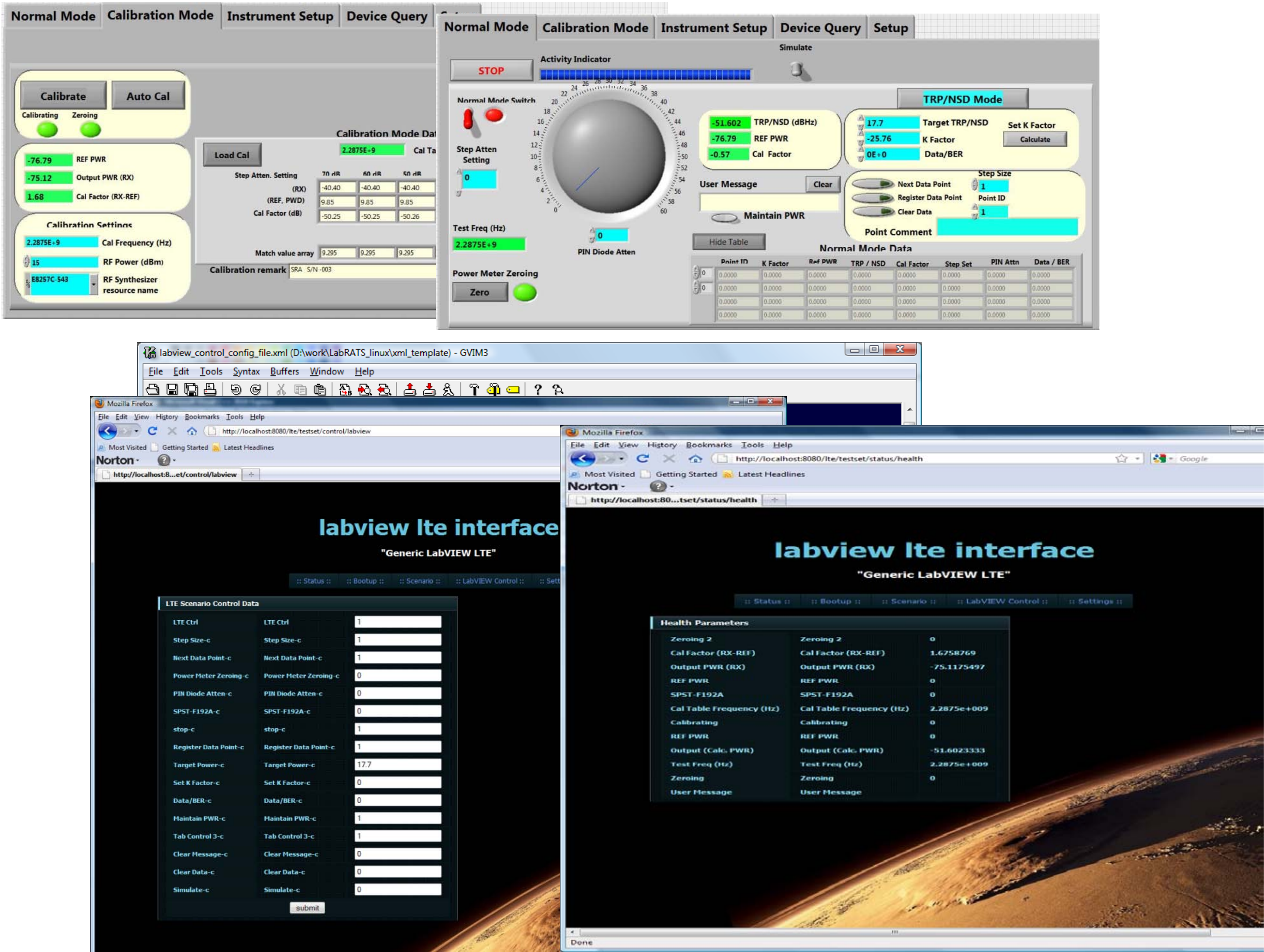
Noun Based

AHA

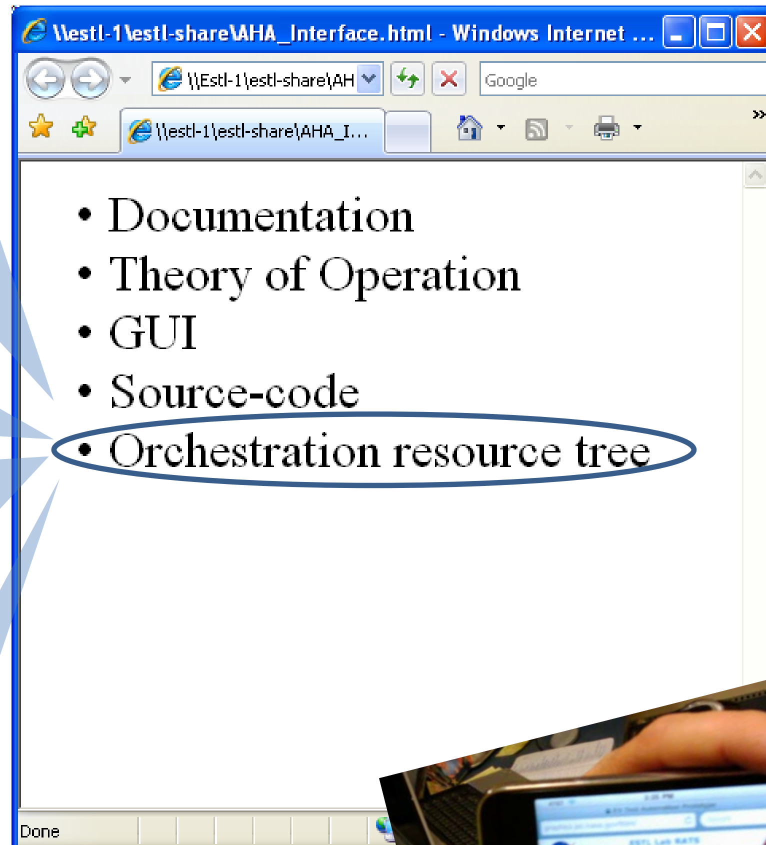
Probing the AHA Protocol



Human- and Machine-Readable Interface

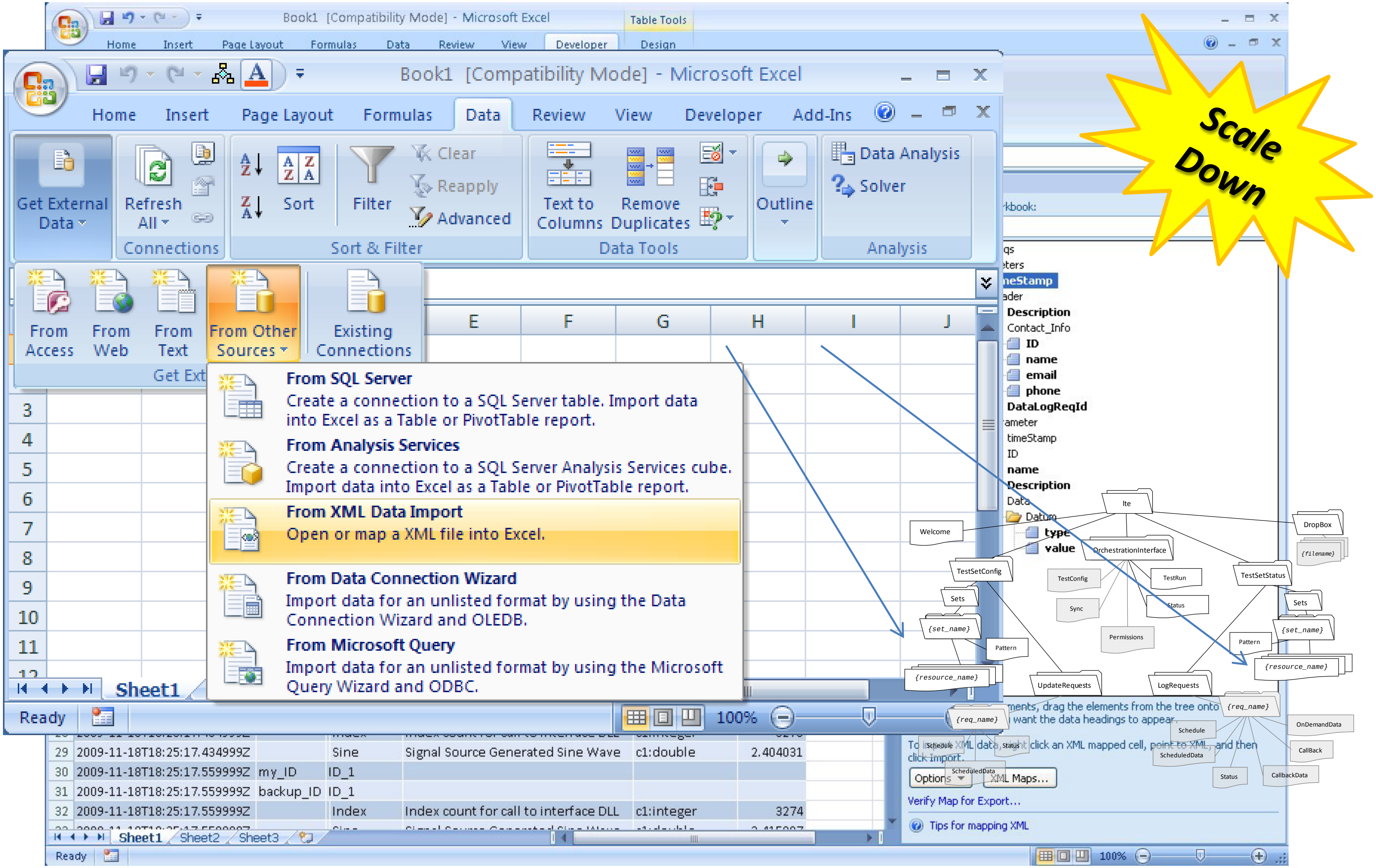


Human- and Machine-Readable Interface



Web Based! *You already have the tools in your hands!*

Value Proposition: Use Instantly for Box-level Development Test
Reuse Later for Subsystem and System Integration Test



A Scale-to-One Architecture

- Box-Level Development
- Subsystem Integration Test
- System Integration Test
- Multi-Element Integration Test

PostgreSQL

Orchestrator Development

MySQL Community Server

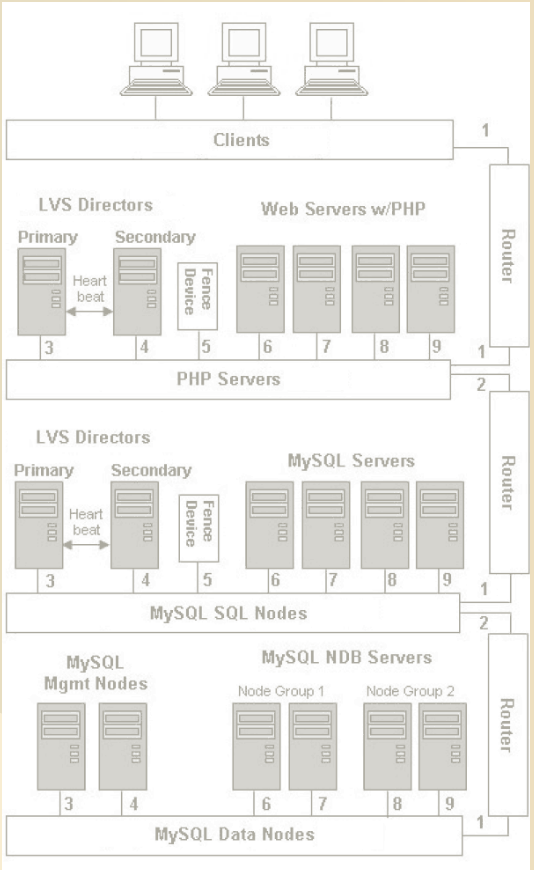
Microsoft SQL Server 2008 R2 Express

EnterpriseDB The Enterprise Postgres Company

Stackable Orchestrator Development

MySQL Enterprise

Microsoft SQL Server 2008 R2 Enterprise



Other Attributes We Like

- Merges with Industry Trends
- Modular Open Source

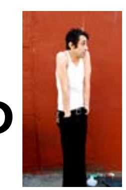
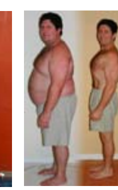
- Small but Powerful Command Set

checkin	installConfig	archiveData
masterCheckin	deployConfig	Cleanup
serviceCheckin	getCurrentConfig	peek
subsystemCheckin	getAvailableConfigs	poke
healthEvent	configureScenario	overrideVariable
stateChangeNotice	getCurrentScenario	resetVariable
archiveDataComplete	getAvailableScenarios	getAvailableParams
reboot	configureFault	remoteCommand
initialize	getActiveFaults	getAvailableRemoteCommands
shutdown	getAvailableFaults	verifyTimeSync
selftest	loadCheckpoint	getStatus
stopTest	saveCheckpoint	ping
startTest	getAvailableCheckPoints	getAvailableCommands
pauseTest	stopRecording	logMessage
resumeTest	startRecording	disconnect
uninstallConfig	cancelArchiveData	generateReport

Get Delete
Put Post

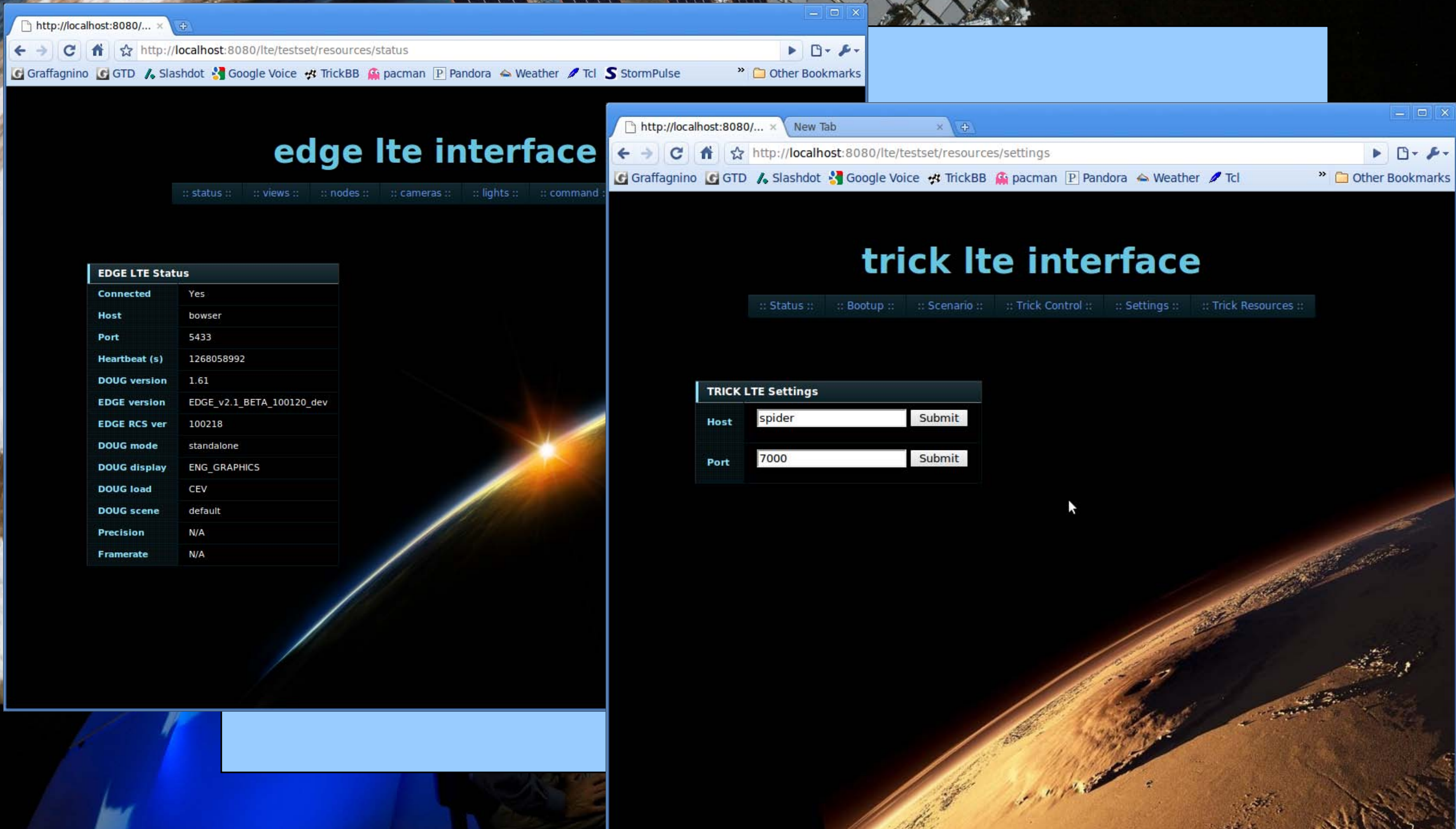
- Rich, Widely Accepted Error-Message Set

- Loosely Coupled

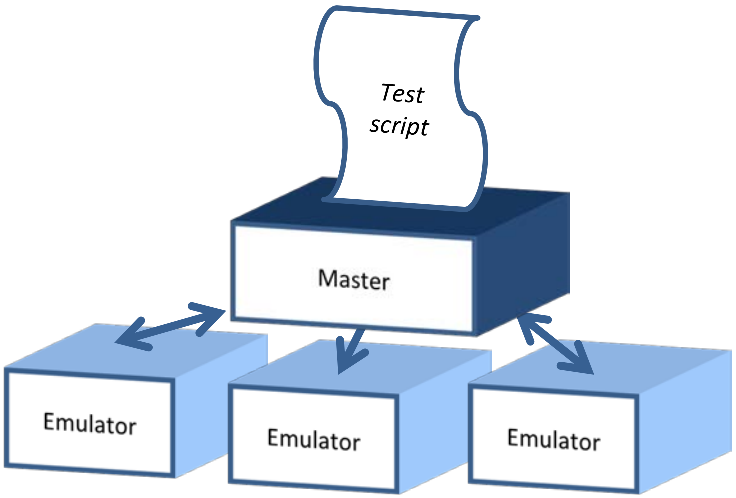



- Metadata, with Data Compression by HTTP
- Security by HTTPS

AIE Orion Simulation

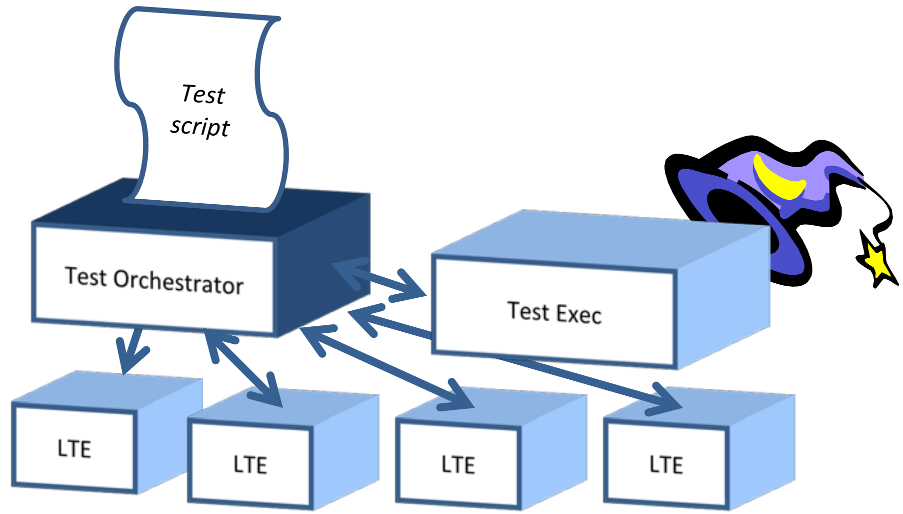


TestExec LTE and Stackable Orchestration



Test Master executes Test Scripts

•Scripts are not defined by ICD, scripts developed for one Master wouldn't work on a different Master
•Scripts are not intelligent or visual
•Scripts are developed by a different process in a different language than other software



Test Exec provides Guided Test Flow

•Test Exec is a plug-in tool to conduct a specific measurement sweep. It incorporates test specialty knowledge generally from the same test specialty team that develops the LTEs.
•Test Exec is developed as a specialized LTE. By this concept, the same design team can develop LTEs and Test Execs by the same processes.
•Script could invoke a series of Test Execs.
•Test Orchestrator is in charge, Test Exec provides steering by evaluating data, fitting models, extrapolating the next point to run ("intelligent"), and displaying results and expectations ("dashboard").

Developers: Division of Disciplines

Process and Data Orchestration

•AHA Protocol expertise
•Database expertise
•IT expertise

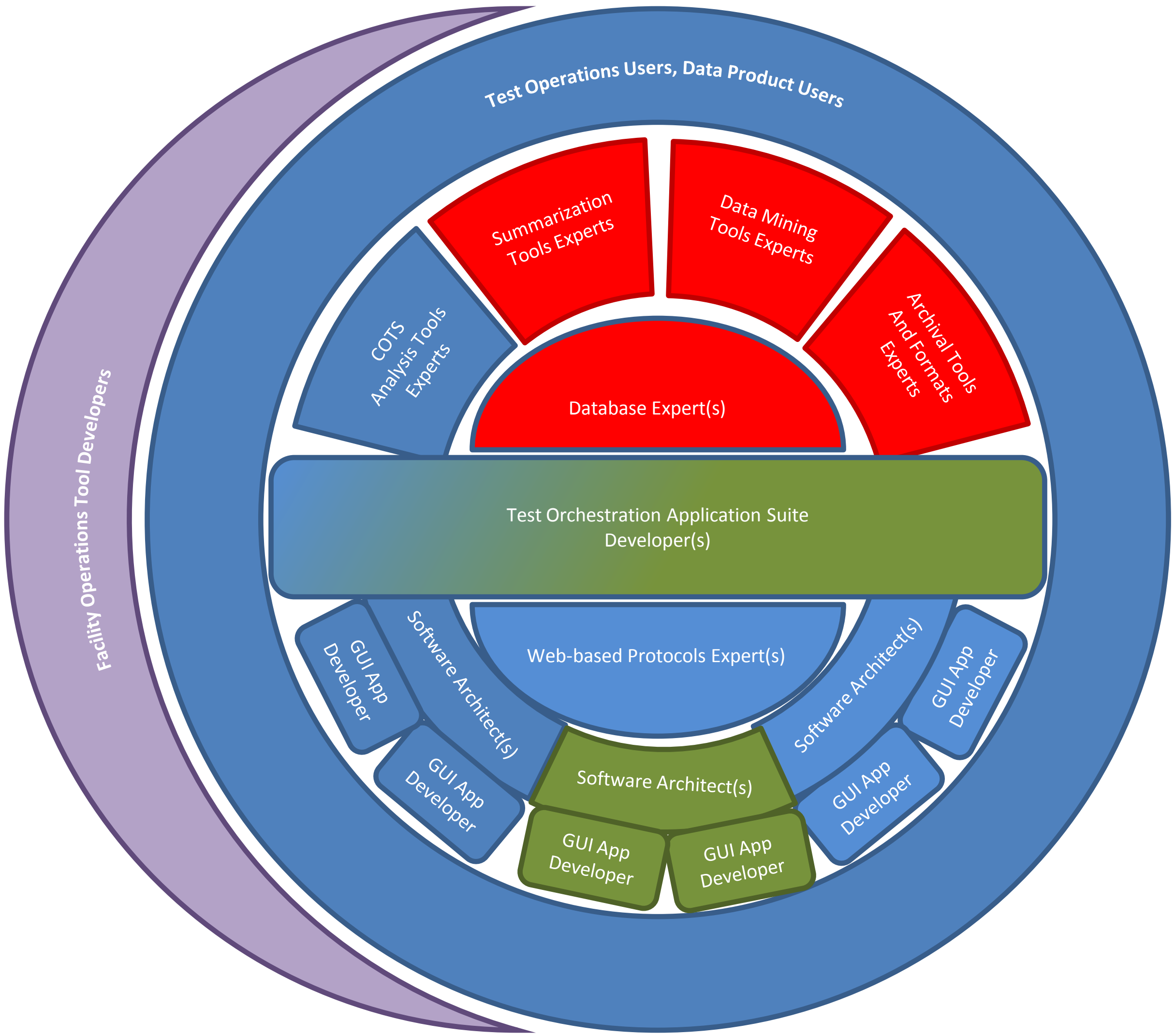
LTE and Test Execs

•Sensor, Actuator, and Simulation tool specialty expertise
•Test and measurement methodology expertise

Multi-Disciplinary Problem

NASA has all of the necessary disciplines, inside the agency!

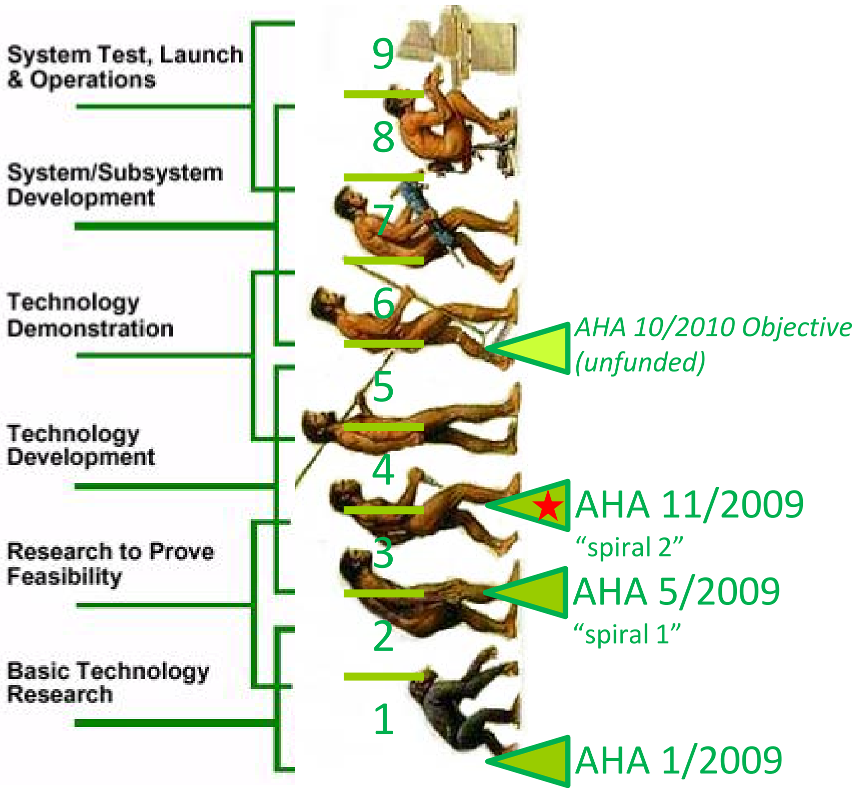
- LabVIEW, Trick and other Developers and Architects at JSC in EV, ER, and ESCG
- Test Orchestration at MSFC and JSC
- METECS has assembled a team of part-time experts to draw on
- Test operations users in ESTL, AIE; data users in Safety
- Data formats and tools developers under CIO's office at Ames, JPL, and elsewhere



Pathfinder Lessons: Leading The Way Around

- Intellectual Property protection
- *mouseclick* Events and Dialog boxes
- IT Security
- Customer needs today’s work done, now
- Traffic and storage unknowns
- IT Plans
- Software Quality
- Many possible approaches

Maturity of AHA as a Test and Measurement Protocol



NASA Technology Readiness Levels, applied to Test Orchestration Protocol	
Technology Readiness Level	Description
9. Actual system 'flight proven' through successful mission operations	Protocol has broad acceptance
8. Actual system completed and 'flight qualified' through test and demonstration (ground or space)	Solid protocol with no TBDs, and good tools exist for easy implementation. Penetration into many communities.
7. System prototype demonstration in a space environment	Stable prototype with large-scale integration of the prototype into real LTEs in a hardware test environment to conduct actual test operations and produce usable data products. The protocol is used in several test environments and is gaining community acceptance
6. System/subsystem model or prototype demonstration in a relevant environment (ground or space)	Stable prototype with small-scale integration of the prototype into real LTEs in a hardware test environment to conduct actual test operations and produce a real and usable data product
5. Component and/or breadboard validation in relevant environment	More stable prototype with small-scale integration of the prototype into real LTEs in a hardware test environment
4. Component and/or breadboard validation in laboratory environment	Prototype integration of available open-source implementations for selected candidate technologies, in a simulation environment, to conduct a simulated test
3. Analytical and experimental critical function and/or characteristic proof of concept	Proof-of-concept integration of available open-source implementations for various candidate technologies, in a simulation environment
2. Technology concept and/or application formulated	Technology down-selection, Technical Merit Evaluation and Community Acceptance Trending
1. Basic principles observed and reported	Technology survey, Communities of Practice survey

- AHA is a collection of TRL-9 protocols. But the resource tree, orchestration flow, and raw performance need to be evaluated in the actual T&M end application(s).

Background

Business Case

Technical Argument

BACKUP

Activities

- Surveyed test technologies used in testing and networking communities (DSIL, ISIL, DoD, Commercial)
- Evaluated these technologies based on our Guiding Principles and ability to support the Test Automation Hooks requirements
- Evaluated several options for discovery and database middleware with standalone prototypes
- Created a minimal set of commands/resources sufficient to implement the basic Test Automation Hooks requirements
- Chose a set of technologies to provide a baseline upon which to further evaluate issues and options
- Prototyped an integrated discovery, commanding, and data gathering configuration using this baseline set of technologies and Trick simulations to represent test rigs

Minimal Command/Resource Set

- Discovery
  - Detect LTE’s on network and provide enough info to test manager to identify LTE’s for test configuration and to define LTE resources
- Synchronize
  - Add LTE’s to test (associate with test_name) and sync time
- Config
  - reload or change LTE configurations (fully or partially)
- Data_Transfer
  - Request status data, database logging, file transfer, event messages (associate with data_log_id and test_run_id)
- Exec_Test_Run
  - Start activity associated with test_run_id
- Stop
  - Stop data logging, test run or test config
- Custom_Commands/Resources
  - Defined by LTE during identification

Test Technology Survey

Test Technology Survey

- Performed a survey (web, interviews, tour) to identify methods/protocols being used for discovery and test control in relevant test communities.
- Looked at two NASA examples
  - DSIL
  - ISIL
- Looked at two DoD examples
  - TENA
  - DoD ATS
- Looked at Commercial Products
  - NI Labview and Test Stand Test Software
  - IXIA IP Test Automation

DSIL

- DSIL is a distributed SIL architecture allowing coordination and orchestration of remotely located SIL assets including coupled dynamics simulations.
  - Flight communication interfaces are through high speed data interfaces
  - Simulation communication and synchronization is through High Level Architecture (HLA)
  - Orchestration and less time critical Cx Data communication between simulations is accomplished with the MAESTRO package developed at Marshall

DSIL

- HLA
  - For the DSIL, HLA provides time advancement capabilities used to interface coupled dynamics simulation. This capability is not required for the Automation Hooks architecture
  - HLA architecture requires a Run Time Infrastructure (RTI) and simulation federate interfaces.
  - Although the HLA architecture is well defined, RTI’s have their own implementations and are in general not interchangeable. Therefore, the protocol interface to the VI’s would not be an open standard.
  - Because the Automation Hooks architecture doesn’t benefit from the advanced features of HLA, the complexity of the architecture does not warrant its consideration as a test architecture

DSIL

- MAESTRO
  - MAESTRO provides orchestration of test commands and has been planned for additional use outside of DSIL
  - It is configured using XML and Python scripts
  - Our understanding is that the command tool runs under Windows. Linux communication tools exist and have been integrated with Trick simulations
  - Commands are sent using C3I protocol which is a Constellation program specific protocol developed for communication between flight and ground systems.
  - C3I is not widely supported in the test industry or commercial communities and so does not meet our Guiding Principles test for the Automation Hooks study
  - It may be of benefit to the Automation Hooks project to look for areas of collaboration with the MAESTRO community in the areas of command sets and test automation flow. Perhaps MAESTRO can be extended to send commands in formats that meet open architecture goals.

ISIL

- ISIL is a test facility for integration and test of ISS software and avionics
- It is located in Sonny Carter Training Facility and its capabilities have grown in concert with ISS assembly
- The facility is now mature. A considerable amount of effort has been devoted to developing efficient automation interface to meet their needs

ISIL

- Communication between test stations (MATE's/Simulators) is accomplished via shared memory – not Ethernet
- Discovery is not required because the bulk of the configuration is well known and remains part of the lab assets
- Personnel involved have a good handle on operation issues in an complex and operational facility and would make a good resource for reviewing and commenting on potential Automation Hooks pitfalls
- The software suite used by ISIL is large and complex and it would take some effort to determine whether components of it would be of benefit to the Automation Hooks project.
- It may be of benefit to the Automation Hooks project to look for areas of collaboration with the ISIL particularly in understanding what test automation issues should be considered in planning for an operational facility

TENA

- Test and Training Enabling Architecture (TENA) is a DoD initiative supporting test ranges and facilities
- It is aimed at distributed simulation and test applications and hence includes complexity that doesn't contribute to the Test Automation Hooks objectives
- TENA is based on CORBA, an object-based messaging protocol that has been declining in popularity because of its complexity and historical difficulty penetrating firewalls (interest has shifted from CORBA and its competitor DCOM to Web Services which are discussed later)
- TENA relies on middleware that appears to be single source and hence does not meet the Guiding Principles
- For these reasons TENA is not considered appropriate for the Test Automation Hooks task

DoD ATS

- The Navy Automatic Test Systems (ATS) Executive manages the Open Systems approach to reduce ATS costs across the DoD
- ATS has defined an ATS framework that is largely based on the following technologies
  - Virtual Instrument Software Architecture (VISA)
  - Interchangeable Virtual Instruments (IVI)
  - Automatic Test Markup Language (ATML)
- VISA and IVI provide lower level instrument drivers and are restricted to Windows OS. They are also not widely used outside of the ATE industry.
- ATML is an emerging XML-based IEEE standard that has potential benefit to the Test Automation Hooks task

Commercial Tools

- Most commercial Automatic Test Equipment provides drivers for test instrumentation and interfaces to execute user-provided test scripts. Examples include National Instruments Test Stand and IXIA Test Conductor (for IP testing)
- Drivers require Windows so Standard Commands for Programmable Instruments (SCPI) commands are used on Linux machines. SCPI is not suitable for system level testing
- The LXI protocol has some interesting discovery capabilities that have been evaluated for use in Test Automation Hooks
- The most interesting development in commercial tools is the emergence of XML-based ATML capabilities although these are in the initial stages of adoption

Technology Assessments

ATML

Discovery

Messaging

Database Middleware

Technology Assessment
ATML

ATML

- Automatic Test Markup Language promotes test equipment interoperability by standardizing around the eXtensible Markup Language (XML) format.
- Consists of a family of standards and XML schemas that represent automate test related information.
- This includes descriptions of
  – Test configuration
  – Test Stations
  – Instrument Description
  – Unit Under Test Identification
  – Test Adapter
  – Test Description
  – Test Results
  – Diagnostics

ATML

- The last of the ATML standards is set to be released as a trial-use standard
- One of the standards that is mature and will likely survive mostly intact is the Test Results (IEEE 1636.1)
- The ATML standard recommends implementation of ATML with use of web services and SOAP

ATML

- A Phase I demonstration of ATML capabilities was held at autotestcon in September, 2008
- A Phase II demonstration is planned for September, 2009 with the following companies expressing interest in participating: Agilent, Boeing, EADS T&S, Geotest, Indra, Lockheed Martin, macPanel, National Instruments, Northrup Grumman, Pideso, Teradyne, TYX, Raytheon, Rohde&Schwarz, Virginia Panel Corporation, Vektrex.

ATML

- The conclusion is that Test Automation Hooks architecture would benefit from protocols that do not preclude incorporation of ATML as it matures
- This supports adoption of an XML-based and web services-based architecture

Technology Assessment
Discovery

Discovery

- Interested in technologies for discovery on Ethernet that allow discovery not only for devices, but also of services provided by those devices
- Technologies of interest were VXI-11, UPnP, and zeroconf (mDNS/DNS-SD)

VXI-11

- VXI-11 is a common protocol in the T&M industry that has discovery capabilities
- Can be used for the discovery of Ethernet connected instruments but has little use and support outside of the T&M industry
- The LXI specification originally mandated VXI-11 for discovery, but eventually moved to the zeroconf discovery method in the 1.2 version of the specification

UPnP

- A zero configuration networking solution backed by Microsoft and Intel, UPnP tries to solve similar discovery issues for small networks without centralized servers to manage all devices.
- Based upon a protocol known as Simple Service Discovery Protocol. Also addresses other layers such as addressing, messaging, eventing, and presentation.
- UPnP is becoming common for certain classes of network devices such as routers and media servers
- Has some inertia, is cross platform (uses HTTP as a transport), has a few open source implementations/SDKs, and is language neutral.
- Was considered as the discovery mechanism by the LXI Consortium, but the consortium eventually went with zeroconf
- Does not appear to have any crossover in the T&M industry.

Zeroconf

- LXI attempts to reuse a technology developed by Apple called zeroconf (also known as mDNS/DNS-SD) to solve the issue of addressing and discovery.
- mDNS/DNS-SD is a zero configuration networking technology originally developed by Apple during their conversion from AppleTalk to IP based networking. It is a combination of multicast DNS and DNS Service Discovery. Apple's current implementation is known as Bonjour (formerly Rendezvous).
- Open source implementations exist (such as Avahi for linux)
- Is cross platform (uses DNS), is language neutral, and has considerable inertia (used in many software products, and networked devices such as printers)
- Is an open standard (specifications available at http://www.multicastdns.org/ and http://www.dns-sd.org
- Has some acceptance by the test and measurement industry: LXI has chosen zeroconf as their methods for auto addressing and discovery. Also, many large commercial tools (i.e. Labiew, MATLAB, etc.) support discovery of LXI equipment using these protocols.

Technology Assessment
Messaging

Messaging

- Also referred to as “control” when referring to device protocols
- Messaging technologies are exploding with respect to web programming and fields such as Messaging Oriented Middleware and Web Services.
- Many of these technologies were examined (low level, protocols, and API level), including XML-RPC, SOAP, CORBA, DCOM, JMS, UPnP, Web Services, REST Web Services, AMQP, RestMS, etc.
- Tons of growth and development occurring in this field. May be too risky to move to an infant technology, but will want to choose a baseline technology that allows growth and evolution with these emerging standards.

UPnP

- UPnP has its own sense of Control and Event messaging.
- Implemented via SOAP and GENA
- Meets many of the goals, but does not have wide acceptance and does not appear to have any use in the T&M industry

JMS

- Java Message Service is a Java Message Oriented Middleware API
- Many implementations and tools available
- Definition at the API level creates interoperability issues
- Not language neutral (although it looks like some have attempted wrappers for other languages)

AMQP

- Advanced Message Queuing Message Protocol originated in the financial industry,
- AMQP defines a wire level protocol ensuring interoperability and language independence.
- AMQP is an open standard, but is in its infancy (1.0 of the specification is currently in draft status).
- Message Oriented Middleware solutions include capabilities we may never need (message queuing and message routing are fundamental to the technologies)

SOAP

- Originally stood for Simple Object Access Protocol
- Successor of XML-RPC
- Usually uses XML as its message format and HTTP as its transport making it language neutral, OS independent, and generally immune to firewall issues.
- Forms the base protocol stack for Web Services

Web Services

- Defined by the W3C as "a software system designed to support interoperable machine-to-machine interaction over a network"
- Typically uses SOAP and XML with WSDL
- Web Services Description Language allows a web service provider to advertise its commands available and types defined for parameters and returns
- Web Services have gained much inertia from major software vendors, and while expected to evolve, should continue to be supported and developed around. (WSDL 2.0 was made a W3C Recommendation in 2007)
- Many implementations and tools available for development
- OS, Language, and platform independent.
- The architecture examples in Annex C of the ATML Trial Use Standard shows the usage of Web Services and WSDL

Emerging Technologies

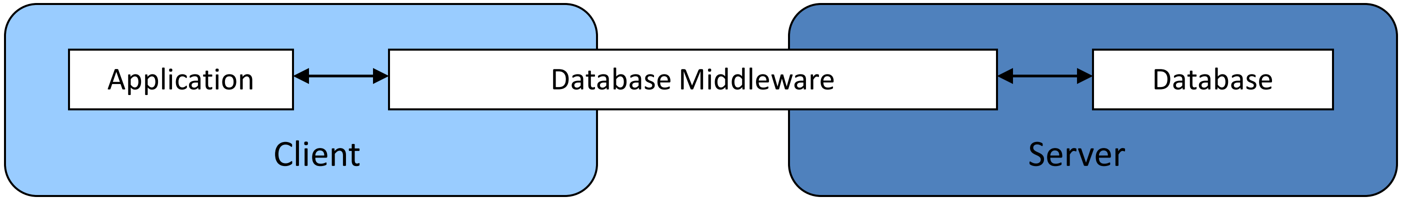
- With so much growth and evolution occurring in this field, it is important to keep an eye on emerging technologies most applicable to test automation
- Web Services over http would allow growth into the following without a large shift in architecture (think version numbers):
  - DPWS – Devices Profile for Web Services
    - An implementation and set of rules to allow web service discovery, identification, control, and eventing on devices (similar to UPnP)
  - RESTful Web Services
    - Web services implemented adhering to the principles of Representational State Transfer
    - Many big web service providers have started to transition from SOAP based Web Services to REST implementations
    - Basically a shift from “command-based” to “resource-based”
    - More similar to how http operates.

Technology Assessment
Database Middleware

Functionality

Database-oriented middleware provides the following key functions:

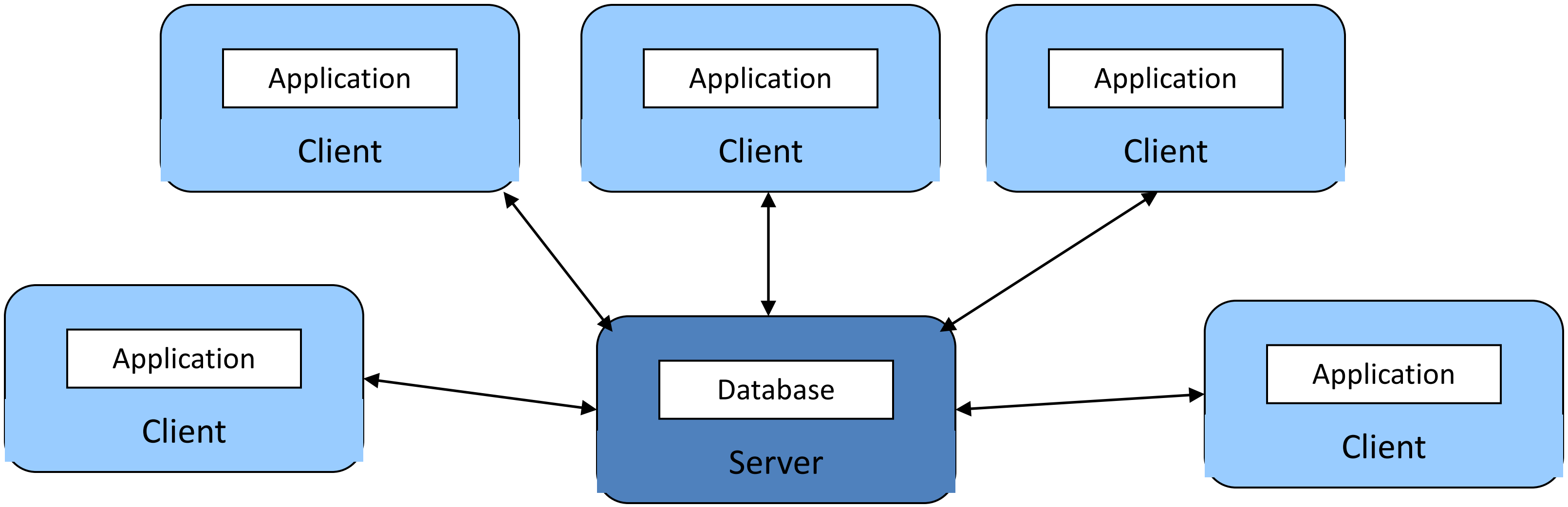
- The ability to convert the application language to be understandable by the database
- The ability to send a query to and process a response from a database over a network



Objective

In a test environment, there are likely to be many client applications and one database

– The main goal is to allow the database to be upgraded without impacting each client

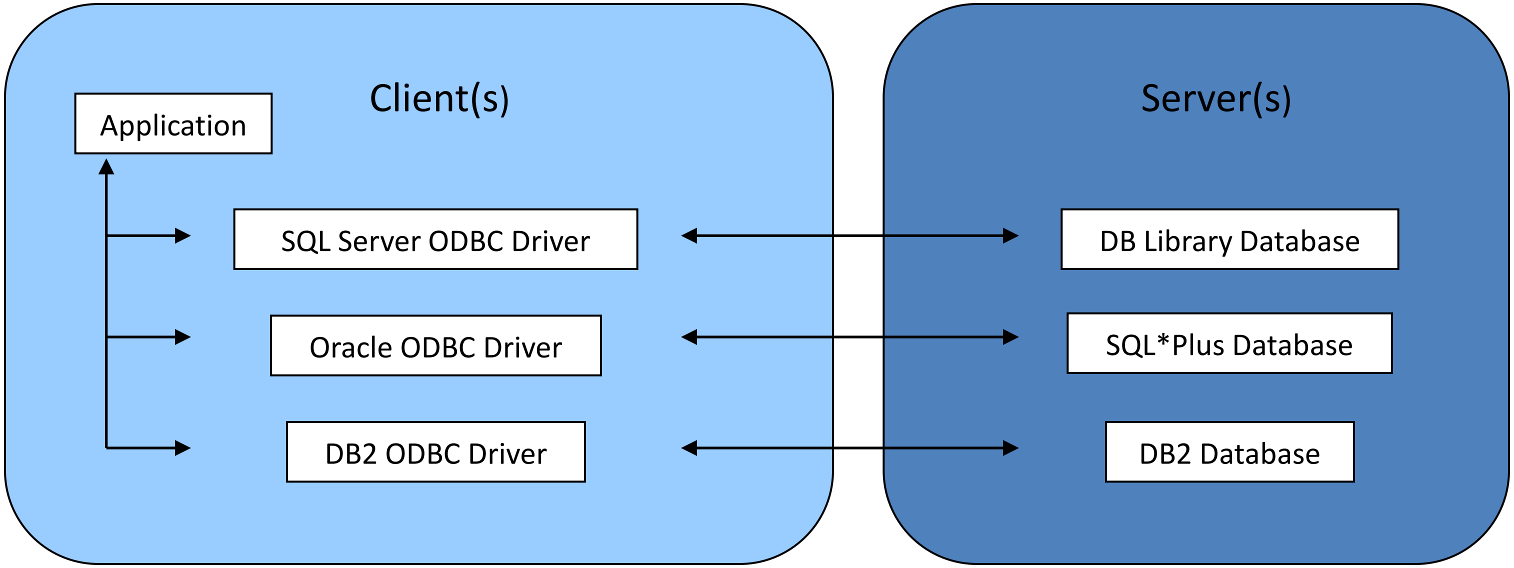


Types of Middleware

- There are two main types of database-oriented middleware:
  - **Native Middleware** – created for a specific database
    - Oracle Call Interface (OCI) for Oracle
    - LibPQ for PostgreSQL
  - **Call Level Interfaces (CLIs)** – provide a single interface to several databases
    - **Open Database Connectivity (OCBC)** is the "gold-standard" for generic database interfacing
    - **Java Database Connectivity (JDBC)** is primarily used for web-based applications
- To avoid making application source code changes on each client when the database is upgraded or replaced, a CLI-type middleware must be used

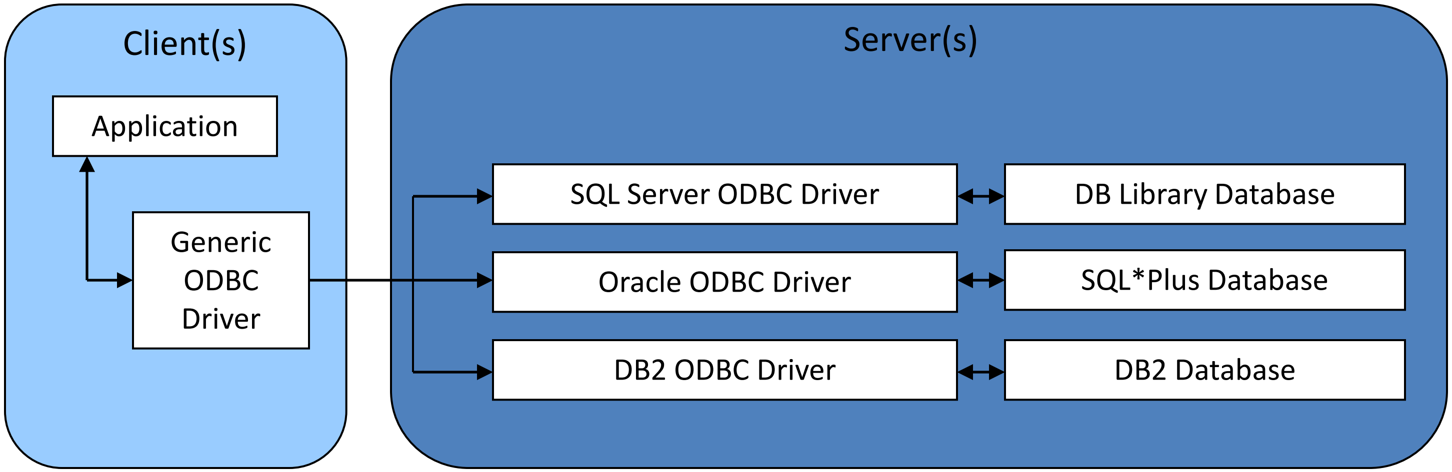
ODBC

- ODBC is a multi-database API standard for programs that use SQL to access data
- An ODBC driver for the specific database must be installed on each client (this is similar to how a printer driver must be installed before a client can print to a specific printer)



Bridge Driver

- An ODBC-ODBC bridge driver provides the capability to install the specific driver on the server and keep the client side completely generic



Compatibility

- Most Microsoft products contain ODBC interfaces (e.g., MS Excel, MS Access)
- ODBC toolkits are readily available for LabView
- XML data can easily be stored in any database as a text string, however this does not allow for detailed searching of the XML field
- XML to SQL conversion middleware software can be used to translate XML data into SQL commands
- Most databases are now offering native XML data types, thus eliminating the need for additional XML middleware

Database Performance

- Since ODBC is a higher-level abstraction, there could possibly be a performance hit when compared to using a native interface
- A small test was setup using the PostgreSQL database to evaluate the performance of using ODBC verses LibPQ, the native PostgreSQL interface
  - A Trick simulation was configured to log data to PostgreSQL running on a different machine using both LibPQ and ODBC
  - A table is created in the database and then a new row with nine double precision variables is added during each logging cycle
- Test Results
  - It took 0.0012 seconds to log a cycle of data using LibPQ
  - It took 0.0013 seconds to log a cycle of data using ODBC

Baseline Technology Set

Purpose

- Based on the technology assessments, a baseline set of technologies was chosen for discovery, messaging, and database middleware
- Purpose of the baseline set is to use as a starting point for end-to-end prototyping and as a comparison for evaluating other options

Rationale

- Discovery
  - Zeroconf: based on test industry support and commonality with LXI
- Messaging
  - SOAP: based on XML compatibility (ATML, NExIOM) and huge user and support base
- Database Middleware
  - ODBC to ODBC bridge: based on support by COTS tools and desire to reduce LTE maintenance

Scientific Technical Information (STI) I Disclosure
Export Control Rationale Checklist

Instructions: This checklist should be attached to the JSC Form 548 (JSC Approval of Scientific and Technical Information for External Release) or NASA Form 1676 (NASA Scientific and Technical Document Availability Authorization (DAA)). In support of requests for disclosures of STI, and information controlled under Export Regulations (International Traffic in Arms Regulations (ITAR) and Export Administration Regulations (EAR). **Please review and sign [next to] the applicable rationale item[s] below[on the next page]. Additional written information is required for Item V.**

Background: Release of NASA information into a public forum may provide NASA technology to countries with interests adverse to the United States. This form will help you efficiently process your proposed disclosure of NASA, Scientific and Technical Information (STI) and assure that the request complies with export control regulations. The JSC Export Services Team (EST) will use this rationale in conjunction with the completed NF 1676 or JF 548 which is required for each domestic and international presentation and publication of Johnson Space Center STI (See CWI J29W01).

Generally, the export of information pertaining to the design, development, production, manufacture, assembly, operation, repair, testing, maintenance or modification of defense articles (i.e. space flight hardware, ground tracking systems, launch vehicles, radiation hardened hardware and associated hardware and engineering units for these items are controlled by the International Traffic in Arms Regulation (ITAR). The complete listing of ITAR controlled items can be found at http://www.pmdtc.org/reference.htm#ITAR. The export of information related to ground based sensors, detectors, high speed computers, national security and missile technology items are controlled by the Bureau of Industry and Security (BIS) under the Export Administration Regulations (EAR).

If the information that you propose for release falls within the above categories, but falls into one or more of the following “Rationale for Public Release” items, the information may be determined to be suitable for public release.

Scientific Technical Information (STI) I Disclosure
Export Control Rationale Checklist

Title of Presentation: Automation Hooks Architecture Trade Study for Flexible Test Orchestration **Author:** Chatwin Lansdowne

Rationale for Public Release I
The information is already in the public domain in its entirety through a non-NASA medium and/or through NASA release previously approved by the JSC EST

____ _____________
Name of Publication Medium (e.g. periodical, meeting, etc.) Date of Publication

Typed Name	Signature	Mail Code	Date

Rationale for Public Release II
The information pertains exclusively to the release of scientific data, i.e. data pertaining to studies of biomedical or planetary sciences without disclosure of information pertaining to articles controlled by the ITAR or EAR such as flight instruments, high speed computers, or launch vehicles.

Typed Name	Signature	Mail Code	Date

Rationale for Public Release III
The information falls into the areas of concern as referenced above, but is offered at a general purpose of high level, e.g. poster briefs and overviews, where no specific information pertaining to ITAR or EAR controlled items is offered.

Typed Name	Signature	Mail Code	Date

Rationale for Public Release IV
This information pertains exclusively to the release of software and assurance methodologies or studies, without disclosing information pertaining to articles controlled by the ITAR or EAR.

Chatwin Lansdowne	*/s/Chatwin Lansdowne*	EV3	6/16/10
Typed Name	Signature	Mail Code	Date

Rationale for Public Release V
There is a compelling written reason for the public release of the information that is not covered by the “rationale” items I – IV above. The information to be released can not be used to exploit or defeat controlled U.S. technologies. It is therefore requested that the Center Export Administrator review the attached supporting statement and approve the release of the information pursuant to the exemption CFR22 - 125.4(b)(13). To use this rationale, the requestor must provide a written statement that explains why the release of the information is a reasonable and advisable action.

Typed Name	Signature	Mail Code	Date