

ATML Demo Phase I

Executive Summary

February 2008

1	Executive Summary	2
2	Conclusions & Recommendations	3
3	DoD ATS Framework Achieved Objectives	5
4	DoD DISR Standards	5
5	Technical Summary	6
5.1	Overview Phase I Core	6
5.2	Objective & Goals.....	8
5.3	Phase II Possible Candidates	10
5.3.1.1	Test Diagram Generation (TOWL) (high)	10
5.3.1.2	ATML Test Description supporting TPS life cycle (high)	10
5.3.1.3	Modular instrument description within a ATS description (high).....	11
5.3.1.4	Use of Test Configuration to support MTPSI (high)	11
5.3.1.5	Interfacing with Test Results archiving databases (high)	11
5.3.1.6	Test Station capability analysis (medium)	12
5.3.1.7	Resource and Path Allocation (medium)	12
5.3.1.8	Capability digital and bus, extending test subject (medium).....	12
5.3.1.9	Mapping Test Description into Instrument based test programs (low). 12	
5.3.1.10	Real UUT – use a real TPS and UUT as a test subject (low)	12
5.3.1.11	Legacy – support of legacy test programs (low).....	13
5.3.1.12	Alternative Platforms – ARGCS, VDATS, RTCASS (low).....	13
5.4	Lessons Learned.....	14
6	Technical Report.....	18
6.1	Demonstrating A Complete Testing Scenario	18
6.1.1	Introduction and objectives	18
6.1.1.1	System level use cases	18
6.1.1.2	Utilized standards.....	18
6.1.2	Hardware test environment and UUT design.....	18
6.1.2.1	UUT hardware and test strategy	19
6.1.2.2	ATE and test configuration.....	20
6.1.2.3	Interface test adapter	22
6.1.3	Software test environment design	24
6.1.3.1	ATE support software	24
6.1.3.1.1	COTS software products.....	25
6.1.3.1.2	ATML and STD signal modeling data.....	26
6.1.3.1.3	Support software outputs	28
6.1.3.2	ATE control software and ATE system software	28
6.1.3.2.1	COTS software products.....	29
6.1.3.2.2	SIMICA data and AI-ESTATE models	29
6.1.3.2.3	Station control software inputs	29

1 Executive Summary

This report provides an Executive Summary of the findings and recommendations made as a result of performing the ATML Phase I demonstration and a full presentation at AutoTestCon 2008. The Executive Summary is followed by a Technical Summary and Technical Report providing more detailed description of the ATML Demonstration.

The purpose of the demonstration was to validate the performance of the collection of ATML standards while providing key evidence showing how the ATML family of standards can advance the DoDs ATS Framework objectives:

- Faster technology insertion
- Improve TPS rehost and interoperability
- Use model based programming techniques
- Modernize test programming environment
- Greater use of commercial products
- Define interfaces to support integrated diagnostics

As a direct result of the demonstration the following activities have been supported and benefits have been achieved:

- As the IEEE Std. 1671 is being updated to a full use standard, the ATML files created for the demonstration are being incorporated as a complete worked example.
- Changes were made to ATML standards based on the lessons learned from the demonstration. These changes were either applied to standards currently within ballot, or raised as issues to support standards updates to occur in the next 12 months
- Provided to the user community a complete set of working files, as an example of how to use ATML in a working system <http://grouper.ieee.org/groups/scc20/tii>
- Advanced the availability of COTS ATML tools.
- Raised the profile of the ATS Framework working group and the benefits of using standards within the test & measurement domain to reduce costs

The demonstration achieved all of its objectives and showed what can be achieved through common goals and industry/government partnerships. The participation and interest is reflected both in the number of active companies (13) involved in the demonstration, and the size of the audience both at the scheduled live demonstrations (30+) and the ATML seminar presentation (50+). The success of these has resulted in a ATML 2009 AutoTestCon slot already being booked to provide a follow on at next year's AutoTestCon.

Having completed the ATML Demo Phase I, there are several recommendations identified by the team for a follow-up phase that would bring continued benefits to help in the transition of ATML from a standard to deployed solutions achieving the DoD ATS Framework Objectives.

ATML Demo Phase I - Executive Summary

- Show innovation and ***Faster Technology Insertion*** through deployed use of ATML information and tools, such as Test Requirements Traceability, Test Strategy reports, Test Diagram Generation
- ***Improved TPS Rehost and Interoperability*** by using ATML within different Test Program Environments that support ATML Test Description for runtime and test requirement configuration control.
- Incorporate the ATML format into existing test station system software so that ATML files provide all the test information, allowing ***Greater Use of Commercial Products***. Extend the use of ATML Instrument Description in the demonstration, for example to map ATML Test Description to instrument based test programs.
- Introduce additional aspects of using ATML files, dependent on the responses received from industry after Phase I, to ***Modernize Test Programming Environments***.
- Utilize the ATML Test Results exchange format to archive information, enabling the demonstration of ***Interfaces that Support Integrated Diagnostics***
- Align ATML Demonstration Phase II with current program needs, to show how an example solution could be used, with continued development, to support existing projects.

2 Conclusions & Recommendations

The Technical Report fully describes the integration of ATML tools used during Phase I to demonstrate the end-to-end integration of all the ATML standards, between all (13) participating companies. The Phase I demonstration took 4 months to put together to achieve the transformation of an ATML Test Description into a test program running on multiple ATS platforms. What became evident during this period was that, in addition to showing how the standards are integrated, the ATML standards allowed the individual participants to integrate the modular components into the whole solution using the ATML standards. Put another way, the Phase I demonstration was able to work because each contributor was able to obtain the test information from the common ATML files produced, and therefore shared test information within their tools.

One of the remaining key challenges is how to advance the ATML standards for practical use, the goal being for ATML standards to be used and provide benefits to fielded solution. Paramount to achieving this is having tools (COTS or custom) that create and consume ATML information that is used to tackle real problems associated with today's working practices. A Phase II demonstration needs to address this area targeting the use of ATML standards in support of specific practices and processes. This feasibility demonstration should encourage the continued use ATML, even after Phase II is complete.

This inherently requires ownership, buy-in, co-operation and interaction from various DoD users and programs in order to demonstrate the feasibility of using test information conforming to the ATML standards. By demonstrating the feasibility on live systems, this also helps overcome the initial hurdle of getting commercial products to interchange ATML files, because this creates an opportunity to have ATML files delivered and used.

ATML Demo Phase I - Executive Summary

A Phase II is needed to specifically look into the use of COTS products that can be configured for fulfilling targeted ATML applications. As in Phase I, it will rely heavily on software tools contributed by commercial vendors. Prototypes are acceptable as long as documentation or support is available. Use of in-house tools will be considered where no match is available. The ATML tools consist of any application that produces, translates, or consumes the ATML format (editors, display tools, converters, database, etc.). This is an area where we expect a wide range of diversity and innovation while supporting the interchangeable ATML format.

Phase II is intended to show innovation and faster technology insertion through the use of ATML tools for existing projects and processes. It targets selected applications and shows how ATML can be used to improve the current processes, while adding or enhancing commercial tool support.

Phase II will also aim to prove the ability to re-use information from the various ATML standards. The tools configured or created can be used by real applications and to be integrated into existing processes or systems.

2.1 Recommendation #1

ATS Framework Group should identify potential DoD users and programs who would be suitable as process owners. The ATS Framework Group should obtain approval from AMB to encourage industry in demonstrating the feasibility of using ATML on existing systems.

2.2 Recommendation #2

An ATML Phase II Feasibility Demonstration should be devised, proposed, and facilitated through the ATS Framework Group & NxText IPT, by establishing a project team from all interested parties. This Phase II Feasibility Demonstration should build upon the Phase I infrastructure and outputs, and incorporate new tools to support the programs identified above, with a goal of showing ATML's Readiness For Use.

2.3 Recommendation #3

ATML Phase II project team should establish success criteria associated with the feasibility demonstration to facilitate continued use of ATML on the selected target projects.

2.4 Recommendation #4

The project team should include members from DoD stakeholders, NxText representation, ATS Framework members, key commercial organizations who have shown a commitment to the success of ATML.

3 DoD ATS Framework Achieved Objectives

The following activities and goals were achieved in support of the DoD ATS Framework Objectives:

Faster technology insertion

- Information reuse allows new resources to be added
- Common information format allows test technologies to be adapted
- XML tools accelerate adoption of technologies

Improve TPS rehost and interoperability

- Information reuse
- Support for multiple test executives

Use model based programming techniques

- ATML used to model ATE hardware and connectivity
- Model and simulation based on Signal Models

Modernize test programming environment

- Access to test executives and development environments through XML support

Greater use of commercial products

- Information exchange allows integration of commercial tools
- Common information format allows access to commercial products
- XML helps overcome access to proprietary information formats

Define interfaces to support integrated diagnostics

- Information flow provides required infrastructure
- Information format provides for common exchange

4 DoD DISR Standards

- IEEE Std 1671 Automatic Test Markup Language (ATML)
- IEEE P1671.1 ATML Test Description
- IEEE Std 1671.2 ATML Instrument Description
- IEEE Std 1671.3 ATML UUT Description
- IEEE Std 1671.4 ATML Test Configuration
- IEEE Std 1671.5 ATML Test Adaptor
- IEEE Std 1671.6 ATML Test Station
- IEEE Std 1636.1 SIMICA Test Results & Session Information
- IEEE Std 1641 Signal & Test Definition

5 Technical Summary

The following provides the technical summary of the Phase I demonstration. An additional “Step by Step Guide” is provided in the accompanying PowerPoint presentation titled “ATML Demo Full”, which details all the steps the team went through to achieve the demonstration. As part of the technical summary an outline of the lessons learned and a breakdown of possible Phase II ATML applications is provided.

5.1 Overview Phase I Core

The Phase 1 Core demonstration took an existing system (RTCASS/IP ATE, test subject (UUT) and test requirements documentation) and demonstrated the process of creating a test program from test information written in the standard format as shown in Figure 1&2.

The demonstration incorporated several COTS tools that were integrated to provide a complete end-to-end TPS development and execution. The initial phase considered specific standards and their application. The integration was constrained only to those parts necessary to complete a final test program execution. The demonstration used several suitable tools which already existed within the same area, and this provided evidence of greater use of commercial products, however it did not represent widespread availability of tools.

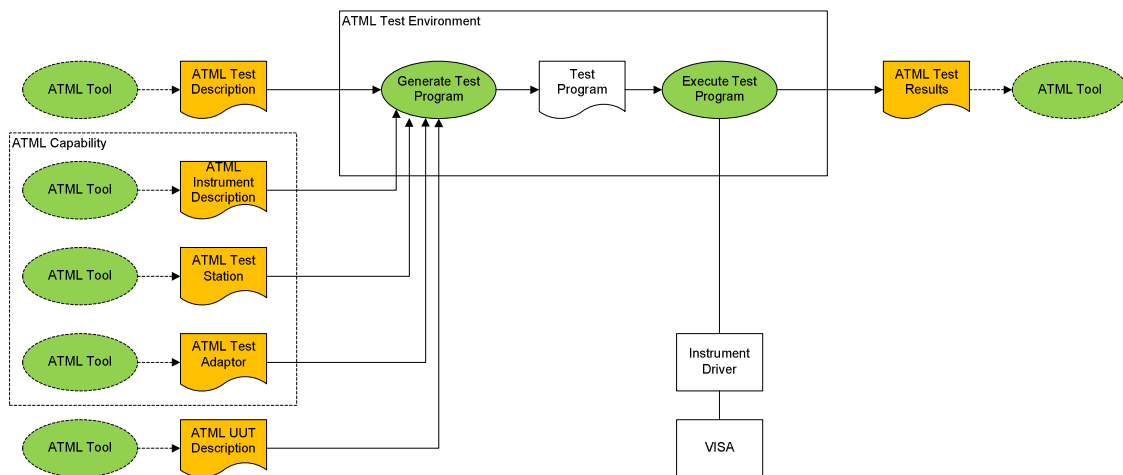


Figure 1 – ATML Test Environment - Phase 1 Core

ATML Demo Phase I - Technical Summary

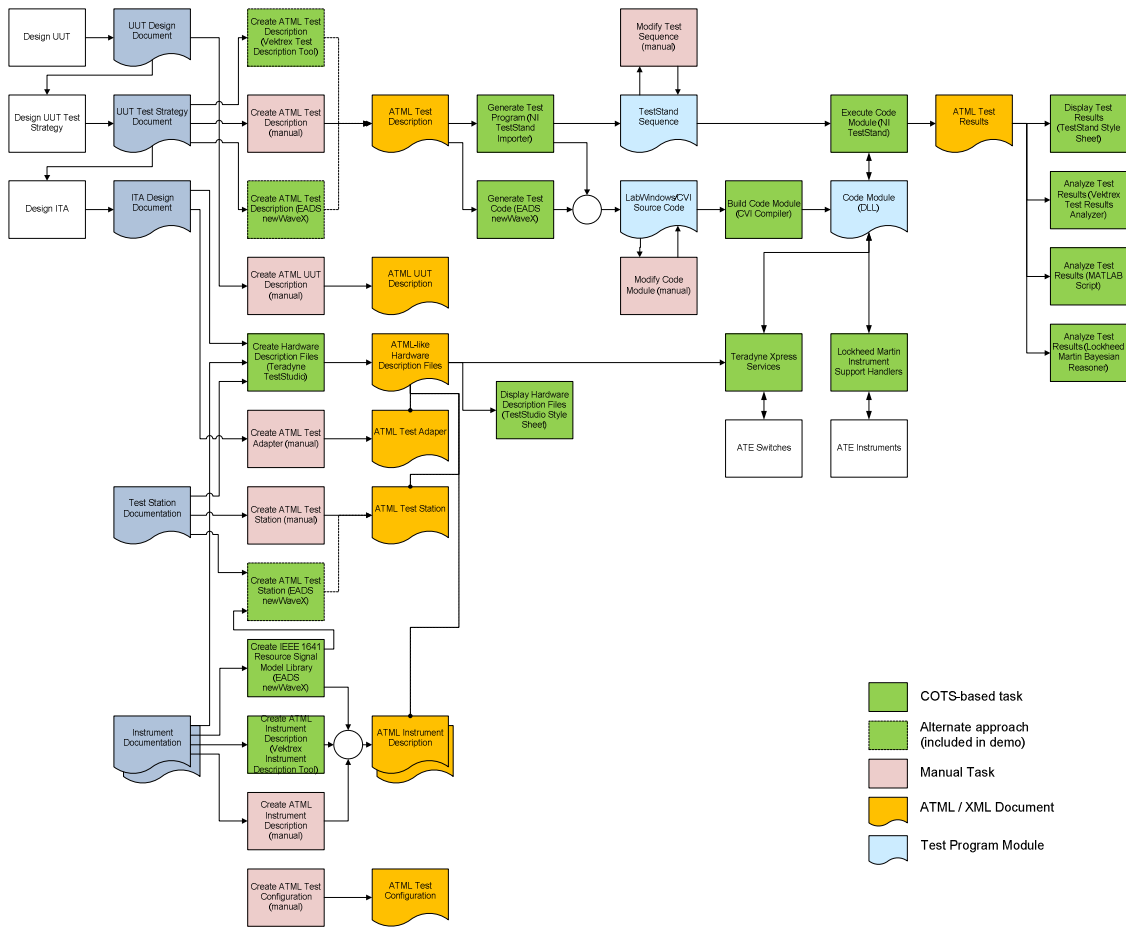


Figure 2 – ATML Test Environment – Complete Demonstration

The Phase I Core (Figure 1) integrated several COTS components into an test environment solution. Future phases should build on this work to add additional components and show transportability between different test executive's components.

The Phase 1 Core demonstrated ATML test information being consumed and translated into an executable test program, test program run on a test station providing ATML test results and session information, and the use of this information for analysis and diagnostic purposes.

The ATML test information was created both manually (with XML editors) and through the use of ATML tools, in some cases producing duplicate information, which could be compared for consistency. The detail of information was sufficient to support the selected test subject.

ATML Demo Phase I - Technical Summary

The ATML information included:

- 1) Test Description and Signal Model Libraries
- 2) Instrument Descriptions (partial)
- 3) Test Station (including ATML description of capabilities)
- 4) Test Adaptor
- 5) UUT Description
- 6) Test Configuration

5.2 Objective & Goals

These are aimed to accelerate the maturity and commercial acceptance of ATML IEEE SCC20 Standards for inclusion into the Department of Defense (DoD) Information Technology Standards Registry (DISR).

- Shows innovation and Faster Technology Insertion through the use of ATML information and tools.
- Highlights Improved TPS Rehost and Interoperability by executing ATML through multiple test executives including a legacy ATLAS test executive.
- Incorporates the ATML format into the existing test station system software so that the ATML files are used directly allowing Greater Use of Commercial Products.
- Introduces additional aspects of using ATML files, dependent on the responses received from industry from Phase I allowing utilizing Modernized Test Programming Environment.
- Utilizes Test Results exchange format to archive information as a Defined Interface to Support integrated Diagnostics

The aim of these target tools is to show how the suite of ATML standards are ready for use and help facilitate the following DoD ATS Framework objectives:

- Faster technology insertion
- Improve TPS rehost and interoperability
- Use model based programming techniques
- Modernize test programming environment
- Greater use of commercial products
- Define interfaces to support integrated diagnostics

Phase II will build on the Core phases, adding additional ATML components and features including items such as:

- TRD to Test Description import
- Test Strategy Report Generation from Test Description
- Use of Test Configuration to support MTPSI
- Interfacing with Test Results archiving databases
- Independent Test Program Environments Support

**Future TPS development - ATML Test Information Using Signal Models
Enhancing Reusability**

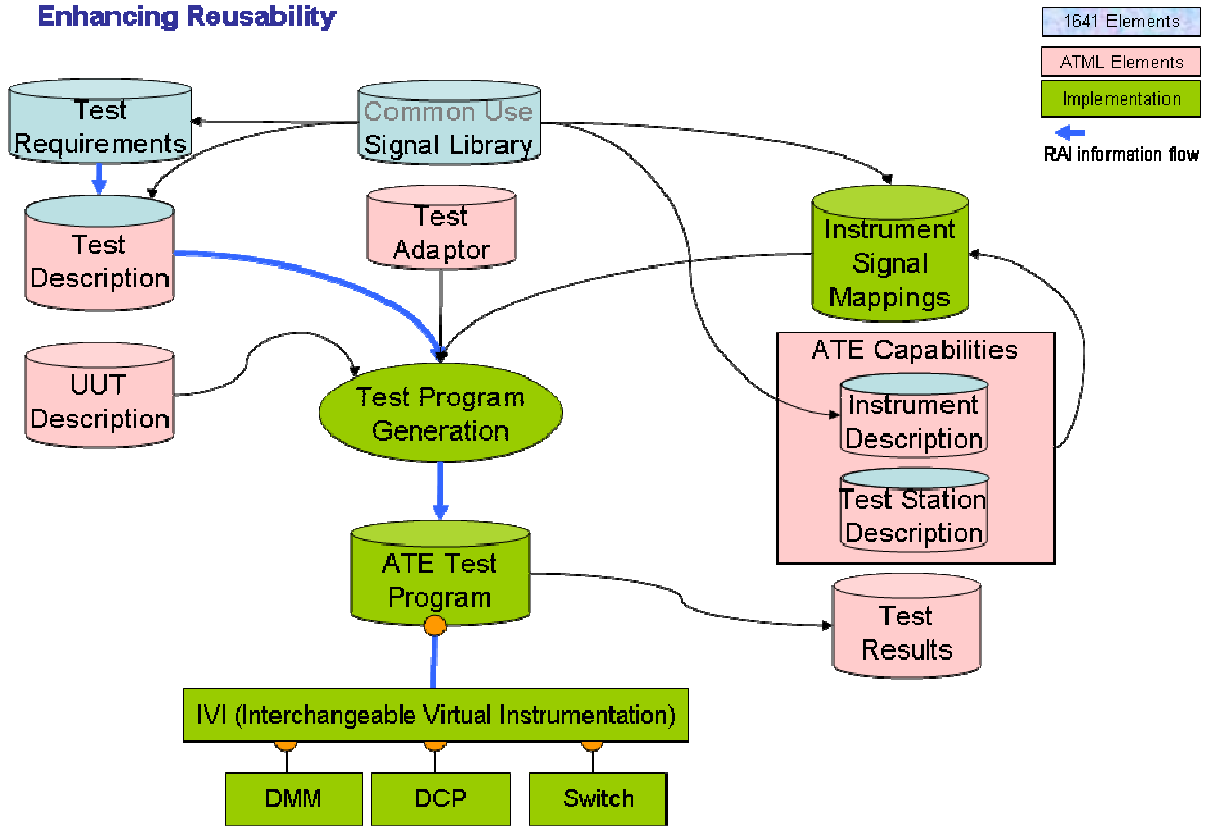


Figure 3 – ATML Test Information Using Signal Modeling

The use of commercial tools and the ability to adapt to new technologies are key to the success of the adoption of the standards and therefore the completion of the ATS Framework Key Elements. As such, this phase aims to increasing both the support for ATML tools and their use as part of real working processes. This will provide a platform for ATML tools and products to come together as an extended environment with a summary presentation at AutoTestCon.

5.3 Phase II Possible Candidates

The following scenarios are offered as possible candidates for an ATML Phase II Demonstration. This represents a shopping list of possible ATML uses, that ideally would be backed by some DoD project or group, wanting to take them further, and incorporate them into existing practice.

5.3.1.1 Test Diagram Generation (TOWL) (high)

Develop a process and/or a tool to take Test Station, Test Adapter and Instrument Description ATML Instance files and using the various Network List information from the Interface and Switch elements to automatically generate test diagrams representing the signal paths for particular tests.

Each agency has their own tool, but there is no standard format that allows these tools to share information. ATML should be that common format, such that the information could be shared across tools.

The information required would be represented in several ATML files, and as such supports the goal of getting ATML test information used in "live" situations, and create a demand of ATML test information to be provided.

This process would also validate the changes recommended from Phase I, associated with Test Configuration.

There are several system suppliers who already provide (and use) these type of tools, who would be interested in the ATML Phase II.

- Boeing: Already using XML files for this purpose; ATML support would be an additional benefit.
- TYX: Users use different tools (GTOWL) often tied to specific CAD software, ATML may help provide a common format.

5.3.1.2 ATML Test Description supporting TPS life cycle (high)

Aim is to provide TPS life cycle support by providing guidance in how ATML Test Description information could help support key activities. It is expected that the outcome of this demonstration task would be incorporated into some DoD Guidance Document explaining/showing key items such as:

Test Requirements Traceability

An ATML tool to convert TRD formats into ATML Test Description, to support existing TRD formats, plus the capability to regenerate TRD from modified Test Description.

Test Strategy reports

A tool to create Test Strategy Reports from ATML Test Descriptions, providing a user/field readable format in terms of ATLAS-like signals and descriptions that explain the steps the test program is going through to test the unit.

ATML Demo Phase I - Technical Summary

Demonstrate the process of generating Test Strategy from ATML Test Descriptions, then show how Test Strategy changes can be incorporated back into the ATML Test Description to provide document version tracking.

LifeCycle

Identify key information required at different milestones/stages in a TPS development cycle/rehost and how that maps to the various ATML standards. Possibly generate reusable examples or templates that encode the above key information.

It been identified that Joe Stanco and Mukund Modi are already involved in a ATS Framework activity, to develop a TPS life cycle model .What is unclear is how the scope and depth of that effort compares with this ATML effort, which is seen as a very practical level requiring the input from experienced TPS practitioners.

5.3.1.3 Modular instrument description within a ATS description (high)

This provides a modular process for describing ATS Capabilities, using ATML Instrument Description, with Test Station and Test Adaptor in line with the “Future TPS Development” Diagram.

This builds on the Phase I tools, but rather than using the Test Station to completely describe Test Station Capability, derives Test Station capability from the individual Instrument Descriptions. This ATS Capability could then be used to compare test requirement against available test platform types, derivative of which would allow comparison of test program rehost analysis, to legacy instrument comparisons.

5.3.1.4 Use of Test Configuration to support MTPSI (high)

There already exists some XML-based MTPSI system; however there is no formal link with the ATML Test Configuration standard. By extending the system to natively use ATML Test Configuration, it aligns the DoD’s internal effort with the standard information format.

Support for this was shown from Lockheed Martin and Boeing, with key skills coming from NAVAIR at Jacksonville.

5.3.1.5 Interfacing with Test Results archiving databases (high)

Test Results probably has the biggest commercial take-up and acceptance of any of the ATML standards. In part this is due to being the first, but also one represents a non contentious format that most vendors need only produce (rather than consume). Results archiving systems exist¹, what is therefore required is to consume test results in the ATML format and map them into the native formats of various software tools. With this approach, ATML Test results can become part of the formal process, regardless of which system they originated from.

¹ Examples of such a system <<http://zone.ni.com/devzone/cda/epd/p/id/5829>>

There is a lot of interest in this area, and it is one where any Phase II could help facilitate and act as a catalyst, for industry contributions. Support for this was shown by Lockheed Martin.

5.3.1.6 Test Station capability analysis (medium)

Develop a process and/or a tool to take an ATML Test Description file and an ATML Test Adapter instance file and compare them against the capabilities of one or more Test Stations using the ATML Test Station and Instrument Description instance files. This will determine if the Test Station meets the requirements to test a particular UUT.

5.3.1.7 Resource and Path Allocation (medium)

Generate a more extensive version of the demo that shows how the various ATML files can be used for advanced runtime and/or pre-runtime Resource and Path Allocation. A basic allocation capability was demonstrated in Phase I.

5.3.1.8 Capability digital and bus, extending test subject (medium)

Demonstrate the applicability of an ATML-based solution to UUTs that require digital and bus testing.

This can be accomplished by enhancing the existing demo UUTs. For instance, we could add a small microcontroller board that controls the amplifier, while providing a serial bus and parallel I/O.

5.3.1.9 Mapping Test Description into Instrument based test programs (medium)

This feature demonstrates mapping ATML Test Description into a test program code using the ATML Capabilities from Test Station, Instrument Descriptions and Test Adapter to create a targeted test program in line with the “Future TPS Development” Diagram.

This builds on the Phase I tools, using the modular instrument description within a ATS description, to help generate generic instrument commands. This would allow developers to help to bridge the gap between implementing “manual” TPS functionality and automatically generation test programs derived from Test Description.

This instrument-based approach helps promote the standards in a manner consistent with commercial ATE applications. Although this technique is not preferred in current military system architectures, however for future systems, there is a strong argument for having individual instrument capability being compounded at a system level to support an Open System Architecture.

5.3.1.10 Real UUT – use a real TPS and UUT as a test subject (low)

Identify an actual UUT (possibly the one attempted at the end of Phase 1?). Need to have at least one UUT spare. Ideally, we would also have schematics for the UUT and ITA.

ATML Demo Phase I - Technical Summary

Implement in an ATML-based system the original TPS, or a part of it. To demonstrate technology insertion, this would be a legacy ATLAS run time system with ATML features. To demonstrate rehosting, this would be an alternative solution like the one demonstrated in Phase I.

5.3.1.11 Legacy – support of legacy test programs (low)

A large portion of existing Test programs exist written in ATLAS. These currently run in their own environment, and supported with their existing tool set. There is no driving requirement to move these TPS from their current environment. However there are two future scenarios that could be relevant

- a) Re-hosting a Legacy TPS using ATML (Test Description)
- b) Extending a Legacy environment to support new Technology Insertion, using for example ATML Test Results and ATML Hardware descriptions, e.g. Instrument Description

Originally during the Phase I conception legacy issues were seen as a key use case; it may be necessary to review whether this is still a requirement.

5.3.1.12 Alternative Platforms – ARGCS, VDATS, RTCASS (low)

5.4 Lessons Learned

Ref	Title	Description	Resolution	Action
1	RemoveAll	In ATML Test Description add an Operation equivalent to REMOVE ALL. This disconnects and resets all active signals.	Add new Operation to 1671.1 Operations	IN
2	Combining Network Lists	<p>Combining Network Lists from different ATML files</p> <p>An issue was encountered when attempting to combine Network Lists from different ATML files to define complete signal paths from instruments to the UUT. There was no clear way to describe which interface ports from one file, such as the UUT Description, connects to interface ports from another file, such as Test Adapter.</p> <p>Since all the ATML files, UUT Description, Test Adapter, Test Station and Instrument Description are stand alone files there was found to be a need for a mechanism to assign interface ports from one ATML file to another to provide a complete system connect.</p>	<p>For the purposes of the Demo the interface ports for the two mating ports where named the same. This was a temporary work around however this problem requires a more formal solution.</p> <p>One suggestion was to add another element to the ATML Test Configuration schema which would provide for the mapping of the connections between the interface ports from the different ATML files used in a particular system or Test Configuration.</p> <p>This is now being discussed within SCC20 as a result of this Demo.</p> <p>Add a document reference attach an ID/Already in Common</p>	TL
3	Instance Attributes	<p>Test Configuration</p> <p>An issue uncovered is that it appears for a family of ATE (RTCASS for example) that when you include the Software (e.g. WindowsXP) that you need to provide attributes that represent instance (serial number specific)</p>	Submit proposal to IEEE &TII WG for Test Configuration Fixed – changed to Item description	CCG TL
4		Problems in mapping ATML (signal parameters) info onto the simplistic ATE API we had chosen	Recommended a Phase II task to show a more detailed mapping	CCG

ATML Demo Phase I - Technical Summary

5	Pins, Ports & Connectors	Explaining Pins & ports This was one are where we needed to keep explaining the standard	Review 1671 Base standard annex that deals with this topic to try and create a simpler introduction	CCG
6	Connector Mating	Connectors mating (especially UUT pins)	New Schema and reference in Test Configuration	TL CCG
7	IEEE Std 1641 Compatibility	When the updated IEEE Std. 1641 files where used. getting a validation error in the STDTSFLib.xsd.	Review Compatibility between standards	IN CCG
8	Measuring Active Components	Measuring a Short on an Active component (using Resistance as a current source)	Add example into 1641 standard	IN CCG
9	Test Description/ Results Compatabilities	TestDescription only requires a Model Name for the UUT TestResults requires a Serial Number		IN
10	TestDescription Outcome	TestDescription requires a Test OutcomeValue What are the valid values for this? TestDescription requires a Test OutcomeValue. TestResults requires a Test OutcomeValue also. Both state that they are an enum but the type is a string. What are the valid values for this? Pass/Fail/Aborted seem to be the expected for the TestResults and list of UUT faults in TestDescription?		IN
11		TestDescription does not require a description of the test. TestDescription does not require a description of the test. It only requires the name and ID of the test. So the TestDescription file does not require the creator of the file to specify anything on the test or what the expected values of the test are.	Test description has a life cycle model where does test description fit in	IN
12		TestResults only require the outcome to be recorded. It does not require the actual and expected values to be recorded. TestResults only require the outcome to be recorded. It does not require the actual and expected values to be recorded. The implication being if the data is not there then it will not be viewable.	See what happens under review cycle	

ATML Demo Phase I - Technical Summary

13	Test Description Example2	<p>Example 2 - An different approach to Signal parameters</p> <p>This issue was discussed during ballot resolution and considered the idea of creating a new mechanism for specifying test behavior, where a Run operation would execute an entire 1641 test definition, including stimuli, measurements and any other things that can be defined in 1641. This may be a good time to actually try that approach, especially as we aim to create samples that will be included in the published standard. Using this type of solution in the current Example 2 would eliminate the implementation difficulties, as both Parameters and Test Results would be defined as scalars and mapped to 1641 attributes.</p> <ul style="list-style-type: none"> - Make a small change to the schema, creating a new type of Operation allowed under Behavior (or maybe use the extension mechanism, if we want to stick to the current version of the schema). This Operation may be called Run and would contain a complete 1641 test definition. - Create an instance document that has test descriptions similar to Examples 1 and 3 (i.e., with scalar parameters and test results), and Behavior defined through the above Operation. For each test, the Operation would contain a 1641 XML test definition that combines the Parameter and Test Result XML fragments that already exist in Example 2. - Use the mechanisms already in place (shown in Example 3) to map signal attributes and measured characteristics to the scalar parameters and test results. - Import the new file in TestStand. As Behavior XML ends up as code comments, hopefully no change is needed at this level (other than maybe accommodating the modified schema). - Use newWave to generate C code from the 1641 test definition embedded in the Run Operation. I hope this will be easier than having to process Test Description XML with 1641 code embedded. 	Example Update Test Description during Ballot Resolution	IN CCG
14	UUT Description	<p>Abstract Types in UUT Description</p> <p>In doing the files for ATML demo we noticed that the changes to hardware common (specifically HardwareItemDescription is now abstract. This prevented valid UUT Descriptions being generated since it uses a Hardware element of that type, but does not define any derived type to use. The documentation of HardwareCommon suggests UUTDescription should have been a derived type.</p>	New UUT description in the pipeline V 1.05	TL

ATML Demo Phase I - Technical Summary

15		<p>Consider adding an element containing input impedance to UUT Description.</p> <p>There is no place in the ATML UUT Description to contain input impedance data for UUT pins. This information is useful in certain situations to calculate applied voltage levels where the output voltage of an instrument is dependent on the relationship between the instrument source impedance and the UUT input impedance.</p> <p>There was discussion also regarding how extensive to model the UUT and how to ensure the vendor supplies this data.</p>	<p>We believe this is defined under Test Description.</p> <p>The formal location of this information should addressed as we normalise the ATML information</p>	IN/CCG
16		<p>Here are a couple of issues brought to while working on the RAI demonstration at ATC 2008.</p> <ol style="list-style-type: none"> 1. Developers indicated it would be nice to be able to identify an ITA pin as belonging to the UUT side of a fixture rather than the tester/ICA side of a fixture without tracking around in other instances. 2. Developers noted that certain UUT loading, impedance and other parametric issues were not supported by the UUT 3. Developers displayed a natural propensity for EEs to imply a networking mindset to signal models and their connections. 	<ol style="list-style-type: none"> 1 – See 15 2 – Review use in test description to all hardware standards during transition to full standard 3 - Yes 	ALL

6 Technical Report

6.1 Demonstrating A Complete Testing Scenario

6.1.1 Introduction and objectives

The following example is the output from the ATML phase 1 demonstration effort of 2008. This demonstration successfully showed the operational benefits of utilizing ATML standards in a real testing scenario. This was accomplished through the utilization of XML instance documents that were based upon the individual ATML standards, by COTS software products, to successfully test and diagnose a UUT on a particular ATE.

This example outlines a scenario where three system level use cases (See [6.1.1.1](#)) are developed utilizing a majority of ATML's family of standards (See [6.1.1.2](#)), and describes both the hardware and software design and development tasks involved with implementing the testing scenario (See [6.1.2](#) through [6.1.3](#)).

6.1.1.1 System level use cases

This examples system level use cases are to:

- a) Provide all the necessary test information, in a reusable format, to take a Test Description in the form of UUT Test Requirements and convert it into a test program (in an ATE oriented language) running on a fielded ATE system.
- b) Use ATML components to compare requirements and capabilities as well as calculate switching paths.
- c) Collect test results and use them in various use cases such as statistical analysis and displaying them to the operator.

6.1.1.2 Utilized standards

This example is based around ATML family standards that have been published, are being balloted, or are in the IEEE balloting process. Specifically, the following IEEE standards and IEEE standards projects are included in this example:

- | | |
|--------------------|--|
| a) IEEE Std. 1671 | ATML Overview and Architecture |
| b) IEEE-P1671.1 | ATML: Test Descriptions |
| c) IEEE Std 1671.2 | ATML: Instrument Descriptions |
| d) IEEE Std 1671.3 | ATML: UUT Description |
| e) IEEE Std 1671.4 | ATML: Test Configurations |
| f) IEEE Std 1671.5 | ATML: Test Adapter Descriptions |
| g) IEEE Std 1671.6 | ATML: Test Station Descriptions |
| h) IEEE Std 1636.1 | SIMICA: Test Results and Session Information |
| i) IEEE Std. 1641 | Signal and Test Definition (STD) |

6.1.2 Hardware test environment and UUT design

The hardware elements designed, chosen, and utilized (UUT, ATE, ITA) for this example are described in [6.1.2.1](#) through [6.1.2.3](#).

6.1.2.1 UUT hardware and test strategy

For the purpose of this example, a “simple” low frequency analog UUT is designed and developed. This dedicated UUT allows both parametric testing and diagnostics through fault simulation, allows for the generation of an ATML UUT Description XML instance document, permits for the test execution on a variety of ATE stations (UUT testing only requires commonly available stimulus and measurement resources – there are no digital, bus communication, RF, etc. requirements), and allows faults to be inserted during execution of tests through the activation of switches incorporated into the UUT design.

The UUT schematic and connector pin definitions are depicted in [Figure 4](#). The UUT connector (and the mating connector that will be part of the ITA) itself is a commercially available D connector containing both low frequency signal and coaxial contacts.

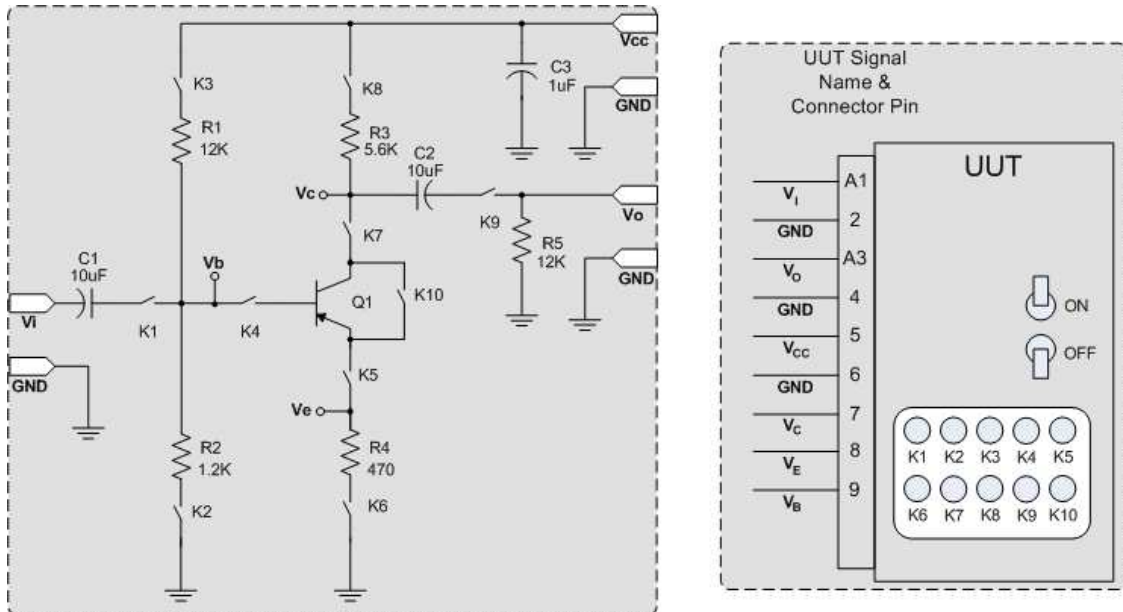


Figure 4—UUT’s schematic and UUT’s connector pins

ATML Demo Phase I - Technical Report

The testing and fault isolation strategy of this UUT is depicted in [Figure 5](#). This strategy is utilized for the ATML Test Description XML instance document generation, as well during the ITA design (and thus the associated ATML Test Adapter Description XML instance document generation).

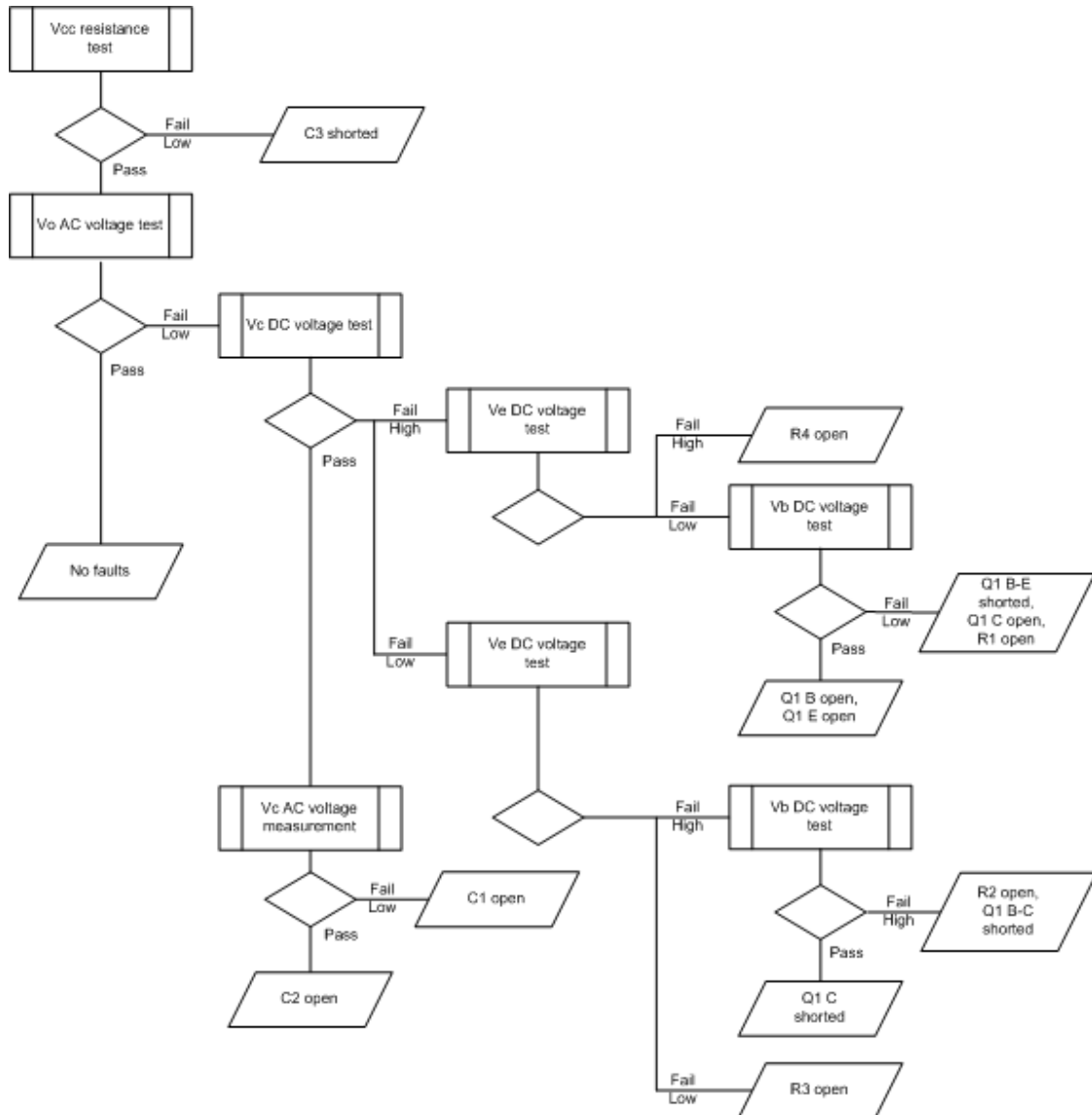


Figure 5—UUT's testing strategy

The UUT is defined within an ATML UUT Description XML instance document. This ATML UUT Description instance document shall be available at:
<http://grouper.ieee.org/groups/scc20/ATML/Demonstrations/PhaseI>

Also, as a note, this UUT is also represented within Annex B of IEEE P1671.1 (ATML Test Descriptions) further detail on the UUT can be found in Annex B of IEEE P1671.1.

6.1.2.2 ATE and test configuration

An ATE was chosen from the DoD approved families of automatic test equipment, although any target platform test station could be used to support both development and the full demonstration which provides the station test resources and supports the types of tests (this assumes however that the target platforms

ATML Demo Phase I - Technical Report

electrical interface is identical, or that the ITA could either be mechanically and electrically adapted to the target platform or developed specifically to the target platform).

The necessary station test resources are those for:

- a) UUT power.
- b) Low frequency signal source.
- c) Simple analog measurements.
- d) Measurement analysis.
- e) Switching.

These test resources as well as the station control is portion of the ATE is represented in [Figure 6](#).

The chosen ATE contains a well documented station interface (both hardware and signal) pin-out definition, has a station control that is PC based; is running a widely use COTS operating system. (allowing the ATML tools to be easily hosted/execute on the ATE itself), and contains the necessary test resources to provide the stimulus and measurement capabilities required to implement the test program that will be derived from the ATML Test Description (See [6.1.2.1](#)) in a ATE oriented language. The stimulus and measurement resources are electrically accessible directly as feed-thru's at the stations interface receiver. This ATE additionally contains low-frequency, coaxial, and power switching; which permits the test assets to be routed to multiple locations as required by a testing implementation, within the ITA.

With a general understanding of the UUT described in [6.1.2.1](#), and a general understanding of the chosen ATE, a ITA can then be designed and developed to interface the two (See [Figure 6](#), and [6.1.2.3](#)).

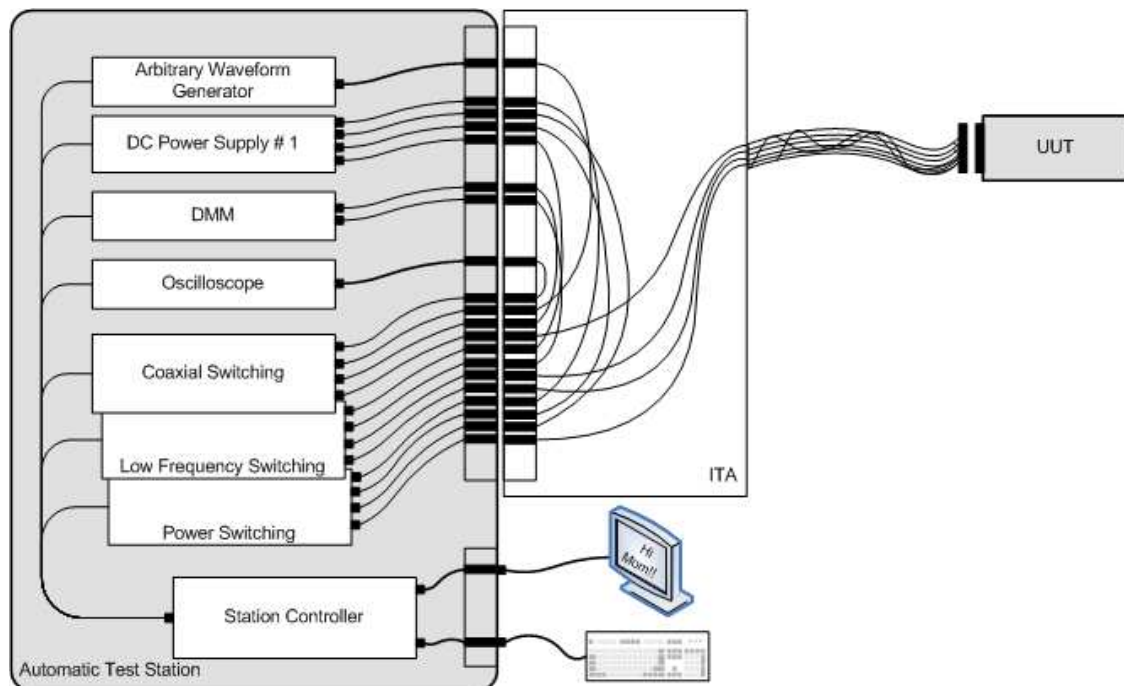


Figure 6—UUT's hardware testing configuration

The utilized portions of the ATE plus the ATS Capabilities are defined within an ATML Test Station Description XML instance document. This ATML Test Station Description instance document shall be available at:

<http://grouper.ieee.org/groups/scc20/ATML/Demonstrations/PhaseI>

ATML Demo Phase I - Technical Report

Each of the utilized instruments functions is defined a within ATML Instrument Description XML instance documents. Each of these ATML Instrument Description instance documents shall be available at:
<http://grouper.ieee.org/groups/scc20/ATML/Demonstrations/PhaseI>

The configuration depicted by Figure 6 (As well as the software installed on the station controller – operating system, ATML tools, COTS software, ect. – e.g., the ATE control software, ATE support software, and ATE system software) is defined within an ATML Test Configuration XML instance document. The ATML Test Configuration instance document shall be available at:
<http://grouper.ieee.org/groups/scc20/ATML/Demonstrations/PhaseI>

6.1.2.3 Interface test adapter

For the purpose of this example, an ITA is designed and developed to electrically interface the UUT (See [6.1.2.1](#)) to the chosen ATE (See [6.1.2.2](#)).

Utilizing the ATE's stimulus, measurement, and switching capabilities on one hand, and the UUT's connector/mating connector pin-out and test strategy on the other hand; an ITA concept can be developed. Each of the tests in the strategy is manually “mapped” to test resources (through the utilization of switch paths, which allow for multiple resources to be routed to the same UUT pin, or for resources to be totally disconnected, for example). The “completion” of each of the test strategies mappings to ATE resources, with each mapping “overlayed” on each other, results in the ITA interconnections depicted by [Figure 7](#).

Note that in developing the ITA concept, the designer “in their head” has assigned resources and utilized switch paths to route signals to/from the UUT connector pins; so that they “know” that each test within the testing strategy can be electrically connected. This “knowledge” is obviously not known by the ATE station software unless there is a method of providing the information to the software. Thus the importance of the ATML Test Adapter Description standard, and the contents of the associated XML instance document representing the ITA.

ATML Demo Phase I - Technical Report

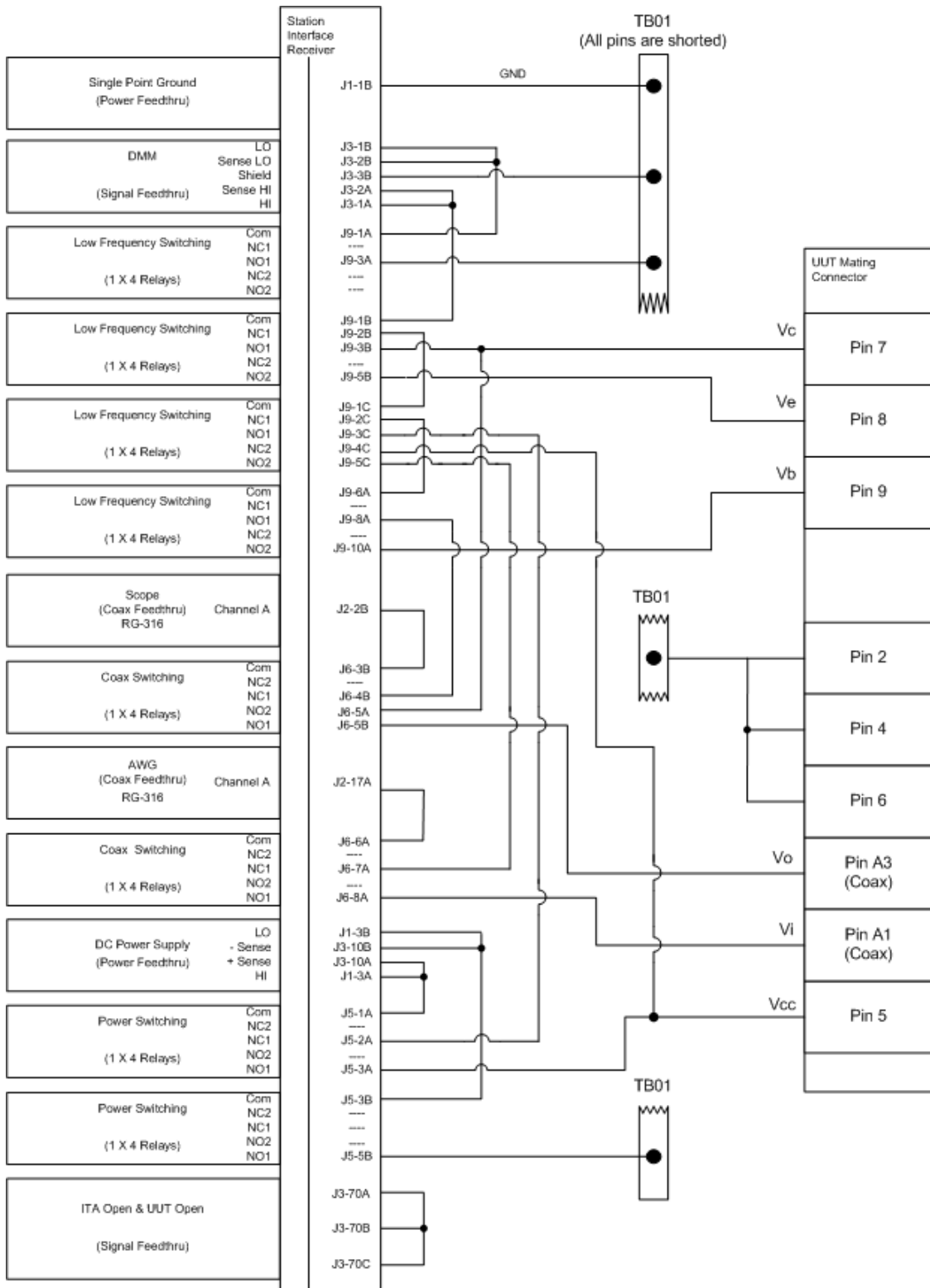


Figure 7—ITA interconnections – ATE resources to UUT mating connector pins

The ITA is defined within an ATML Test Adapter Description XML instance document. This ATML Test Adapter Description instance document shall be available at:

<http://grouper.ieee.org/groups/scc20/ATML/Demonstrations/PhaseI>

6.1.3 Software test environment design

The software environment that is utilized for the development and execution of the UUT's test program is depicted by Figure 8. All "non-shaded" elements of Figure 8 represent ATML family standards. Shaded elements represent tools, COTS software, and software outputs from the execution of tools.

6.1.3.1 through 6.1.3.2 further detail the ATE support software, ATE system software, and the ATE control software. For the purpose of the example, the TPS will be a C/C++ program.

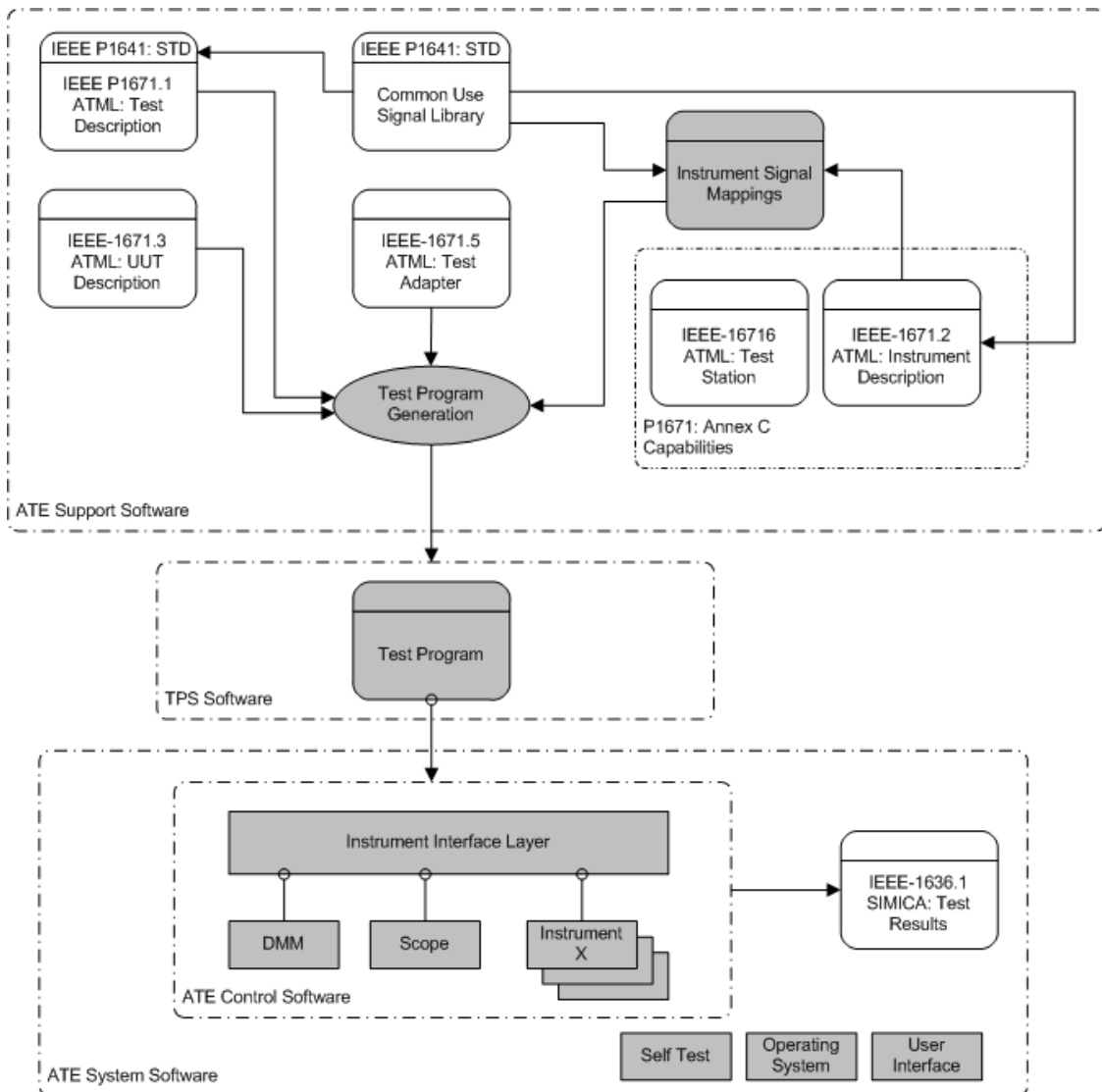


Figure 8—UUT's software testing configuration

6.1.3.1 ATE support software

ATE support software consists of the software which aid in the preparation, analysis, and maintenance of UUT test programs. The ATE support software typically is available "off station", however, there may be instances where the ATE support software is also available on the target ATE. Within the scope of this example, the location of the ATE support software is irrelevant.

ATML Demo Phase I - Technical Report

In order to actually conduct tests on the UUT (See [6.1.2.1](#)) utilizing the hardware items (ATE and ITA) described in [6.1.2](#), a UUT test program needs to be “developed” from the UUT testing strategy (See [Figure 5](#)), so that it may be “executed” by the ATE station control software.

Historically, the elements of a particular ATE’s support software incorporated “interpretations” by the ATE developers of such items as instrument’s vendor data sheets/documentation, ATE hardware design material, etc. These “interpretations” are effectively turning one data format into a second (usually proprietary) format (e.g., A instrument’s data sheet contents in PDF format, put into a compilers instrument database’s “unique” file format). This usually always “loses something in the translation” as information is lost, doesn’t “have a home” etc. Thus the importance of ATML’s Test Station Description and Instrument Description standards and the contents of the associated XML instance documents that can be shared between vendors and ATE organizations; eliminating any need to “interpret”, or to utilize proprietary formats.

For this example, COTS software products (See [6.1.3.1.1](#)) that utilize/incorporate ATML XML instance document data and STD (IEEE Std.1641) based signal modeling concepts (See [6.1.3.1.2](#)) can be “assembled” to allow for the development of an ATE oriented language test program for the UUT.

6.1.3.1.1 COTS software products

This example was reliant on the use of COTS tools already developed to create and consume the ATML information as part of the operational ATS. Prototype COTS tools were acceptable as long as documentation or support is available, and the tools comply with the conformance section of this standard. ATML tools may consist of any application that produces, translates, or consumes the ATML format (editors, display tools, converters, database, etc.). This is a technological area where it is expected that a wide range of diversity and innovation will occur within COTS products, while supporting the interchangeable ATML format.

The COTS software products included in the ATE support software are:

- a) STD based signal modeling tools.
- b) ATML test and instrument description tools.
- c) Test executive ATML importer tools.
- d) Switch path analysis tools.
- e) C/C++ language compilers and linkers.

6.1.3.1.2 ATML and STD signal modeling data

The ATML information utilized by these COTS software products includes:

- a) STD based signal model libraries.
- b) ATML test description XML instance documents.
- c) ATML instrument description XML instance documents.
- d) ATML test station description XML instance documents.
- e) ATML test adaptor description XML instance documents.
- f) ATML UUT description XML instance documents.
- g) ATML capabilities.
- h) ATML test configurations.

This support software architecture, in the form of an “information flow” diagram, is depicted by [Figure 9](#). Various tools (as depicted by the shaded oval shape) are utilizing various ATML and STD information/libraries (as depicted by the shaded “document” shape).

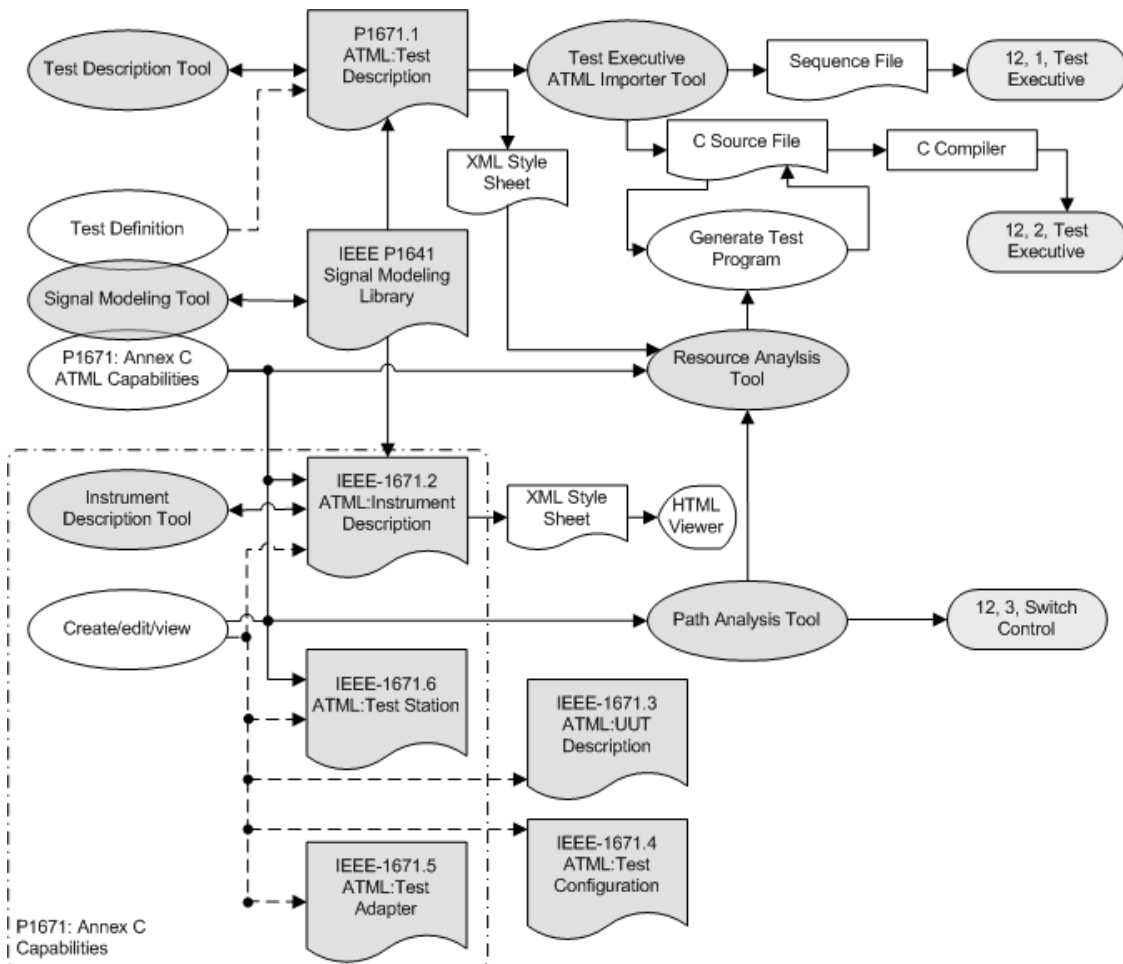


Figure 9—“Support” software architecture

ATML Demo Phase I - Technical Report

ATML capabilities are used as the “glue” to allow the UUT’s signal requirement to be mapped to the ATE test resource (ultimately to the ATE connector pins) that can provide the capability through the ATE and ITA to the required UUT connector pin. This is depicted by [Figure10](#).

The ATE support software provides for the creation, editing, and viewing of this “glue”.

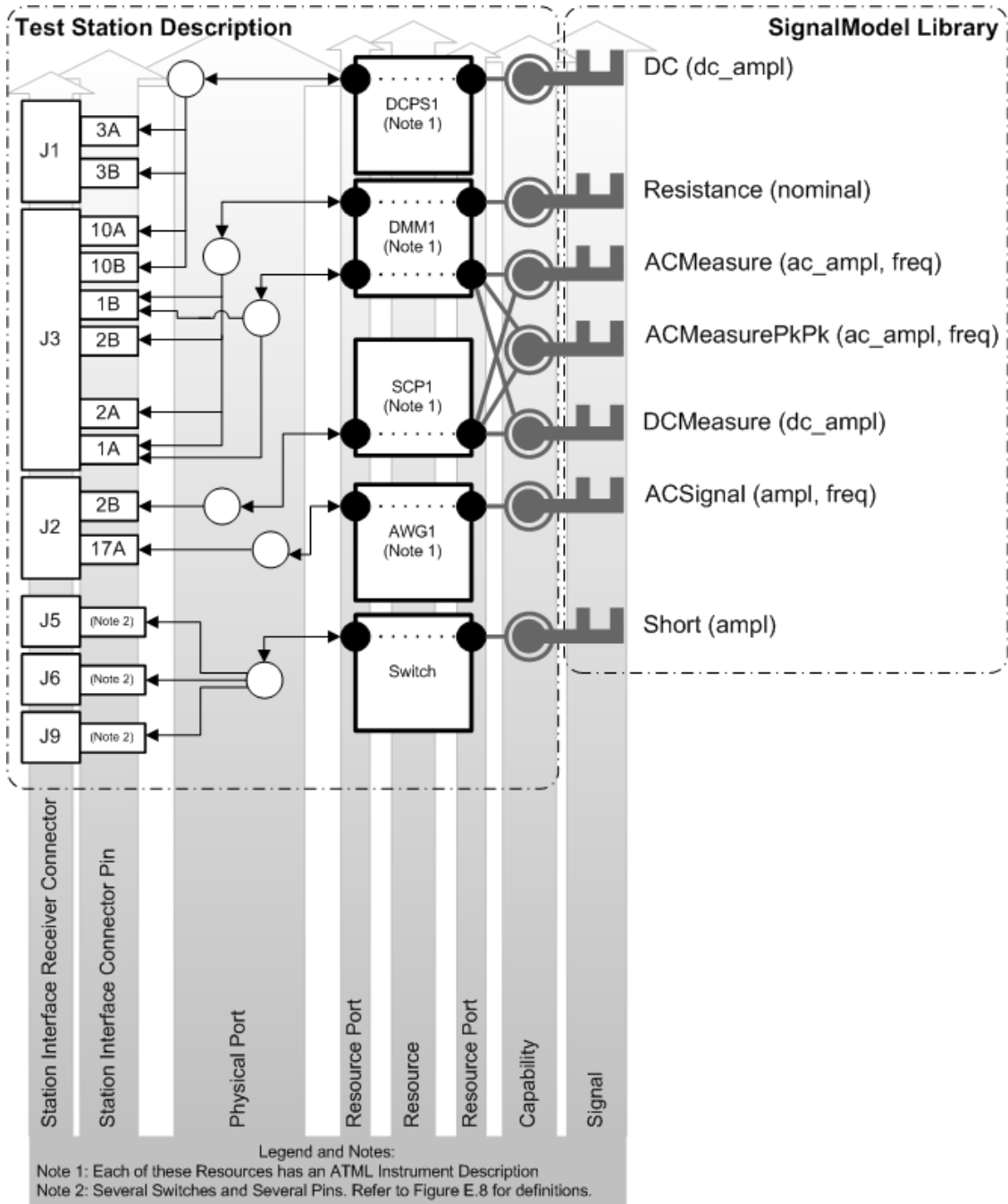


Figure 10—Resource capabilities

6.1.3.1.3 Support software outputs

The “outputs” of the tools in [Figure 10](#) collectively result in the following (which will be utilized by the ATE control and system software (See [6.1.3.2](#)). These three outputs are the three inputs of [Figure 11](#).

- a) A sequence file for use by the test executive.
- b) The generation of the test program (which is then compiled and linked with a C/C++ compiler).
- c) Switch control information.

XML style sheets and HTML viewers are used to display to the user the XML information.

6.1.3.2 ATE control software and ATE system software

ATE control software is used during the execution of a UUT test program, to control the non-testing operations of the ATE. This software is used to execute a test procedure but does not contain any of the stimuli or measurement parameters used in testing the UUT.

ATE system software is the total software environment of the ATE including operating system, test executives, user interface, system self-test, and other software required to run UUT test programs. For the purposes of this example, the PC’s operating system, ATE system self-test, and user interface will not be further described.

In order to run the tests on the UUT, utilizing the ATE and ITA, the UUT test program created via the ATE support software (See [6.1.3.1](#)) needs an environment to be “executed” from.

Historically, the elements of a particular ATE’s station control software interfaced with and produced data in proprietary formats (test program intermediate programming languages, test results, etc.) This meant that only the “matching” ATE support software could be utilized, and test results were represented/store in a format unique to that ATE. ATML permits the actual test program to be implemented in the “language of choice”. Therefore, the complimentary test execution environment for that “language of choice” must be an element of the ATE station control software. Thus the importance of ATML’s reference to the SIMICA test results and session information standard and the contents of the associated XML instance documents that can be shared between vendors and ATE organizations; eliminating any need to utilize proprietary formats for the recording of UUT test results.

For this example, COTS software products (See [6.1.3.2.1](#)) that utilize/incorporate ATML component standards (See [6.1.3.2.2](#)) can be “assembled” to allow for the execution of an ATE oriented language test program for the UUT, create and view UUT test results, and provides an interface to reasoner based test execution.

6.1.3.2.1 COTS software products

This example is reliant on the use of COTS tools already developed to create and consume the ATML information as part of the operational ATS. Prototype COTS tools are acceptable as long as documentation or support is available, and the tools comply with the conformance section of this standard. ATML tools may consist of any application that produces, translates, or consumes the ATML format (editors, display tools, converters, database, etc.). This is a technological area where it is expected that a wide range of diversity and innovation will occur within COTS products, while supporting the interchangeable ATML format.

The COTS software products included in the ATE control software are:

- a) Test Executive.
- b) Switch path control.
- c) Instrument support handlers.
- d) Instrument drivers.
- e) Virtual instrument software architecture.
- f) Bayesian reasoner.
- g) Test results analyzers and viewers.

6.1.3.2.2 SIMICA data and AI-ESTATE models

The ATML information utilized by these COTS software products includes:

- a) SIMICA test results and session information.
- b) The AI-ESTATE Bayesian reasoner model.

This station control software architecture, in the form of an “information flow” diagram, is depicted by [Figure 11](#).

As can be seen by [Figure 11](#), the graphic visualization and test result analyzer tools (as depicted by the shaded oval shape) are utilizing SIMICA test results information (as depicted by the shaded “document” shape).

6.1.3.2.3 Station control software inputs

The “inputs” of the tools in [Figure 11](#) collectively result from ATE support software outputs (See [6.1.3.1](#)). These three inputs are the three outputs of [Figure 9](#).

- a) A sequence file for use by the test executive.
- b) The compiled and linked C/C++ language UUT test program.
- c) Switch control information.

XML style sheets and HTML viewers are used to display to the user the XML information.

ATML Demo Phase I - Technical Report

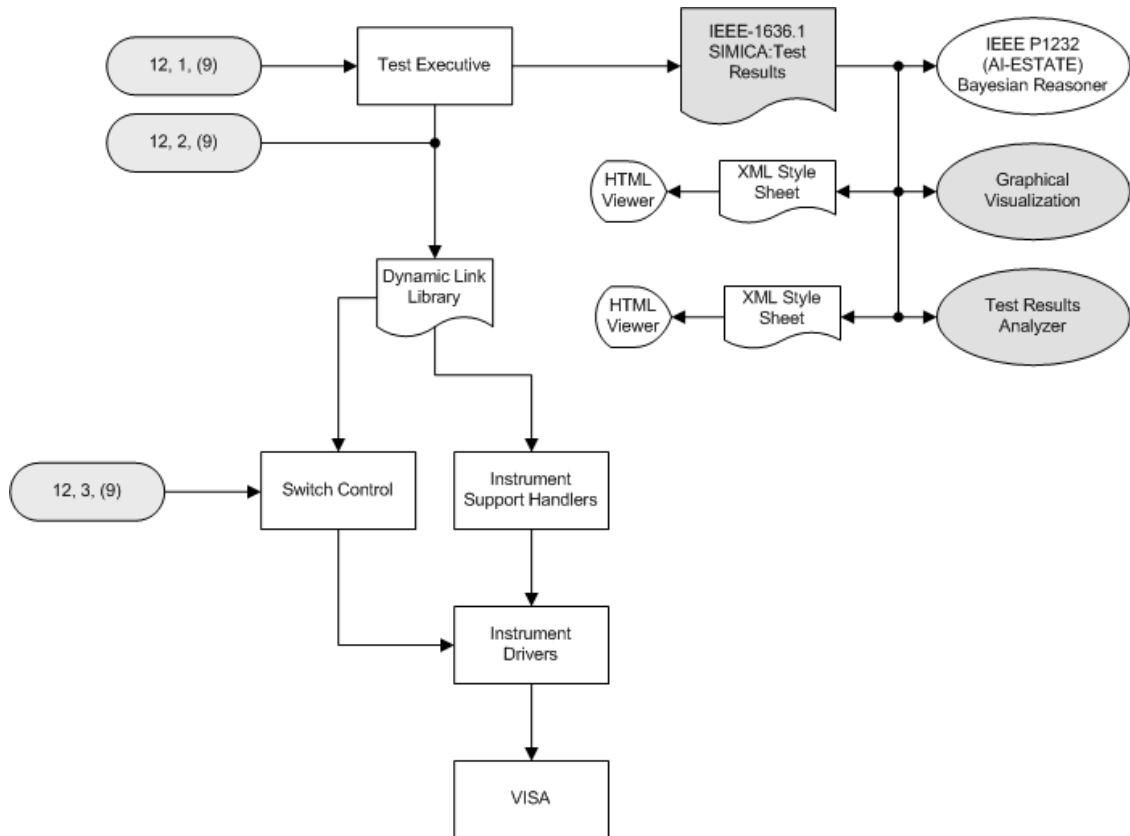


Figure 11—"Station control" software architecture