



APEX

Technical Summary

Author: I Dankiewicz
Reference: SSSL/S7726/TEN/014
Issue: 1.0
Date: 08/06/2004

SciSys (Space & Defence) Ltd
Methuen Park
Chippenham
Wiltshire
SN14 0GB
United Kingdom

Tel: +44 (0)1249 466466
Fax: +44 (0)1249 466661

www.scisys.co.uk

1. INTRODUCTION

The Automated Procedure Execution (**APEX**) system is a portable procedure definition and execution tool that automates spacecraft testing and operations. Its functionality, performance and scalability allow it to support any mission configuration.

Operations automation is a key area where SciSys has successfully responded to customer demands for greater automation of legacy or new control systems. Control system automation decreases the work load for operations staff which reduces manning profiles and leads to significant mission life time cost savings.

APEX provides a generic environment designed to require minimal integration and configuration effort to deploy for each new mission. Emphasis is placed on a low deployment and lifecycle costs and minimal resource footprint. APEX is based on Java technology and open standards, without the use of costly 3rd party COTS software environments.

APEX builds on SciSys' extensive experience and investment in providing automated monitoring & control systems, most recently through deployment of the UNiT toolkit for EUTELSAT, the UK MOD and EUMETSAT. In particular, APEX uses the proven conceptual model of UNiT automated operations procedures. Gensym's G2 used for Operations Language (**OL**) execution in UNiT has been replaced by SciSys' Java-based Integrated Common Operations Language (**ICOL**). The UNiT procedure model has been modified and re-implemented for lightweight execution of individual procedures, with a clear networked separation between the procedure execution engine server functions and support for client visualisation and display.

The APEX architecture also supports the partitioning of the procedure execution environment into Domains and their distribution over multiple server hosts. This approach enables distributed execution of procedures, resulting in a flexible solution which is fully scalable for deployment in a multi-satellite fleet or constellation control context.

The procedure definition format has been modified to increase compatibility and interoperability with the draft PLUTO procedure language standard currently being specified by the European Committee for Space Standardisation (**ECSS**). An area in which the Pluto specification has been used by APEX to improve UNiT is that of automated recovery from failures detected during procedure execution.

2. APEX MODEL OF PROCEDURE EXECUTION

APEX Procedures correspond to pre-defined operational activities that can either be scheduled as a ground-based activity, or manually initiated. Procedures may also call other Sub-Procedures, permitting their decomposition into smaller, more maintainable units, which can be reused in the context of several different operations.

The definition of a procedure comprises its public interface (including arguments), local variables and a set of Thread definitions with their constituent Steps shown in Figure 1. Procedures contain a single Primary Thread and, optionally, a number of Secondary Threads. Each Thread constitutes an independent flow of control through the procedure. Procedures are also defined to execute with in a particular Domain which is a level of application sub system decomposition defined in the Space System Model (SSM).

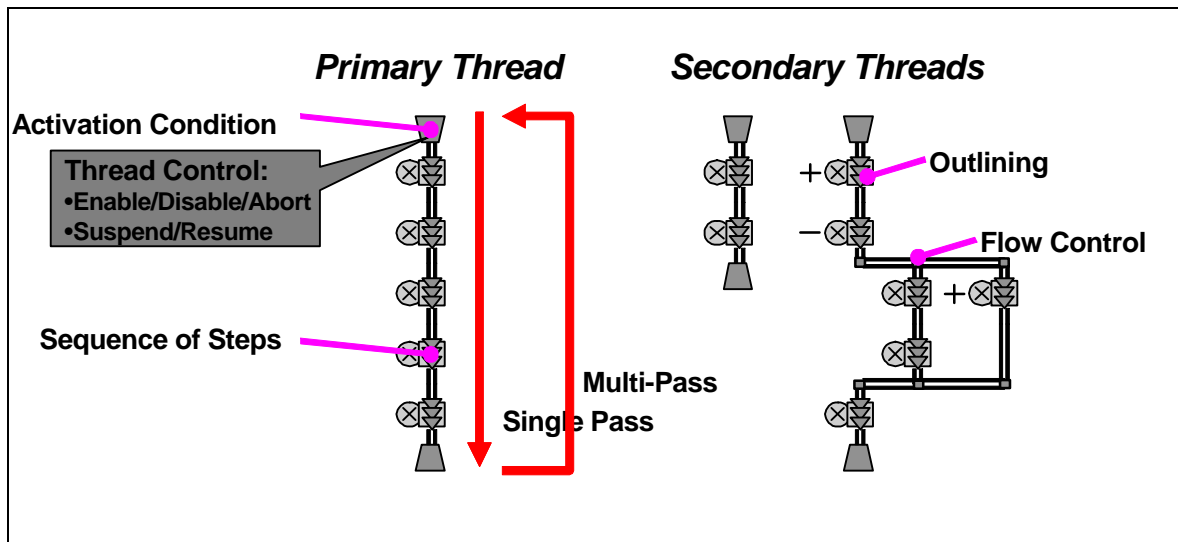


Figure 1 APEX Procedures and Threads

Steps shown in Figure 2 comprise a trigger condition, body and confirmation condition. The body of a step may constitute either:

- An action such as sending a command, asserting a system parameter value, raising an event etc
- A flow control construct such as branches (If and Case), loops (While, Repeat), Failure Recovery etc
- An interaction with the operator allowing manual input of data or decisions to be made.

Each Step is broken into three clear phases that of triggering, body execution and confirmation shown in Figure 2.

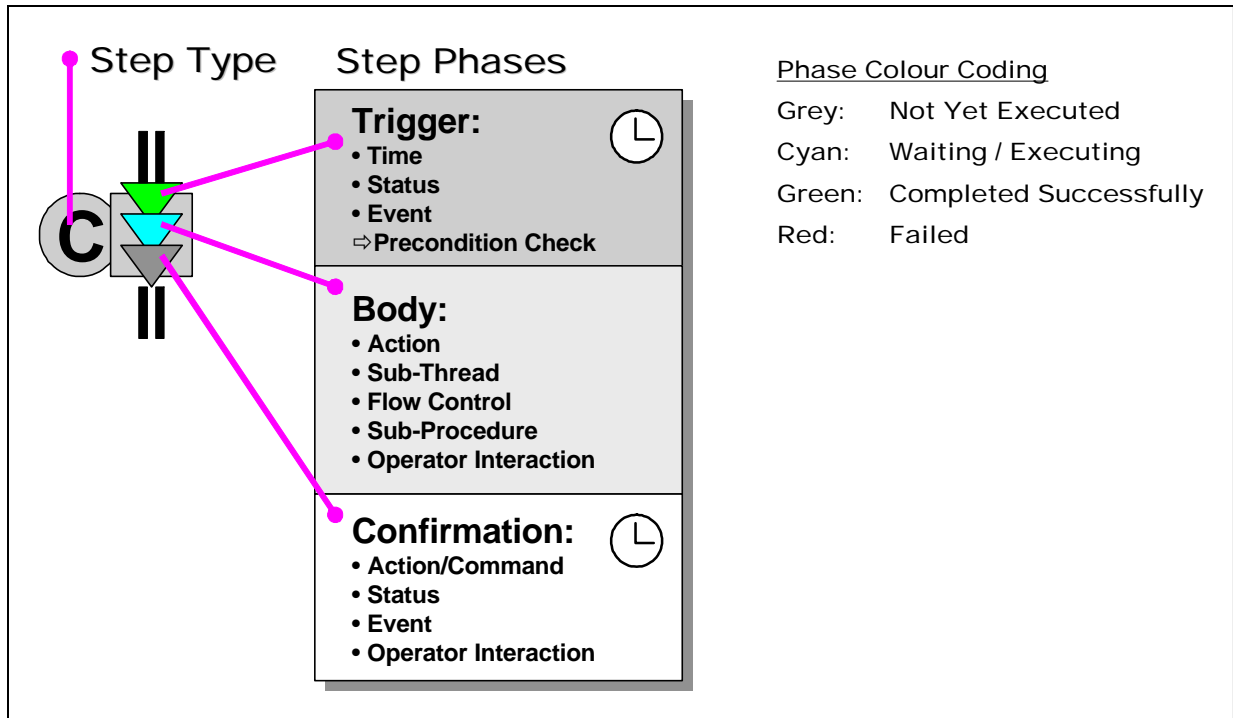


Figure 2: APEX Procedure Steps

Triggering - A Step may have an optional Trigger which controls the synchronisation and checking that conditions are correct to start execution. A Trigger consists of a Wait Condition and a Pre Condition. On activation the Step will await its Trigger Wait Condition. This is followed by the checking of the Pre Condition before progressing on to execute the Body of the Step.

Body - Execution of the Step Body involves execution of the intended Step logic e.g. execution of an Application Action or flow control construct such as a loop or branch statement. An optional Watch Condition can be associated with the Step Body which is used to check that execution is proceeding normally such that, on failure, suitable Failure Recovery contingency handling can be applied.

Confirmation - Once the body is complete then an optional confirmation phase of the current step can be executed before moving on to execute the next Step of the sequence.

4. PROCEDURE EXECUTION ENVIRONMENT

APEX supports the execution of procedures in the context of a particular subsystem of the Space System Model (referred to as a Domain). The association of procedures to Domains and Sub-Domains allows the logical partitioning of a particular automated application into a subsystem hierarchy. The concept of Domains also supports the physical partitioning of procedure execution on different server nodes of a computer network.

The APEX architecture supports the distributed execution of procedures by allowing each server node in a network to be configured to host a different sub set of the Domains and Sub-Domains defined in a Space System Model (**SSM**). For example the top level Domain of a M&C system could represent a particular spacecraft in a constellation or fleet. Figure 4 shows the flexibility and scalability of APEX which would allow each Domain to be executed on a different server node or for different subsets of Domains to execute on the same node.

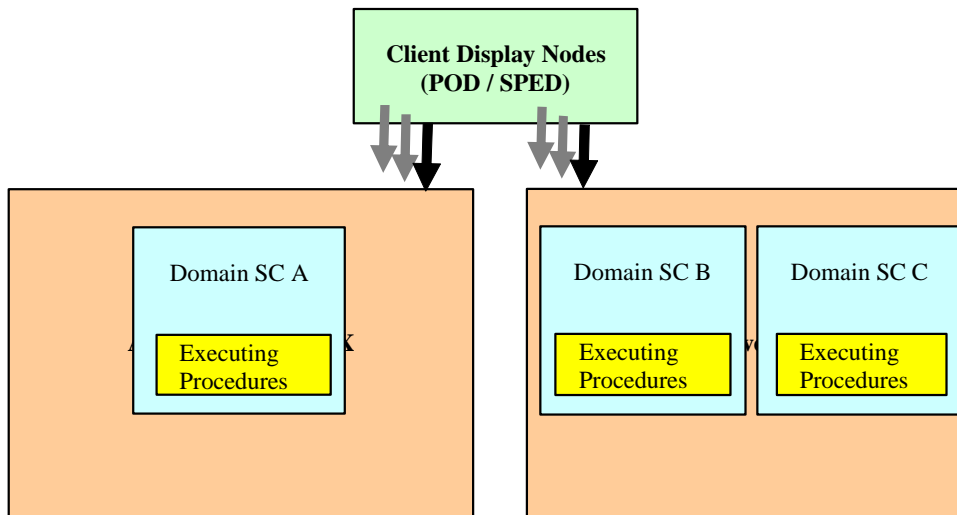


Figure 4 APEX Distributed Architecture & Domain Configuration

APEX supports control of procedure execution at the level of the three entity types shown in Figure 4 i.e. Server Node, Domain and Procedure. The networked architecture allows the connection of multiple client display nodes to the server nodes and the creation of the node, domain and procedure control interfaces. The client display nodes host the Procedure Overview Displays (**POD**) Figure 5 and Single Procedure Execution Displays (**SPED**) Figure 6.

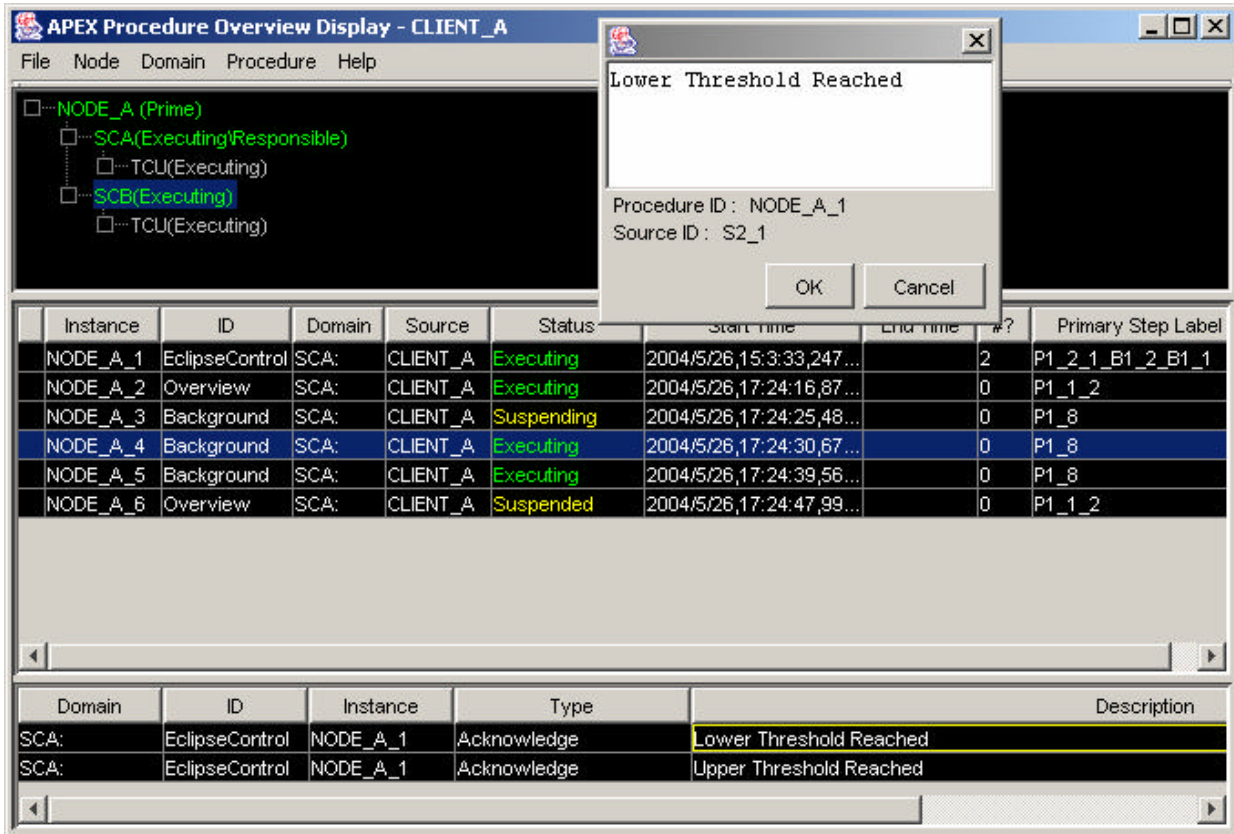


Figure 5 Procedure Overview Display

The POD shows an overview of Node, Domain and Procedure High level control information.

The graphical SPED can be launched from the POD. It shows the detailed execution status of an executing procedure showing its threads, steps and automatically tracks the current execution point.

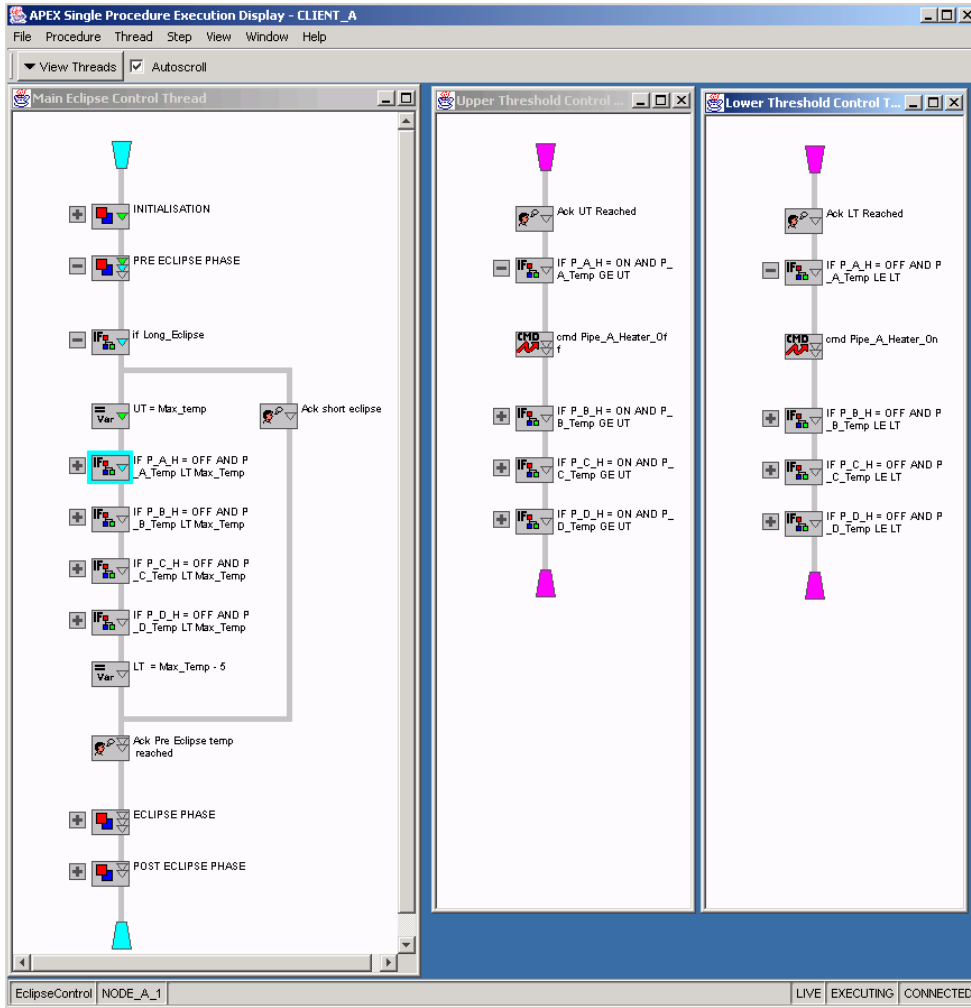


Figure 6 Single Procedure Execution Display

5. CONCLUSION

APEX provides a best-of-breed solution for automated procedure execution in operational and test environments. It builds on the considerable, proven success of UNiT with use of the latest open technologies and standards.

The APEX environment supports both the definition and execution of procedures which allow the logic of automated operations to be captured. Procedure definition uses XML and XSD technology which is also supported by standard low cost COTS XML editors. Support for external languages, PLUTO in particular, is included.

SciSys is currently developing graphical editors for APEX comparable to the UNiT graphical editors and to complement the existing APEX editors based on COTS.

APEX delivers a distributed networked procedure execution architecture which is based on Java technology. This architecture is scalable from EGSE test scripts for a small payload to critical mission operations for a large constellation. Procedures execute in a hierarchy of domains which may be distributed over a network of server nodes and controlled by portable graphical displays. The graphical displays of procedure execution provided allow an operator at a single work station to monitor the progress of multiple procedures executing in a complex domain hierarchy distributed over many server nodes.