NOVEL LANGUAGE FEATURES IN OPAL

H. Kaunzinger
CENTACS
and
R. Turkington
RCA Corporation

ABSTRACT

The Operational Performance Analysis Language (OPAL) development started in the early seventies in an evolutionary process from its predecessor, the COMPASS Test Language (CTL). The language philosophy and initial definition must be credited to Massachusetts Computer Associates (COMPASS) under U. S. Army contracts for several years. The development was initially funded by Frankford Arsenal, Philadelphia and later by the Center for Tactical Computer Systems (CENTACS), Port Monmouth, NJ. The final 18 month phase of the OPAL development under contract with RCA was completed in March 1978 and has culminated in a final definition of OPAL containing all the language facilities needed for efficient computer implementation and execution.

OPAL can be kept a stable standard because the general-purpose core constructs are in the top levels of the syntax pyramid while nouns, modifiers, units, and pin descriptors, which must be flexible to express state-of-the-art advances in testing and test hardware, are low-level terminal language elements not affecting the language stability.

This paper is broken down into two categories, a general language description and a specific treatment of novel language features not available in other test languages.

General Language Description:

OPAL has been developed based on testing experiences gathered throughout the existence of the ATLAS committee, originally administered by the Aeronautical Radio Inc. (ARINC) but sponsored by the IEEE since 1976. OPAL has been designed as an efficient test programming language containing modern computational and control capabilities similar to the features of modern computer languages (such as ALGOL, PL1, or PASCAL) and containing test capability equivalent to that of ATLAS. However, the test statements and structure of OPAL are virtual-resource oriented, not signal oriented, thereby eliminating ambiguities and statement detail associated with the signal model.

The general language description covers the following:

- Test program structure, which is based on a set of separately compiled modules which may contain any number of main or subsidiary subroutines.
- Test statement structure, which is based on the ATLAS test statements but with more consistent definitions and format.
- Computational capability, including built-in functions.
- Control capability which supports structured programming.
- System Input/Output capability, which is based on a virtual resource concept.

This section also includes a description of module utilization with export and import rules for variable and resource names and a general arrangement of programs and program flow. The general description also includes details of the virtual resource characteristics in REQUIRE statements, description of needed information of the unit under test (UUT) or the adapted UUT at the ATE/adapter interface with IDENTIFY and SPECIFY statements, and description of the means of assigning the general-purpose virtual resources to the existing resources of a specific ATE (Automatic Test Equipment) system. The last item in this general description is the range of nouns, modifiers and units required for coverage of a wide range of test applications.

The novel features in OPAL included:

- An expanded set of virtual resource types:
  - Source: Synchronizer
  - Sensor: Signal Conditioner
  - Load: Continuous Monitor
  - Switch: Operator Control
  - I/O Device

The Synchronizer is fundamental to the
OPAL approach to signal timing and synchronization. The Signal Conditioner allows the programming of complex and specialized signal types. The Continuous Monitor, modeled after existing test capability, eliminates much of the need for parallel processing in the ATE system.

- An expanded set of virtual resource commands (single-action verbs):
  - SETUP OPEN
  - CHANGE START
  - RESET STOP
  - CONNECT WAIT
  - DISCONNECT SENSE
  - CLOSE SWITCH

These commands, augmented by START and STOP commands and Synchronizing signals provided by the Synchronizer resource type, provide a powerful and effective means of programming complex signals and timing sequences.

- The partitioning of variables into ranges named and defined by the test programmer, and the evaluation of parameters in terms of these ranges.

- Decision tables, their format and usage.

- Dimensional units in arithmetic computations.

OPAL contains general-purpose constructs which can express the most complex digital test problems, including synchronization and shaping of leading or trailing edges. Therefore, unlike ATLAS, there are no specific digital constructs in OPAL. This capability has been achieved with the introduction of the Synchronizer and Signal Conditioner resource types and the expanded control concept of the Source and Sensor resource types. Even interconnections between virtual resource types are definable to aid programming and ATE resource allocation.

The most valuable innovation in OPAL is the capability for binding separately compiled and validated modules before run-time execution without recompilation. The savings in programming time, disk space and compile time are substantial.