Lua in ATE – Evolutions in Cable Assembly and Wire Harness Analysis and Functional Testing

CABLETEST
www.CableTest.com

August 05

Outline of Presentation

CABLETEST
www.CableTest.com

- CableTest Overview
- Lua in ATE:
  - Introduction
  - Data Conversion Wizards
  - A Lua Implementation of a Netlist Database
  - Integration of Lua into Legacy Scripting Language
  - JIT Conversion to Lua
  - Lua Support for Engineering Values
  - Integrating Lua in Other CableTest Products

CableTest Overview

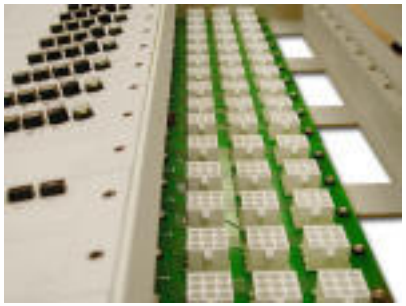
CableTest Systems

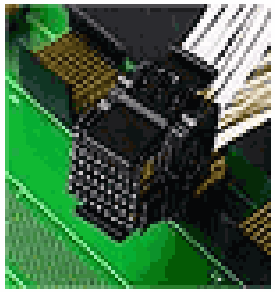
CABLETEST
www.CableTest.com

- Interconnect Test Experts:
  - Continuity (density/speed)
  - High Voltage (range)
  - Measurement (accuracy)
- 1,500 systems installed worldwide and growing
- Multi-lingual technical support
- 25 Employees

Applications

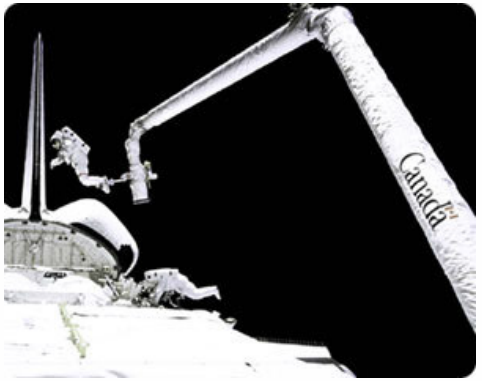
- Backplanes
- Connectors (including SCSI and Filter)
- Functional Test
- Fuse Blocks
- Harnesses
- Power Cords
- Twisted Pair Telecom Wire
- Wiring Systems

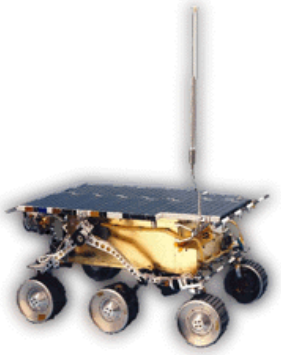




Industrial Sectors Served

- **Aerospace**
- **Mass Transit**
- **Military**
- **Telecom**
- **Other**





MPT Family

CABLETEST
www.CableTest.com

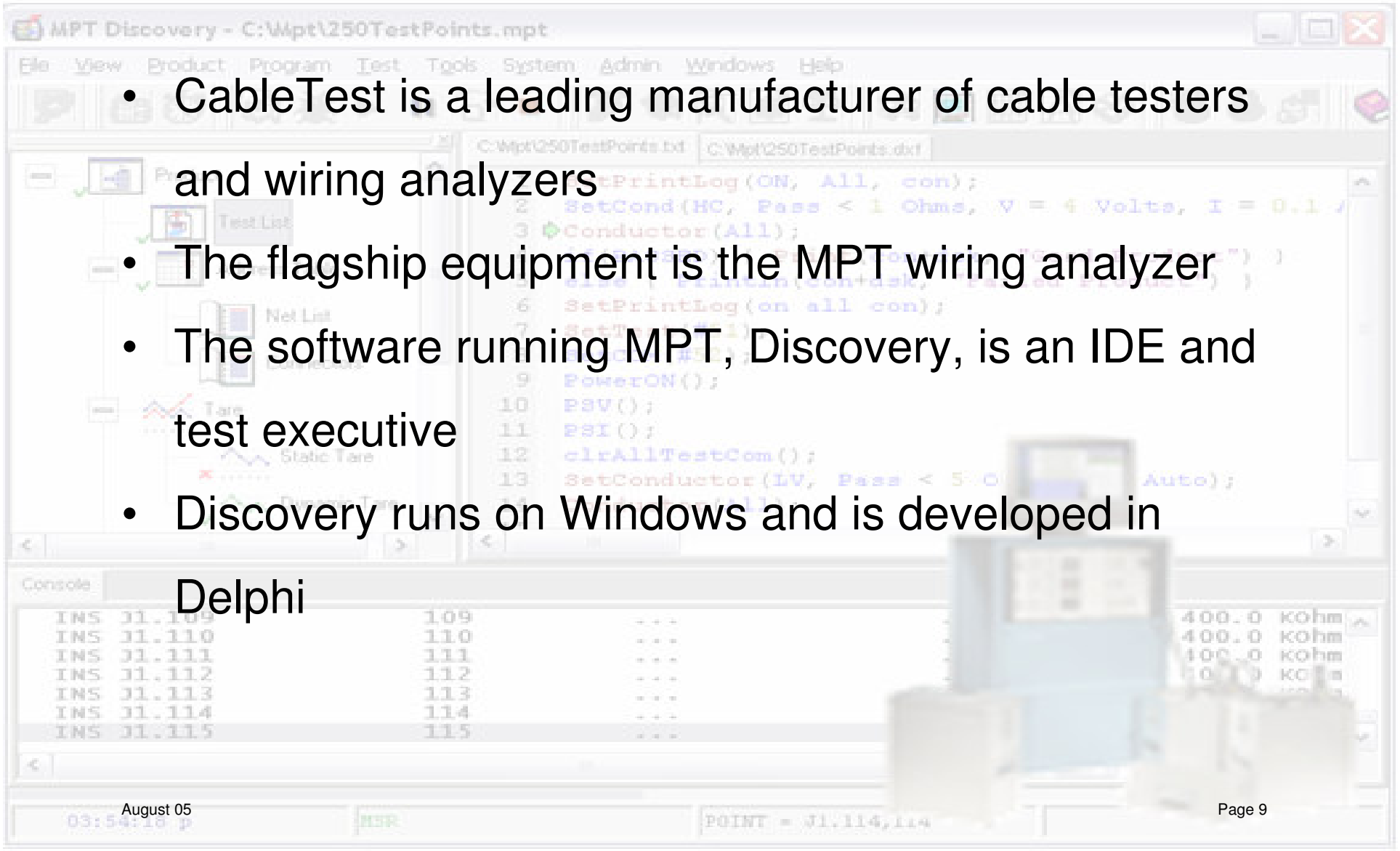
- Mixed high & low voltage energization
- Floating ground measurement
- Mass HiPot capability
- Up to 60,000 test points
- Test capabilities:
  - Programmable or Fixed 5A
  - Low Voltage
  - Up to 20,000 VDC
  - Up to 6,000 VAC

Lua in ATE

1. Introduction

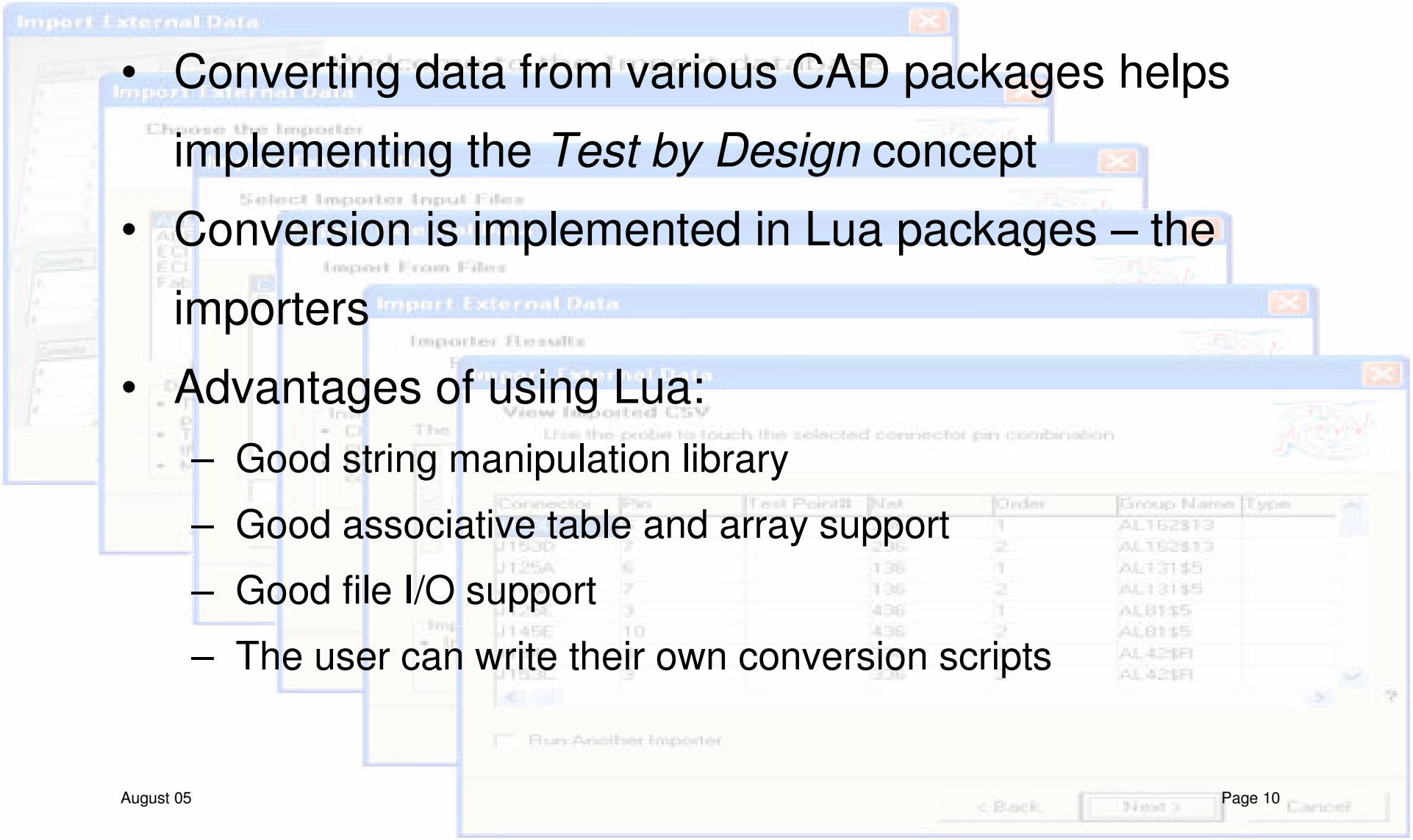
CABLETEST
www.CableTest.com

- CableTest is a leading manufacturer of cable testers and wiring analyzers
- The flagship equipment is the MPT wiring analyzer
- The software running MPT, Discovery, is an IDE and test executive
- Discovery runs on Windows and is developed in Delphi



2. Data Conversion Wizards

CABLETEST
www.CableTest.com



- Converting data from various CAD packages helps implementing the *Test by Design* concept
- Conversion is implemented in Lua packages – the importers
- Advantages of using Lua:
  - Good string manipulation library
  - Good associative table and array support
  - Good file I/O support
  - The user can write their own conversion scripts

Data Conversion Wizards (continued)

CABLETEST
www.CableTest.com

- Unique public interface across the importers helps utilizing them in a conversion Wizard

```
Public.name = 'SomeTestFixture'
Public.version = '1.1'
Public.start_info = [[This module is intended to import a drawing file (.df) ...]]
Public.input_info = [[Select the drawing file(s) (.df) to be used as input(s) ...]]
Public.output_info = [[Select the MPT address table (.csv) file(s) to be used as ...]]
Public.end_info = [[This was the final step for generating a complete MPT address ...]]
Public.input_filter = [[CAD Files(*.df)| *.df |Any Files(*.*)|*.*]]
Public.output_filter = [[CSV Files(*.csv)| *.csv |Any Files(*.*)|*.*]]
function Public.initialize()
  ...
end
function Public.finalize()
  ...
end
function Public.import(in_file_tbl, out_file_tbl, sort_order)
  ...
end
```

3. A Lua Implementation of a Netlist Database

CABLETEST
www.CableTest.com

- Replaced BTrieve database with Lua implementation
- Eliminated lack of flexibility in BTrieve implementation
- Improved application throughput significantly – in some cases from ~30min to ~30s
- Extended the database functionality with support for arbitrarily complex component networks
- Removed the field width limitations
- Implemented data consistency checks directly in the database code

A Lua Implementation of a Netlist Database (continued)

CABLETEST
www.CableTest.com

- Reduced the number of database files to 1 per netlist
- The database preloads with the information sorted in several different ways to optimize the automatic test pattern generation
- Efficient memory consumption due to table references
- Efficient retrieval due in part to table traversal semantics
- Took advantage of Lua's recursive calls to parse component networks of arbitrary complexity
- Used PIL like interface to delimiter separated files

4. Integration of Lua into Legacy Scripting Language

CABLETEST
www.CableTest.com

- Legacy scripting language is a home-brewed, electrical test oriented, C-like language
- Lua code can be embedded with constructs like:

```
SetPrintLog(CON, ON, AllVolt);
Lua(
  function adjust_current(i)
    if i == 0 then
      sethcs{dev = HC3} --Turns off source
      return
    end
    local ballast
    ...
```

- Lua chunks run in a sand box to prevent altering the environment
- The user can write custom event handlers in Lua
- Ability to create custom report formats

- Manual bindings to Delphi functions, variables and constants

```
Type
   PLUAVarXchgRec = ^TLUAVarXchgRec;
   TLuaVarXchgRec = record
      Name: PChar;
      setter: lua_CFunction;
      getter: lua_CFunction;
      case integer of
         0: (realval: real);
         1: (intval: integer);
         2: (byteval: byte);
         3: (boolval: boolean);
         4: (stringval: pchar);
         5: (addr: pointer);
         6: (func: lua_CFunction);
   end;
```

```
// Declare variables, functions and constants for
// registration with Lua
SimpleVariables: array [0..100] of TLuaVarXchgRec =
((Name: 'htmlcheck1'; setter: SetBoolean;
  getter: GetBoolean; addr: @HTMLCheck1),
 (Name: 'programpath'; setter: nil;
  getter: GetFunc; addr: @ProgramPath),
 (Name: 'BUS_MAIN'; setter: nil;
  getter: GetIntConstant; intval: BUS_MAIN),
 ...
// Register variables, functions and constants
RegisterSimple(SimpleVariables);
// Registers table variables
RegisterTableAndFields('stats', StatsStruct);
```

- Getters and setters use pointer to data as upvalues

5. JIT Conversion to Lua

CABLETEST
www.CableTest.com

- Current embedding scheme goes to depth 1 only
- Current syntax highlighter only handles one language at a time
- User has to switch between two different syntaxes

- We would like to migrate to doing a JIT conversion of the legacy scripting language to Lua (in order to preserve some valuable syntactic sugar)
- We experimented so far with Gema

6. Lua Support for Engineering Values

CABLETEST
www.CableTest.com

- One of the beloved features of the legacy scripting language is the user's ability to enter engineering values in natural format:

```
SetConductor(LV, Pass < 5.5 Ohm, I = Auto);
Continuity(All);
SetResistance(5V, Pass = 1.20 KOhm, 1.60 KOhm, I = Auto);
Resistor(P1.2, P1.3);                          //Test coil resistance
...
```

- Lua would support engineering values (magnitude/dimension/ unit/precision) through tables or userdata, however the syntax is complex
- Would like to tackle this in JIT preprocessing stage.

7. Integrating Lua in Other CableTest Products

CABLETEST
www.CableTest.com

- Horizon 1500 – an embedded system – is used as a stand-alone cable tester
- Current scripting capabilities are addressed with Tcl
- While powerful, Tcl is hard to grasp by our customer base mainly due to its peculiar syntax
- Would like to either replace Tcl scripting with Lua scripting or have them integrated side by side