TTCN-3 Code Generation

TTCN-3 User Conference Asia 2009

- Venkata Ramana Gollamudi
Huawei Technologies India Pvt Ltd.

Agenda

1. Motivation
2. Modular Test Suite Design
   - Advantages of Modular Test Suite Design
   - Modular Test Suite Design using TTCN3
3. TTCN3 Test Suite Code Generation
   - Overview
   - Standardizing the “Modular test suite design” for Code Generation
   - Sample TTCN3 Framework Code generation Wizard.
4. Conclusion

Motivation

- ***Reusability of test code*** is important for improving efficiency and maintainability. But people fail to inculcate due to **lack of a standard framework for reusable components/libraries**.

- Reusable code developed might not be eventually used, due to its **poor extendibility** and no extension points defined.

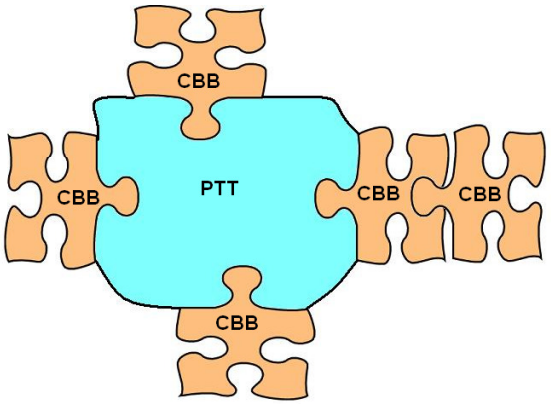
- Require a streamline mode that will help test teams to **select the reusable components and customize them with ease.**

- Difficult to maintain **common design** for **similar test suites** across teams.

Abstract

There has been a strong urge to address these challenges with a new approach of **automatic generation of TTCN3 Test Suite Framework** by selecting and customizing:

- PTT - Predefined Test Suite Templates
- CBB - Common Building Blocks



This is an **attempt to standardize reusable components**; there by improving **reusability** to achieve test code efficiency.

Terminology

- **Predefined Test Suite Templates (PTT)** are built from different categories of test suites, test topologies or application of TTCN3. They contain the **test suite skeleton with sample test cases**.

  Eg. Test Topologies- Test suite simulating different network elements, Multiple instances of same network element.

  Application Area- Protocol Conformance testing, Web app testing and so on.

- **Common Building Blocks (CBB)** are built based on the **common modules and functionality** used in different test suites.

  Eg. Synchronization module, Log module, Verdict handling, Profiling, test data import and so on.

PTTs and CBBs shall follow Modular Design approach

Modular Test Suite Design

Modular Design is an approach to subdivide a system into smaller parts (modules) that can be independently created and then used in different systems to drive multiple functionalities.

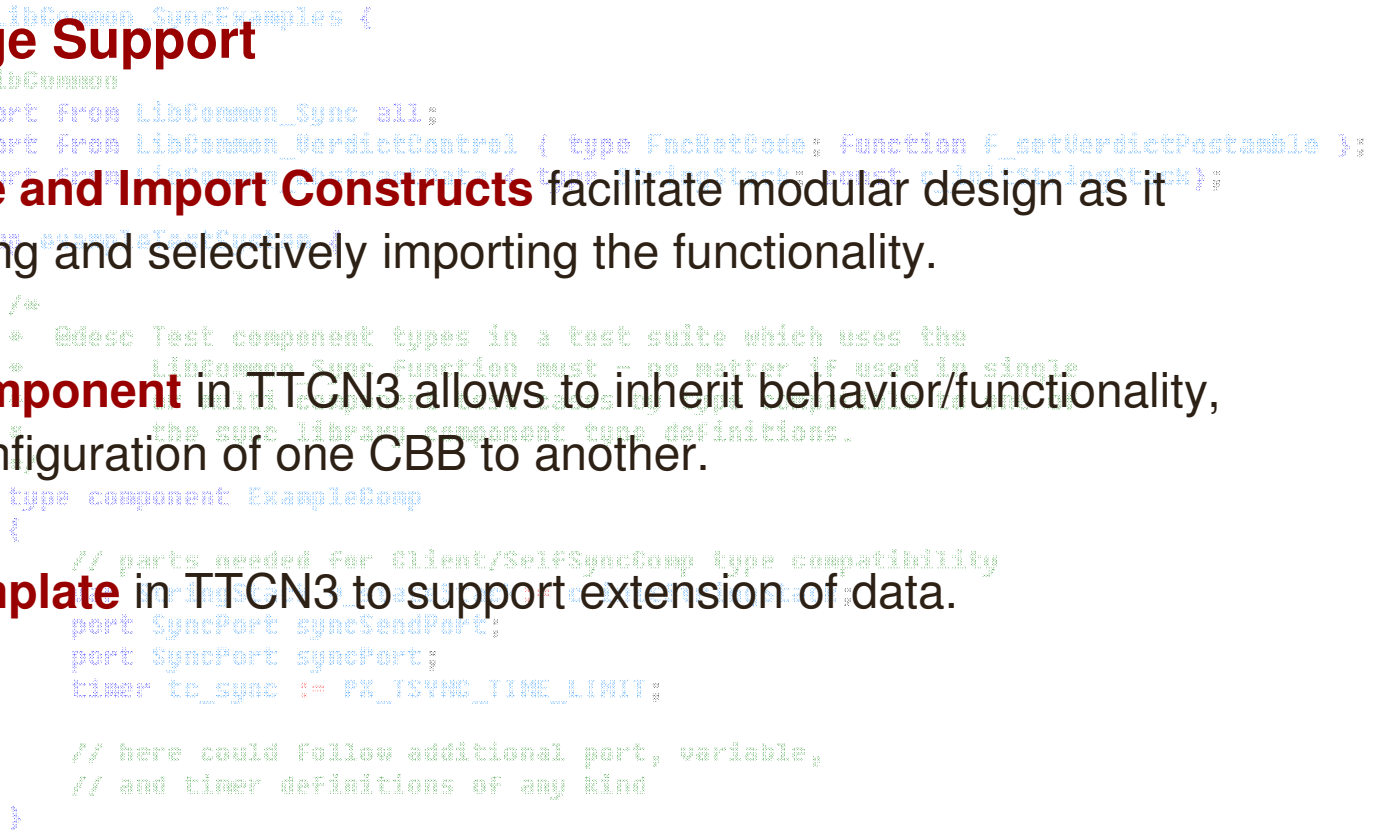
Advantages

- **Reduction in effort**
  - due to lesser customization, and less learning time
- **Flexibility in design**
- **Augmentation and Exclusion**
  - adding new solution by merely plugging in a new module and excluding it

Modular Test Suite Design Using TTCN3

TTCN3 Language Support

- **TTCN3 Module and Import Constructs** facilitate modular design as it allows separating and selectively importing the functionality.
- **Extending Component** in TTCN3 allows to inherit behavior/functionality, content and configuration of one CBB to another.
- **Modifying Template** in TTCN3 to support extension of data.



Modular Test Suite Design Using TTCN3

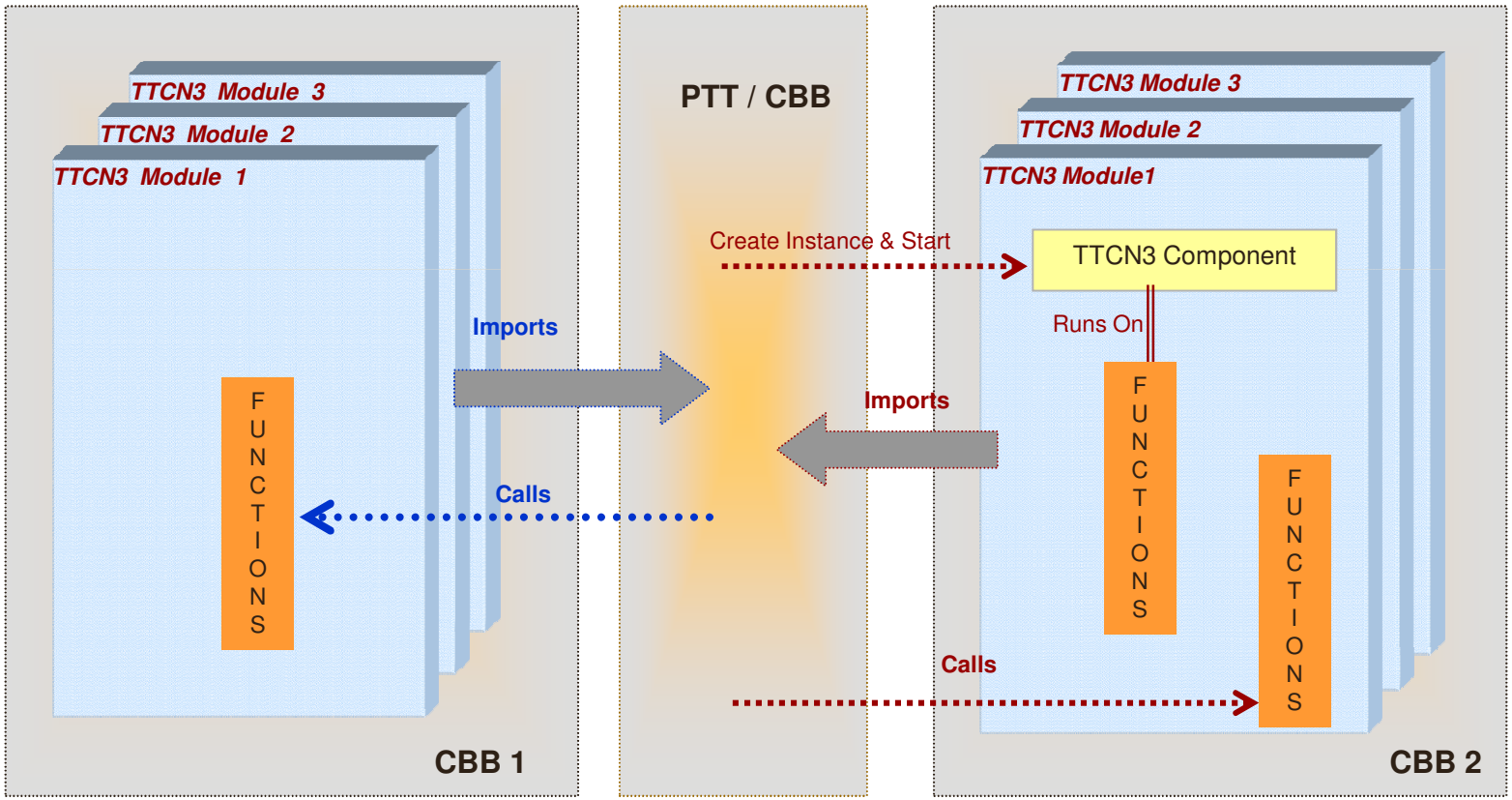
Implementing PTT and CBB Using TTCN3 by segregating:

1. Independent functionality into CBB
2. TTCN3 component (s) into CBB
3. Some TTCN3 component content and corresponding functionality into CBB

Modular Test Suite Design Using TTCN3

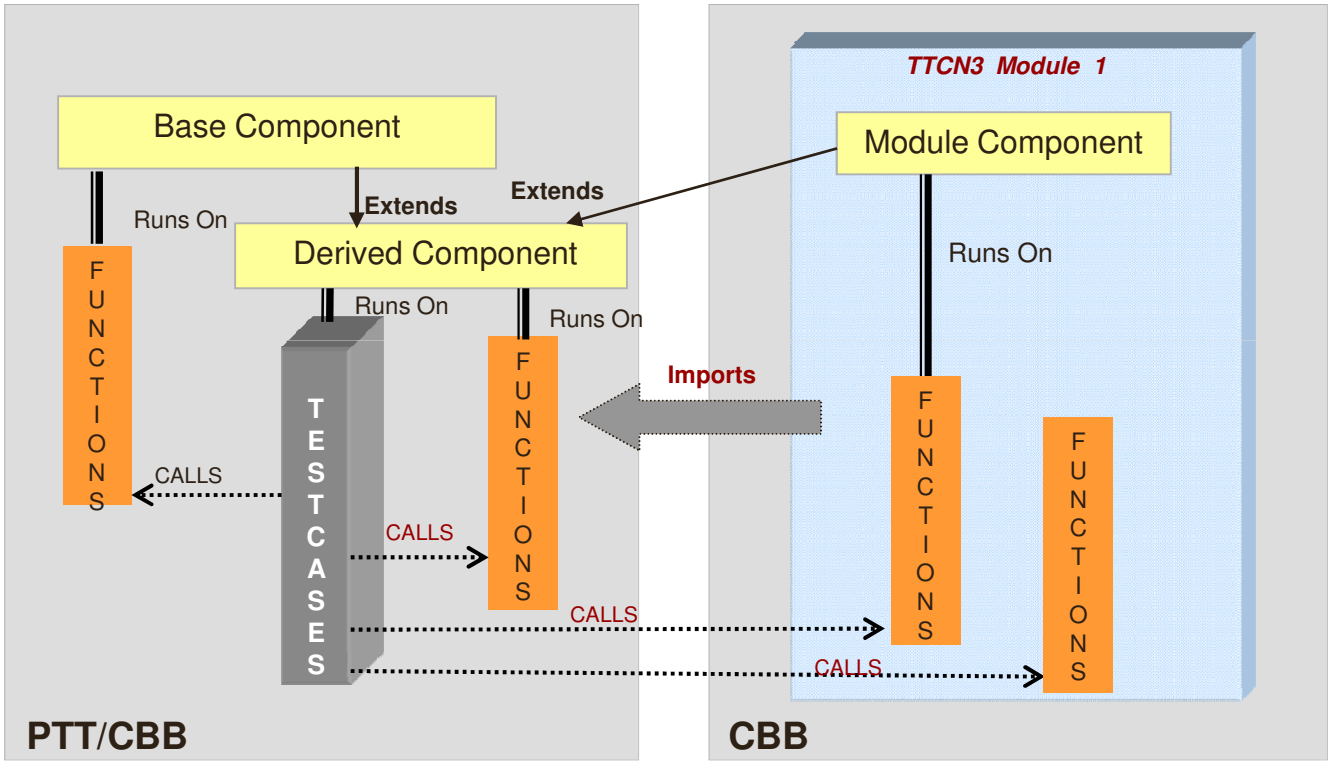
1. Segregating **Independent functionality into CBB**

2. Segregating TTCN3 **component into CBB**



Modular Test Suite Design Using TTCN3

3. Segregating some **TTCN3 component** content and corresponding functionality into **CBB**



CBB will be used in test suite (PTT/another CBB) by **extending the component (s)** to get all the required functionality and component content imported.

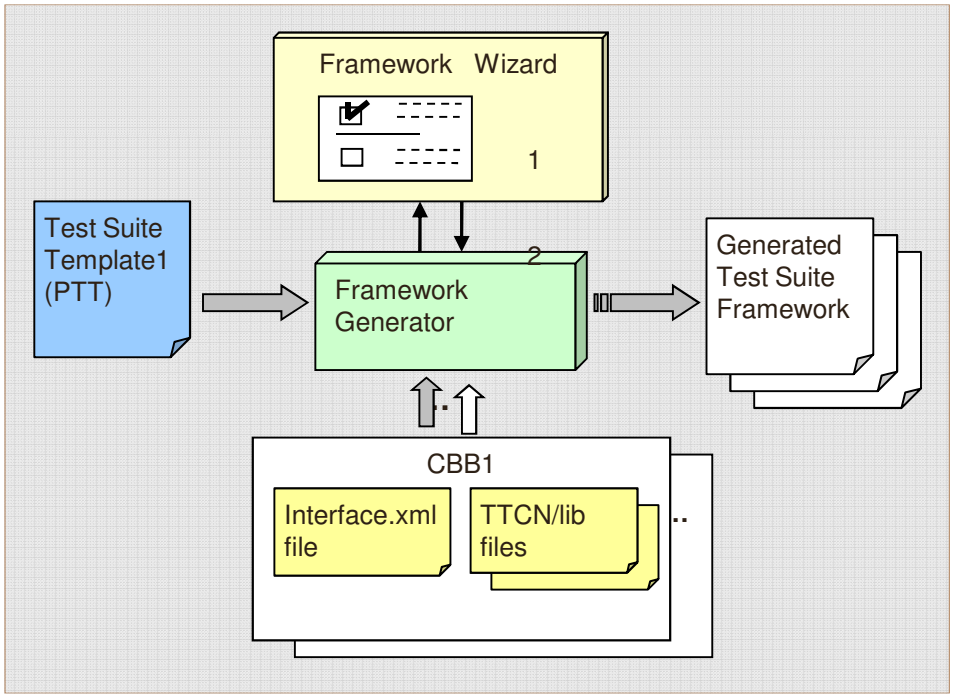
TTCN Framework Code Generation

Auto Test Framework Wizard

Assists users in quicker test suite development. Generates the test suite framework with modules integrated and sample test cases as per user selection and configuration.

Options :

- Selecting Predefined Test Suite Template (PTT) and configuring parameters.
- Selecting essential CBBs and configuring the CBB parameters.
- Generating the required test suite framework code by importing the selected CBBs.
- Adding user-defined CBBs / PTTs.



Standardizing the “Modular Test Suite Design” for Code Generation

The **CBB & PTT format** must be **standardized**, so that a generic code generation wizard can use the CBB(s) and PTT during its code generation.

Sample Structure of CBB is as follows (PTT structure in similar lines):

- **Interface File** (say Interface.xml) with details such as:
  - CBBId
  - Description
  - DependsOn CBBs
  - ConflictsWith CBBs
  - Is MultipleInstances Allowed etc

Standardizing the "Modular Test Suite Design" for Code Generation

- **Interface File (Contd):**
  - Configuration Parameters - through GUI during generation and are replaced during code generation, also can guide the logic of generation.

    Example: *%CBBId%ParamId%*

  - Interface Snippets - contains TTCN code to call the interfaces exposed by the module. One Snippet can also override other CBB Snippets.
  - Hooks Exposed - from current CBB, so that other CBBs can implement those hooks for Code Injection.
  - Hooks Implemented - in current CBB, which are exposed from other CBBs.
  - Target file structure - specifying files to be generated from input files.

Standardizing the “Modular Test Suite Design” for Code Generation

- **TTCN Code Containing Generation Tags**:

  These generation tags are **replaced during generation** with appropriate parameters or interface calls.

  Generation tags -

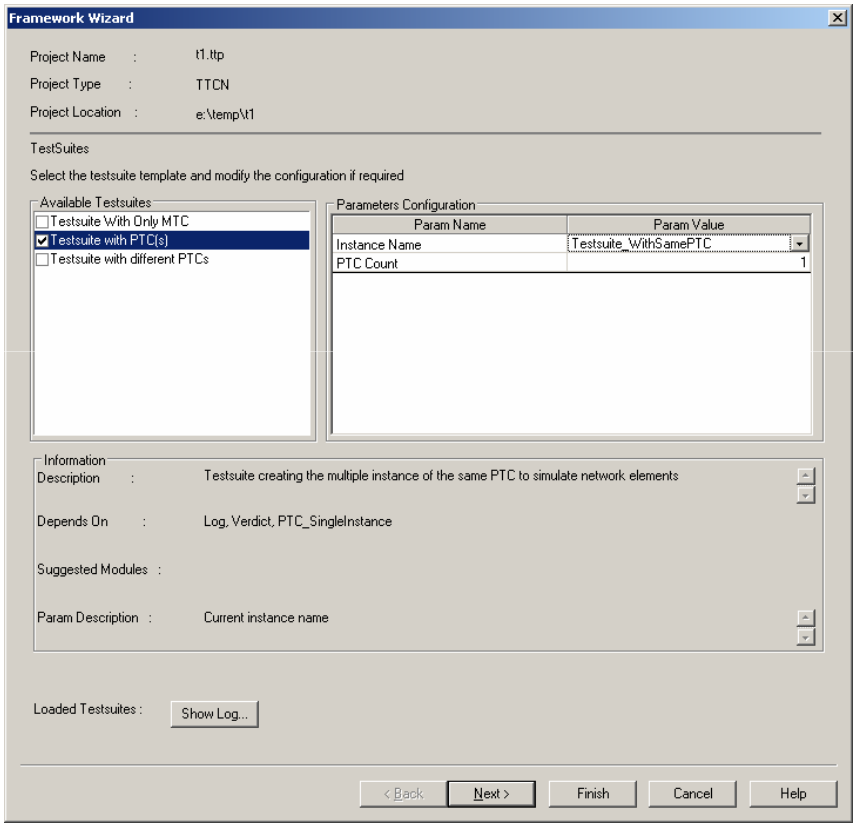
  - Used to specify the logic of generation:
    <foreach>,<if>, <else>, <ifexists>
  - Work on selected CBBs and Parameters configured
  - Can be used inside interface file, TTCN3 code files
  - Can be nested

Example:

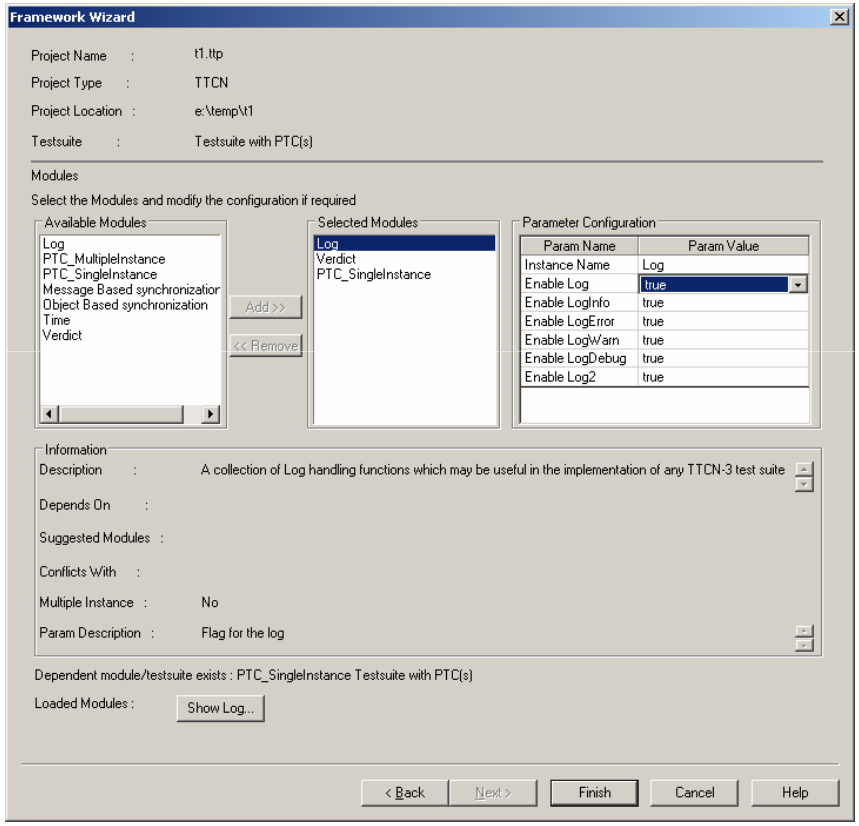
```
<foreach value = "%%ICount%">,
<if value = "%Msg_Sync%pPreCondSync%" equals="Yes">,
<else>,
<ifexists cbb=“Msg_Sync">
```

Sample Code Generation Wizard

PTT Selection Page



CBBs Selection Page



Conclusion

- This method is more than a library as it **can generate the complete integrated code with sample test cases**.
- The collection of CBB(s) & PTT(s) over the time makes a **rich tool kit** for easy test suite framework generation.
- Method is language independent, so single generation wizard **can work across languages**.
- Can be extended to **configure the coding guidelines** of TTCN3, by implementing Coding guide lines as a CBB and configuring its parameters during generation.
- This can also be **merged with generation of templates** by giving the required data structure (say ASN) as input during generation, making it more complete.
- Can be further extended to make a **Code generation language**.

Thank You