Testing Web Services with TTworkbench

An Introduction to TTCN-3 and its Application for WSDL/SOAP

www.testingtech.com.cn Email:sales@testingtech.com.cn Tel:010-56497908

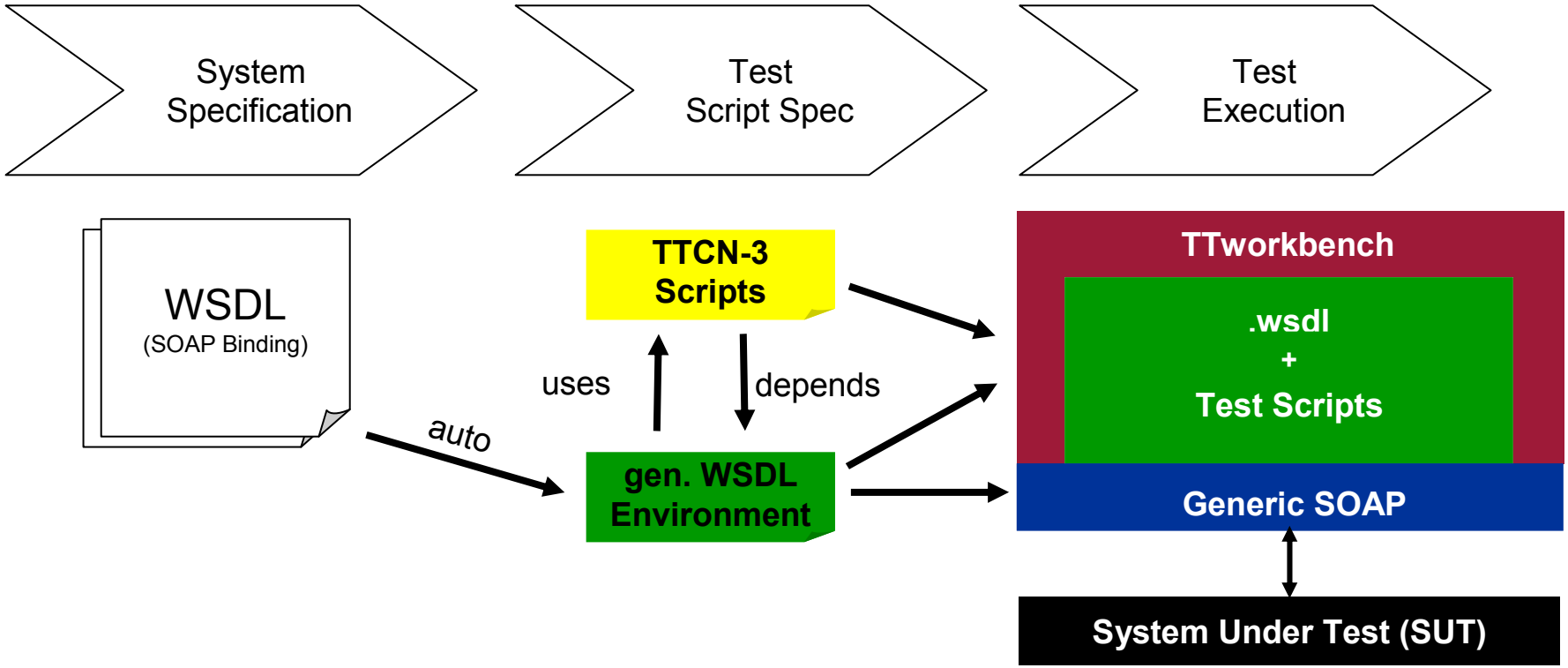
Current Problems

- Systems
  - Increase in complexity, concurrency, and dynamics
  - Heterogeneous technologies, platforms and tools
  - Rapid change/extension of features and capabilities
  - Security threads and vulnerabilities
- Processes
  - Insufficient common terminology between architects, developers, testers, managers
  - Inconsistent documents on requirements, specifications, documentation
  - Lack of test integration
  - Low degree of automation
  - Limited support of cooperative development

Motivation

- Systems
  - Increase in complexity, concurrency, and dynamics
  - **Heterogeneous technologies, platforms and tools**
  - **Rapid change/extension of features and capabilities**
  - Security threads and vulnerabilities
- Processes
  - Insufficient common terminology between architects, developers, testers, managers
  - Inconsistent documents on requirements, specifications, documentation
  - **Lack of test integration**
  - **Low degree of automation**
  - Limited support of cooperative development

Workflow



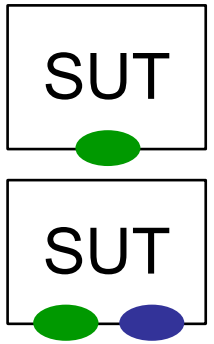
- System specification phase produces WSDL specifications
- Test script specification phase produces test scripts based on generated TTCN-3 WSDL environment (TTCN-3 types and runtime components)
- Test execution phase executes WSDL-based test scripts using SOAP binding against web service

TTplugin-WS

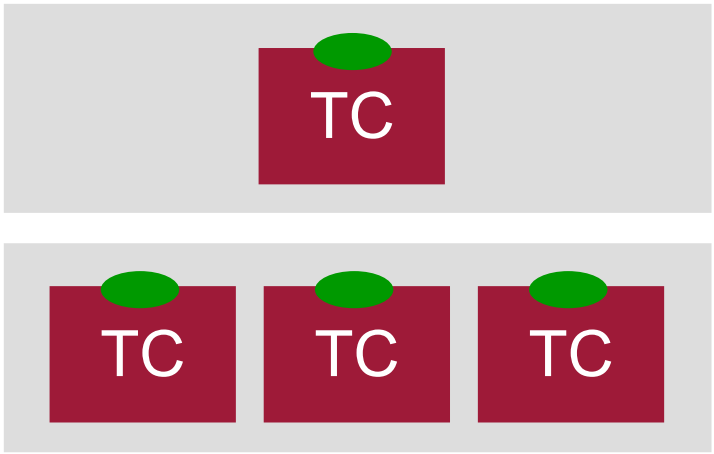
- WSDL/SOAP highlights
  - Testing of invalid parameter values, i.e. out-of-bounds parameters
  - Access different web services at the same time, from same components or different components
- TTCN-3 highlights
  - Automatic import of WSDL specifications into TTCN-3
  - Multiple test components and multiple-port mapping
  - Freely combinable with additional test access provided by TTplugins
- Specifications supported
  - WSDL 1.1 / SOAP 1.1
  - TTCN-3 v3.2.1 and above

Different Test Scenarios and Requirements

- SUT driven
  1. Only one web service exposed
  2. Multiple web services exposed

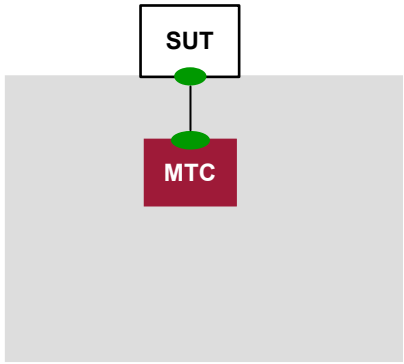


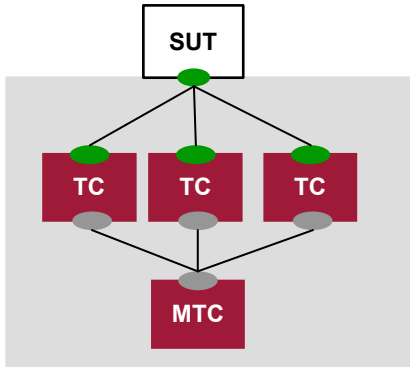
- Test aspects
  3. Single client
  4. Multiple concurrent clients



Summary of Test Scenarios

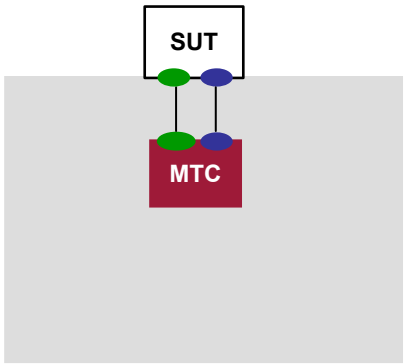
One client
One service
(1-3)

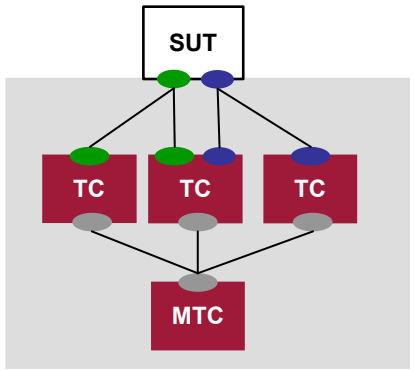




Multi client
One service
(2-3)

One client
Two services
(1-4)



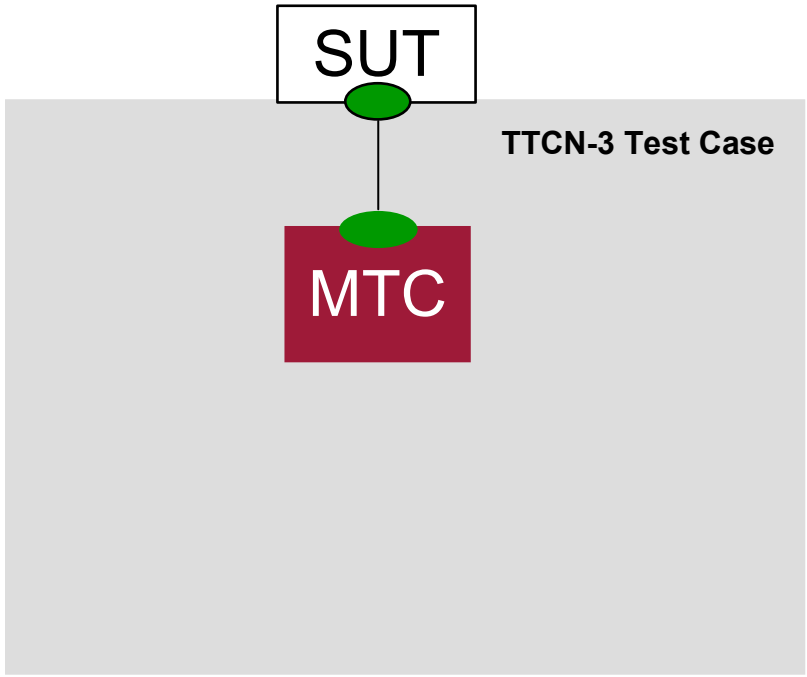


Multi client
Multi services
(2-4)

Dynamics of Scenario 1-3

Testing Technologies

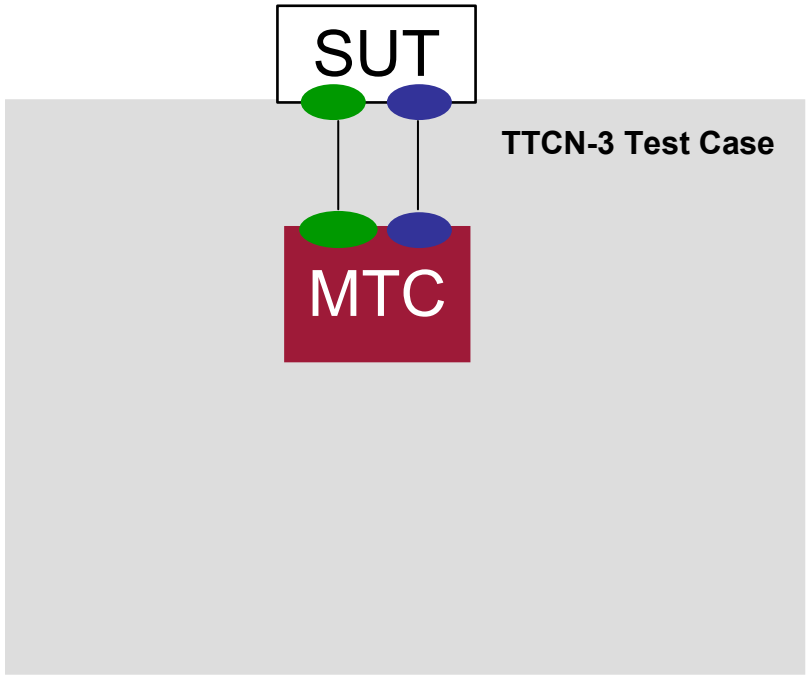
Single Web Service – Single Client



- One client (represented through the MTC) tests one service
- Mainly used for functional testing

Dynamics of Scenario 2-3

Multiple Web Services – Single Client

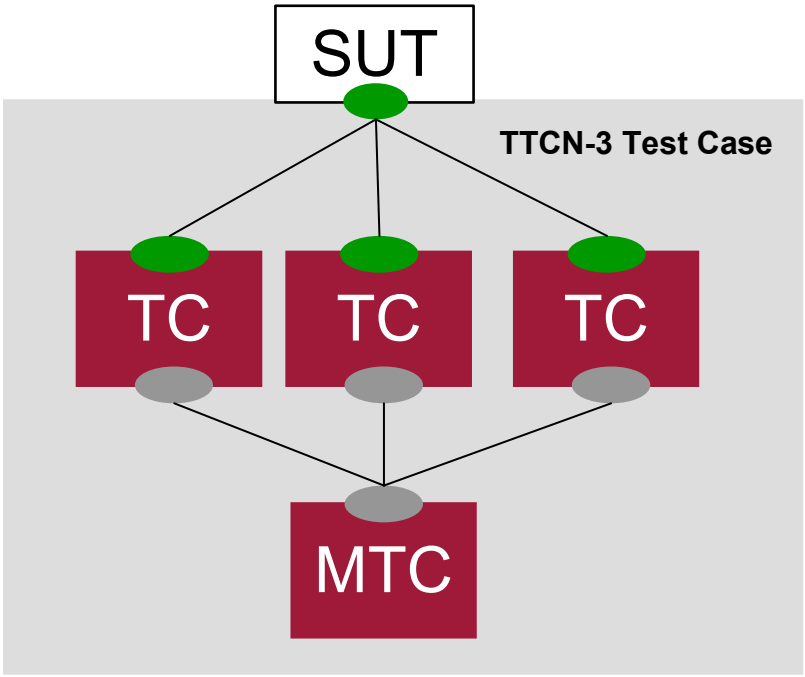


- One client uses two interfaces to communicate with the SUT
- Mainly used for integration testing

Dynamics of Scenario 1-4

Testing Technologies

Single Web Service – Multiple Clients

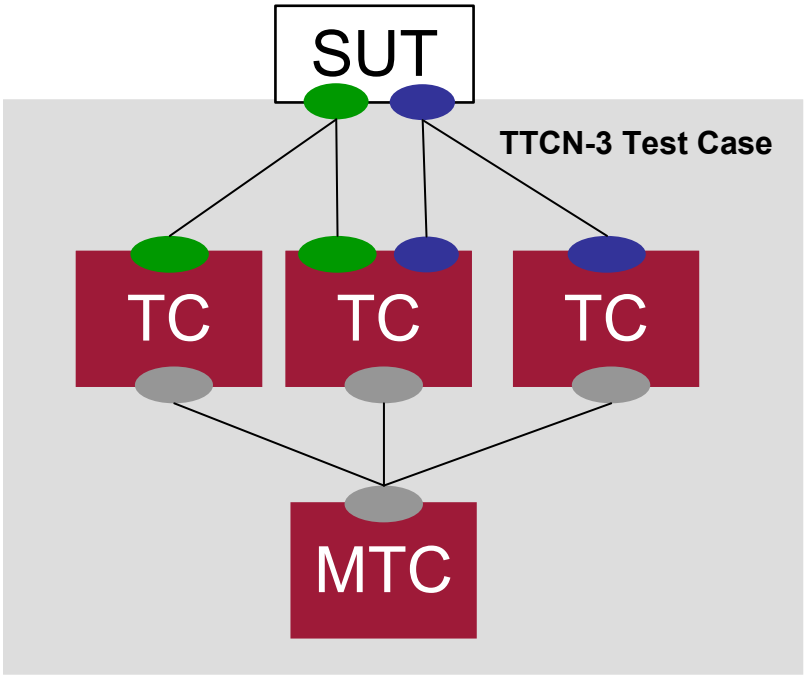


- MTC controls TCs, which execute the test behavior
- Test behavior could always be the same or different behavior
- Mainly used for functional, scalability or load testing

Dynamics of Scenario 2-4

Testing Technologies

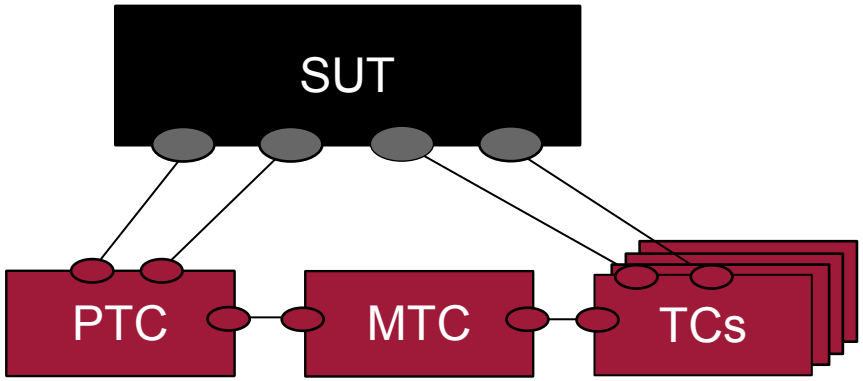
Multiple Web Service – Multiple Client



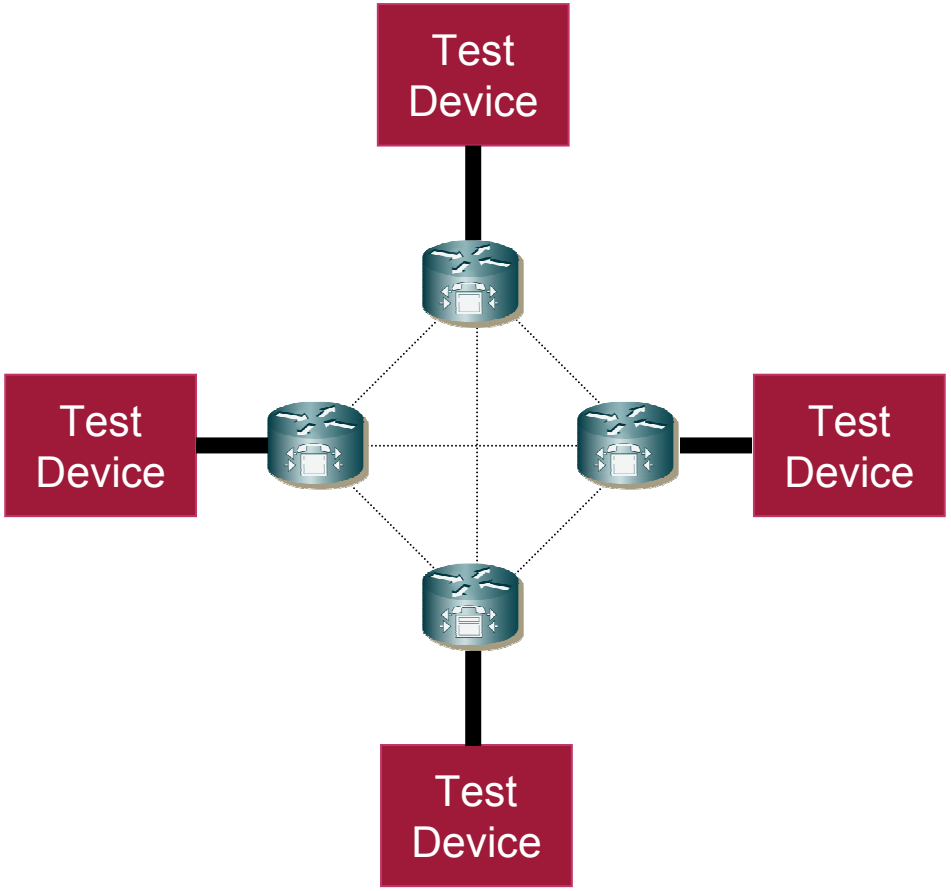
- MTC controls TCs, which execute the test behavior
- Test behavior tests one or more web services
- Mainly used for functional, scalability or load testing

Multiple Test Components Running on Multiple Test Devices

Logical dynamic configuration with TTCN-3

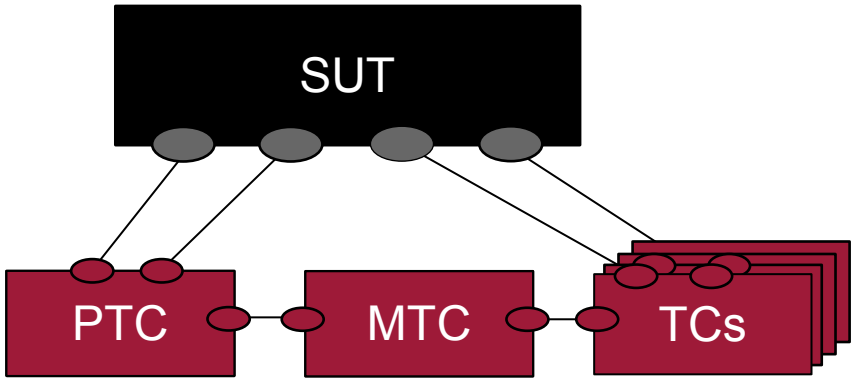


Physical configuration with real test devices

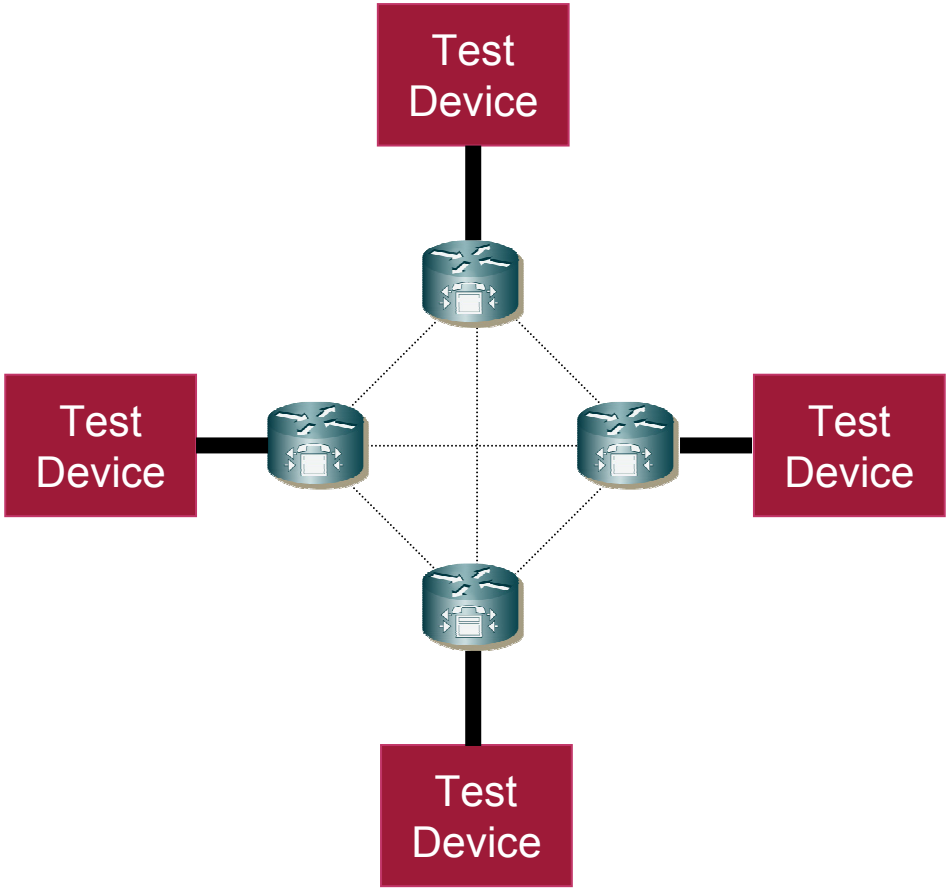


Multiple Test Components Running on Multiple Test Devices

Logical dynamic configuration with TTCN-3

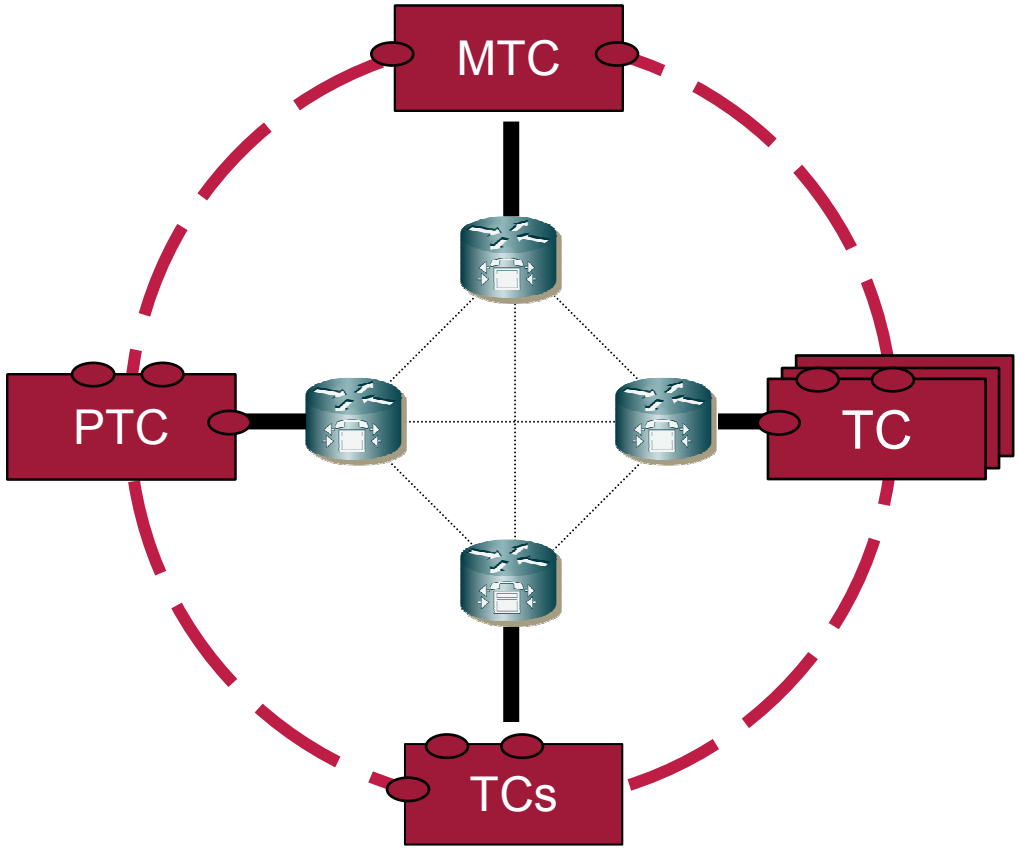


Physical configuration with real test devices



Multiple Test Components Running on Multiple Test Devices

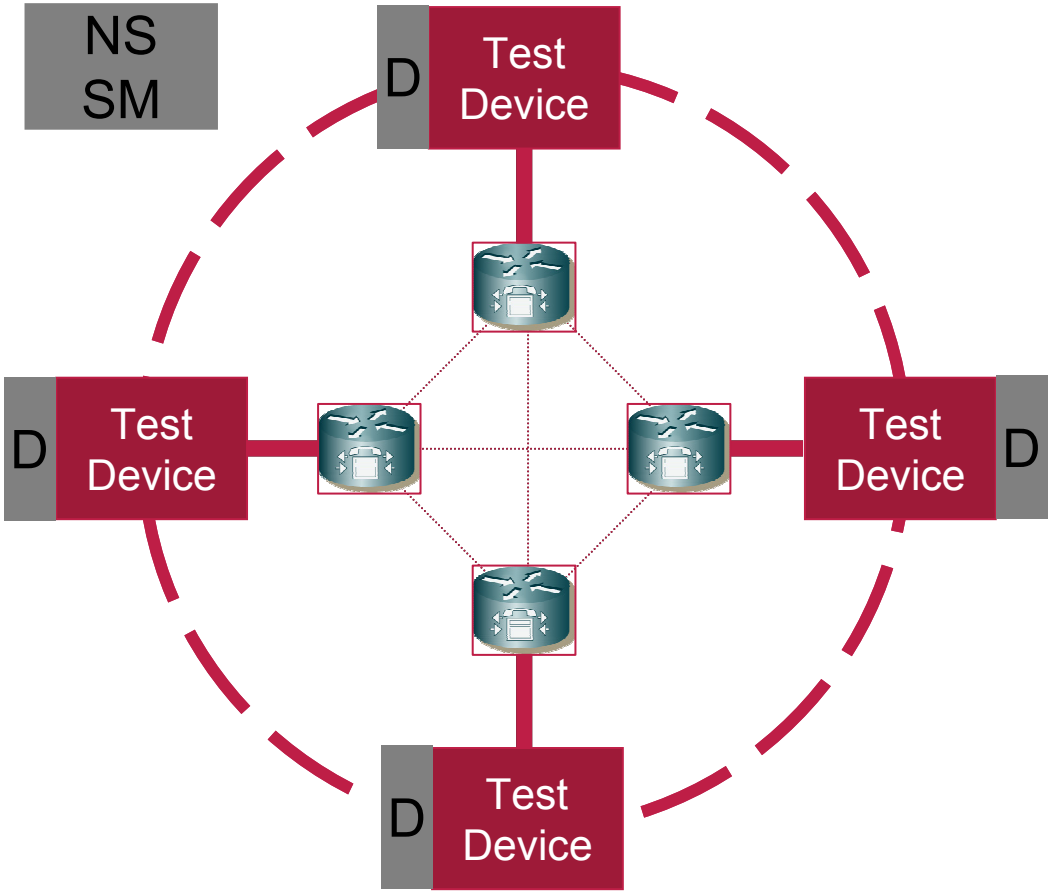
Physical configuration
with real test devices



Ressources Involved

- TTmex Infrastructure
  - NS: Name Service
  - SM: Session Manager
- Per Test Device
  - D: Daemon
  - C: Container
- Per Test Session F1
  - Fn: Test Suite and Libraries for device n

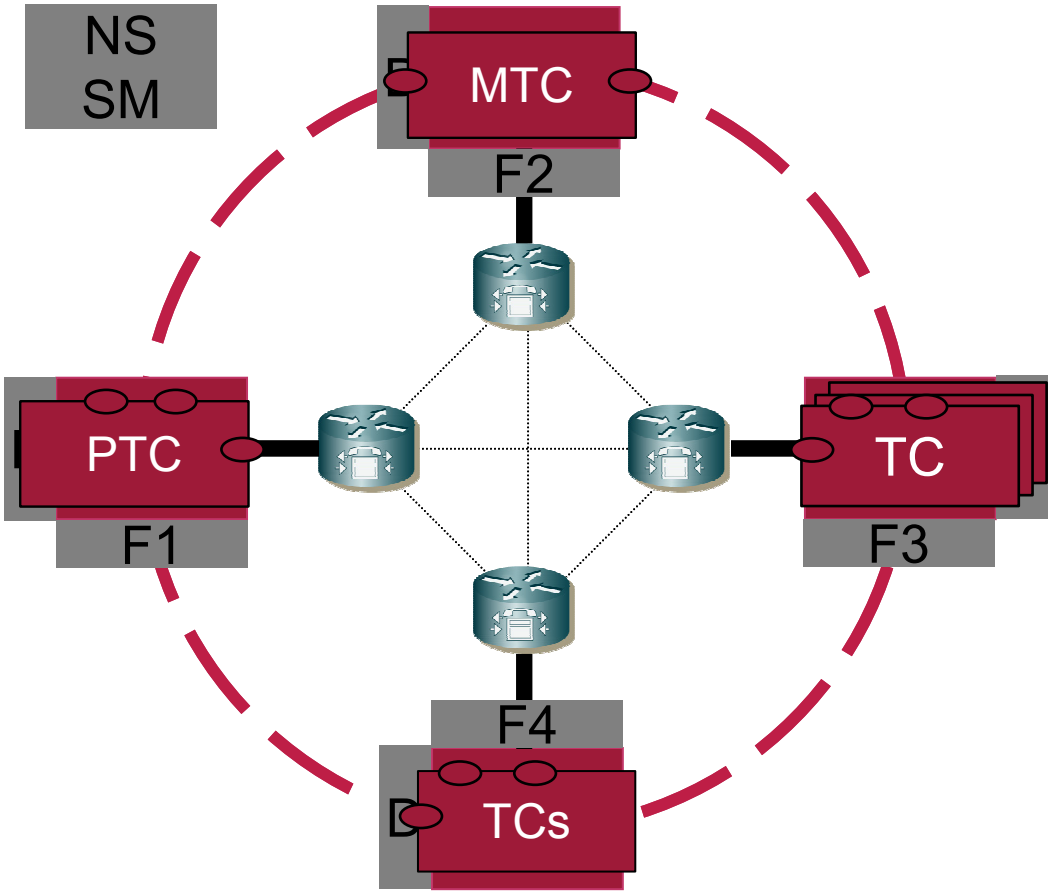
Physical configuration with real test devices



Ressources Involved

- TTmex Infrastructure
  - NS: Name Service
  - SM: Session Manager
- Per Test Device
  - D: Daemon
  - C: Container
- Per Test Session
  - Fn: Test Suite and Libraries for device n
- Run the session

Physical configuration with real test devices



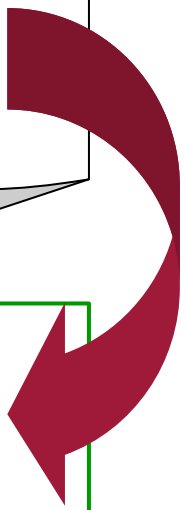
Small Example

- GoogleSearch API
  - Provide a search string with some parameter
  - Evaluate the returned results
- Use the google WSDL as provided by Google
- Check for
  - Availability
  - Content of the result
  - Responsiveness

A Small Example – WSDL2TTCN3

```
<xsd:complexType name="GoogleSearchResult">
    <xsd:all>
      <xsd:element name="documentFiltering" type="xsd:boolean"/>
      <xsd:element name="searchComments" type="xsd:string"/>
      <xsd:element name="estimatedTotalResultsCount" type="xsd:int"/>
      <xsd:element name="estimateIsExact" type="xsd:boolean"/>
      <xsd:element name="resultElements" type="typens:ResultElementArray"/>
      <xsd:element name="searchQuery" type="xsd:string"/>
      <xsd:element name="startIndex" type="xsd:int"/>
      <xsd:element name="endIndex" type="xsd:int"/>
      <xsd:element name="searchTips" type="xsd:string"/>
      <xsd:element name="directoryCategories" type="typens:DirectoryCategoryArray"/>
      <xsd:element name="searchTime" type="xsd:double"/>
    </xsd:all>
</xsd:complexType>
```

GoogleSearch.wsdl



```
type record GoogleSearchResult {
   boolean documentFiltering,
   charstring searchComments,
   integer estimatedTotalResultsCount,
   boolean estimateIsExact,
   record of ResultElement resultElements,
   charstring searchQuery,
   integer startIndex,
   integer endIndex,
   charstring searchTips,
   record of DirectoryCategory directoryCategories,
   float searchTime
 }
```

GoogleSearchTypes.ttcn3

A Small Example – WSDL2TTCN3

```
…
 <message name="doGoogleSearch">
    <part name="key" type="xsd:string"/>
    <part name="q" type="xsd:string"/>
    <part name="start" type="xsd:int"/>
    <part name="maxResults" type="xsd:int"/>
    <part name="filter" type="xsd:boolean"/>
    <part name="restrict" type="xsd:string"/>
    <part name="safeSearch" type="xsd:boolean"/>
    <part name="lr" type="xsd:string"/>
    <part name="ie" type="xsd:string"/>
    <part name="oe" type="xsd:string"/>
 </message>

 <message name="doGoogleSearchResponse">
   <part name="return" type="typens:GoogleSearchResult"/>
 </message>

 <portType name="GoogleSearchPort">

….

    <operation name="doGoogleSearch">
      <input message="typens:doGoogleSearch"/>
      <output message="typens:doGoogleSearchResponse"/>
    </operation>

 </portType>
```

GoogleSearch.wsdl

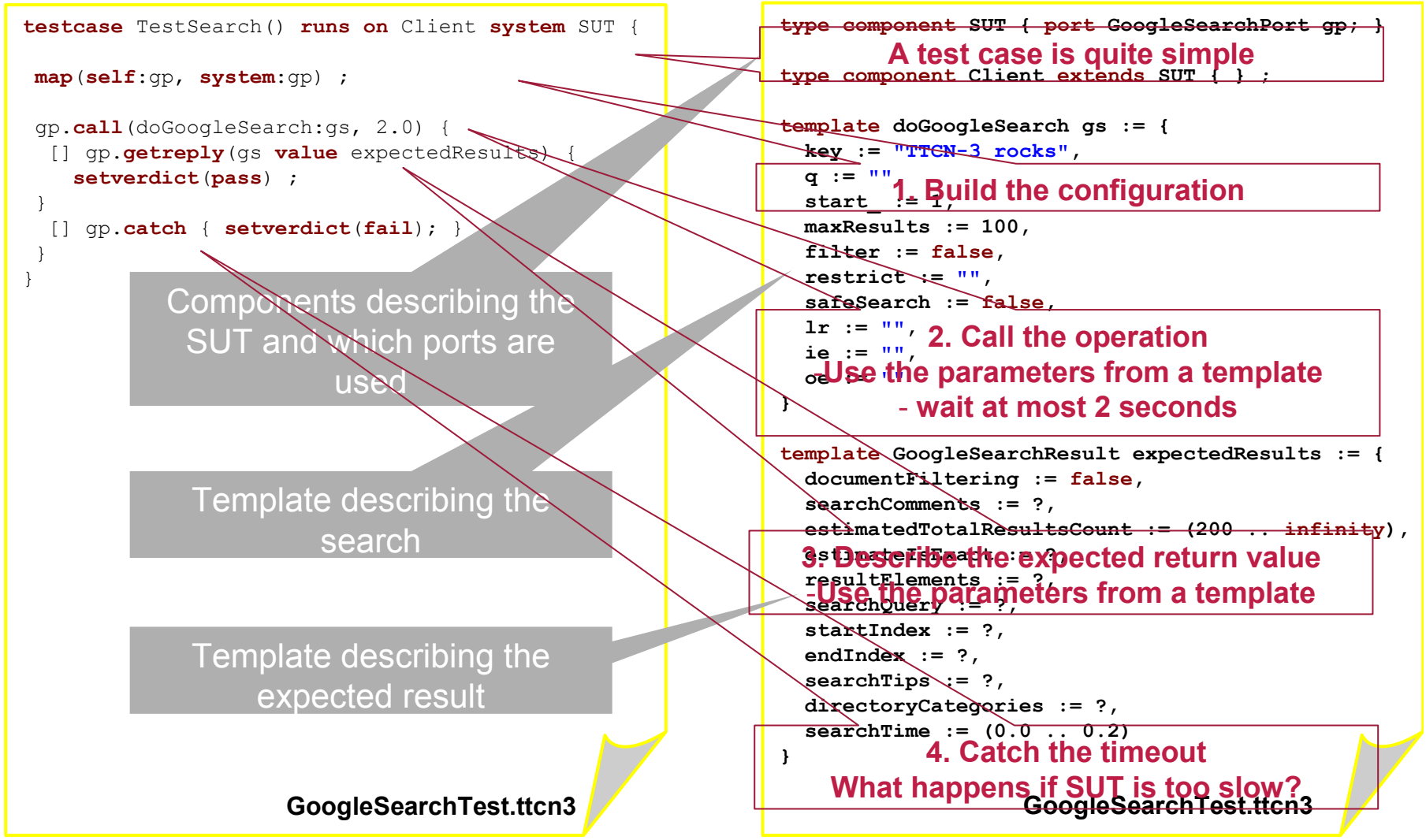
```
signature doGoogleSearch (
              in charstring key,
              in charstring q,
              in integer start_,
              in integer maxResults,
              in boolean filter,
              in charstring restrict,
              in boolean safeSearch,
              in charstring lr,
              in charstring ie,
              in charstring oe
)
   return GoogleSearchResult;
```

GoogleSearchSignatures.ttcn3

```
type port GoogleSearchPort procedure {
   inout doGetCachedPage; // REQUEST_RESPONSE,1
   inout doSpellingSuggestion; // REQUEST_RESPONSE
   inout doGoogleSearch; // REQUEST_RESPONSE,1
 }
```

GoogleSearchPorts.ttcn3

A Small Example – WSDL Test Script



Conclusion

- TTsuite-WSDL/SOAP offers zero-coding features for test execution
- Plug-in concept enables free combination of WSDL/SOAP support in different contexts
- Leverages TTCN-3 towards API testing support
- Problems
  - Heterogeneous technologies, platforms and tools
  - Rapid change/extension of features and capabilities
  - Lack of test integration
  - Low degree of automation

Testing Technologies

Questions?

Thank You!