

# A keyword-driven service testing framework based on TTCN-3

Wen Yongxin  
Huang Shifu

Testing Technology Research Dept, A&S

[wenyongxin@huawei.com](mailto:wenyongxin@huawei.com)

[www.huawei.com](http://www.huawei.com)

HUAWEI TECHNOLOGIES CO., LTD.

Huawei Confidential



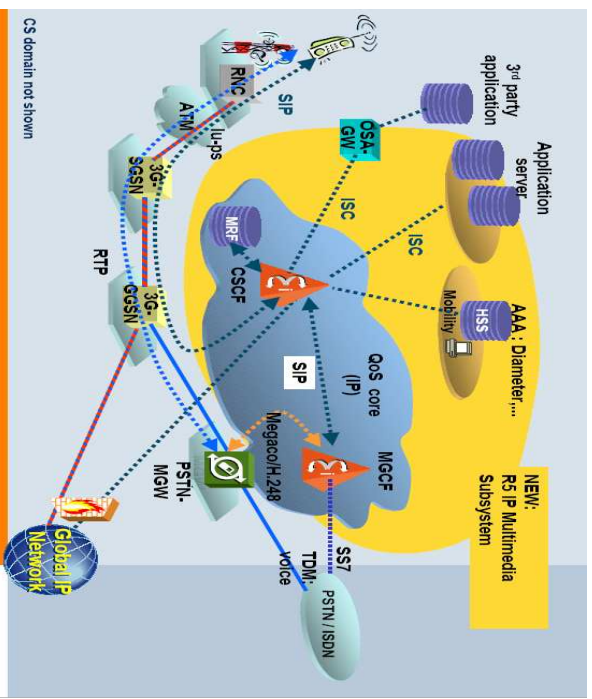
## Contents

- ◆ **Service testing**
- ◆ **TTCN-3 & AW solution**
- ◆ **Typical application**
- ◆ **Benefits**
- ◆ **Future work**



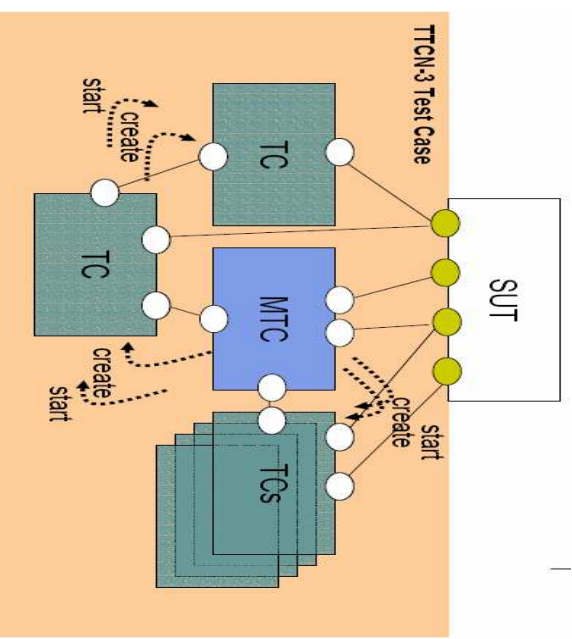
# Service testing – characteristics

- ◆ Many protocols are involved
- ◆ Testers care little about the protocol details



# Service testing – Why TTCN-3

- ◆ Special language designed for testing
- ◆ Especially on protocol conformance testing
- ◆ Simulation of network elements by the parallel components



# Service testing – Difficulties

- ◆ Require more programming skill
- ◆ High cost of script-based test case maintenances
- ◆ Without a test API specification, It is easy to bring too many APIs, which is difficult to use .

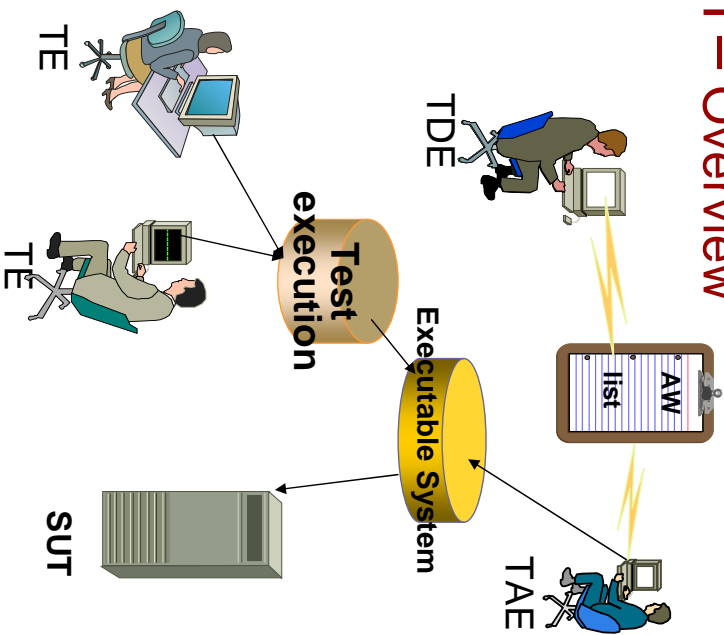
# Contents

- ◆ Service testing
- ◆ **TTCN-3 & AW solution**
  - Overview
  - Framework
  - Presentation layer
  - Script generator
  - Execution layer
  - Application area
- ◆ Typical application
- ◆ Benefits
- ◆ Future work

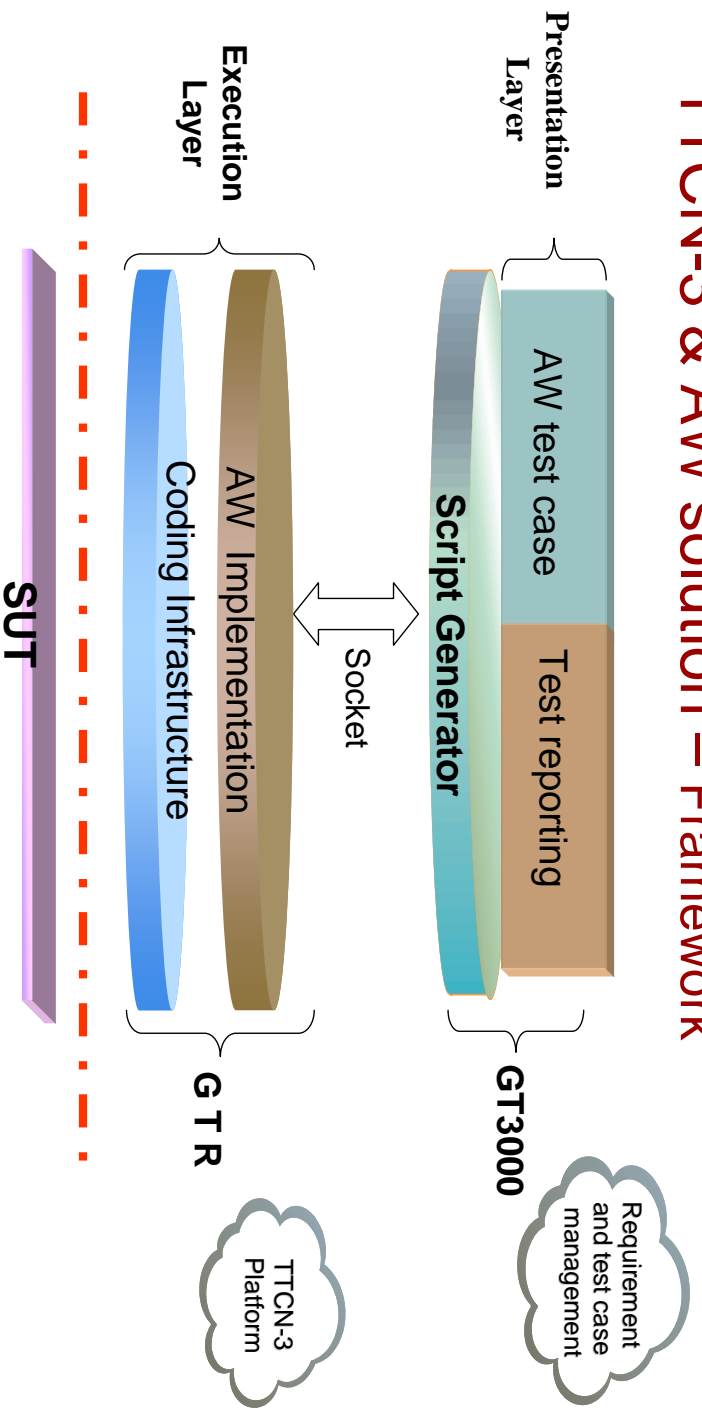


# TTCN-3 & AW solution – Overview

- ◆ Action word (AW ) is a methodology of keyword-driven testing from Huawei
- ◆ A 3rd generation of automatic testing
- ◆ Separates test design from test execution
- ◆ Graphical format of test case.
- ◆ Easy to create and maintain



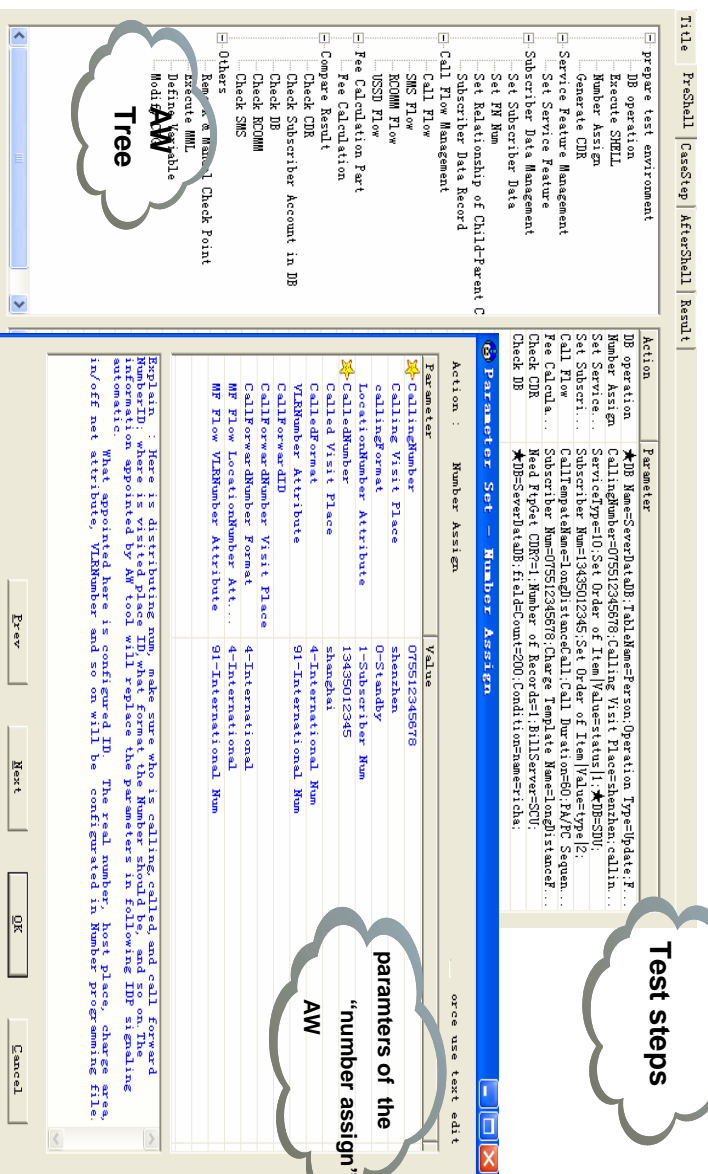
# TTCN-3 & AW solution – Framework



# TTCN-3 & AW solution – Presentation Layer

- ◆ Supports both tabular format and graphical format
  - ◆ Different to ETSI standard
  - ◆ Ease of complex value assignment
  - ◆ Similar to real test environment
  - ◆ User friendly
  - ◆ Extensible
- Can be extended by C++, Delphi

## 1. Testcase & AW



The screenshot displays the software interface for test case and AW configuration. It is divided into several panes:

- Left Pane:** A tree view showing the project structure, including folders for 'Testcase' and 'AW'.
- Top Pane:** A list of test steps with checkboxes, such as 'Prepare test environment', 'DB operation', 'Execute SQL', 'Number Assign', 'Generate CIR', 'Service Feature Management', 'Subscriber Data Management', 'Set PK Num', 'Subscriber Data Record', 'Call Flow Management', 'Call Flow', 'SMS Flow', 'USSD Flow', 'Fee Calculation Part', 'Compare Result', 'Check CIR', 'Check Subscriber Account in DB', 'Check DB', 'Check RCDM', and 'Check SMS'.
- Right Pane:** A 'Parameter Set - Number Assign' dialog box with a table of parameters and their values.

**Parameter Set - Number Assign Table:**

Action	Parameter	Value
Callisubnumber	079512345678	Shenzhen
Calling Visit Place	0-Stationary	0-Stationary
CallingFormat	1-Subscriber Num	13495012345
Called Number Attribute	4-International Num	4-International Num
Called Visit Place	91-International Num	91-International Num
CallFormat	4-International	4-International
CallForwardNumber	4-International	4-International
CallForwardNumber Format	91-International Num	91-International Num
MF Flow LocationNumber Att...		
MF Flow VLNNumber Attribute		

**Annotations:**

- A cloud-shaped callout labeled 'Test steps' points to the list of test steps in the top pane.
- A cloud-shaped callout labeled 'parameters of the "number assign"' points to the parameter table in the right pane.

**Bottom Pane:** A text area containing a detailed description of the 'Number Assign' action, explaining that it sets attributes for a calling number and call forward information, and that parameters are configured in a separate file.

## 2. User interface

Action : UC INFO Scene use text edit

Parameter	Value
Scene	Conference Setup
Role	client
Wait syn	
Set syn	
Local number	`\${_LocalFixPhone}
Access code	`\${_OutgoingAcCode}
Save	Conference_ID=conferenceId
Request	Members=075512345678&account=\${_AfterAccl}&account ...
Response	Result=success

FIN AW Assistant V1.0 【 UC INFO Scene|Role 】

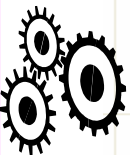
Name	Value
Scene	Conference Setup
Role	client
Wait syn	
Set syn	
Local number	`\${_LocalFixPhone}   075566661001   local telephone
Access code	`\${_OutgoingAcCode}   900   Out going access code
Save conference ID	conferenceId
Request	075512345678
*Request	Members
*Request	Account
*Request	Account password
*Request	Subject
Request	Background music
Request	Discussing work plan
Request	listen war
Request	Record
Request	Conference type
Request	Conference size
Request	Conference password
Request	123456
Response	Result
Response	success

## TTCN-3 & AW solution – Script Generator

- ◆ Convert tabular test case into TTCN-3 script
- ◆ Each TTCN-3 AW has only one parameter, containing all the AW parameters in a string, it also supports optional parameters: paramA{valueA} paramB{valueB} paramC{valueC} paramD{valueD}...
- ◆ Generate PTCs to simulate different network elements surrounds the SUT

# 1. Convert Tabular AW to TTCN-3 function

Action	DB operation
Parameter	Value
Comment	
★DB Name	ServerDB
Compound SQL	
TableName	Person
Operation Type	Update
Field Name	address age
Value	shenzhen in China 10
Where Condition	name='richa'
Input SQL	
SQL Statement	



One parameter contains all the tabular parameter value

```
SQL_AW("DBName{ServerDB},TableName{Person},Operation{Update},Fields{address|age} Values{shenzhen in China|10},Condition{name='richa'}");
```

# 2. PTCs simulate different network elements

**two PTCs**

```
function AWdemoComp1_create;
//Caller
AWdemoComp1 := AWdemoComp.create;
var AWdemoComp awdemoComp2 := AWdemoComp.create;
awdemoComp1.start(AWdemoComp1_func());
awdemoComp2.start(AWdemoComp2_func());
awdemoComp1.done;
awdemoComp2.done;

function AWdemoComp1_func() runs on AWdemoComp
{
  MakeCallFlow("signalFlowName{longDistanceCall},callTime{60},releaseRole{Calling hang up}
  IDPNType{0-Full Number},CONNNECTDest{1-user}");
  SendRCOMM("tmpRCOMMName{RecomMsg}RCOMMValue{10,20}");
}

function AWdemoComp2_func() runs on AWdemoComp
{
  MakeCallFlow("signalFlowName{longDistanceCall},callTime{60},releaseRole{Calling hang up}
  IDPNType{1-Calling Short Number},CONNNECTDest{1-user}");
}
```

### 3. Convert tabular test case to TTCN-3 test case

```

module Test_20070910154641_1 {
  import from ServerFunc all;
  testcase Test_20070910154641_1_TESTCASE() runs on AWDemoComp system AWDemoComp {
    //Preshell
    SM_AW("DBName{ServerDB}TableName{Person}Operation{Update}Fields{address|age}
    values{shenzhen_in_china|10}condition{name='ficha'}");
    PrePhoneNum("callingId{075512345678}callingSid{shenzhen}callingId{13435012345}
    farwardUrlFormat{91-International Num}");
    SetServFlag("servType{1}servFlag{args{status}1}DBName{SDU}");
    SetUserData("phoneNum{13435012345}userData{args{type}2}");
    //Caller
    var AWDemoComp awdemoComp1 := AWDemoComp.create;
    //callee
    var AWDemoComp awdemoComp2 := AWDemoComp.create;
    awdemoComp1.start(AWDemoComp1_Func());
    awdemoComp2.start(AWDemoComp2_Func());
    awdemoComp1.done;
    awdemoComp2.done;
    //AfterShell
    callFee("phoneNum{075512345678}chargeModel{1}omgDistance{chargeRate{calling}default{count}{normalCount}});
    JudgePShell("billTemplate{billCallTemplate}billType{calling_CDR}isGet{1}billNum{1}billServer{SDU}");
    CompareDatabase("DBName{ServerDataDB}table_Field_Bit{count}compareValue{count=200}condition{name='ficha'}
  }
}

function AWDemoComp1_Func() runs on AWDemoComp {
  MakeCallFlow("signalFlowName{1}omgDistance{call}callTime{60}releaseRole{calling hang up}
  IDPNumType{0-Full Number}CONNECTDest{1-user}");
  sendRCOMH("tempRCOMName{RecomMsg}RCOMValue{10,20}");
}

function AWDemoComp2_Func() runs on AWDemoComp {
  MakeCallFlow("signalFlowName{1}omgDistance{call}callTime{60}releaseRole{calling hang up}
  IDPNumType{1-Calling Short Number}CONNECTDest{1-user}");
}

```

PostShell

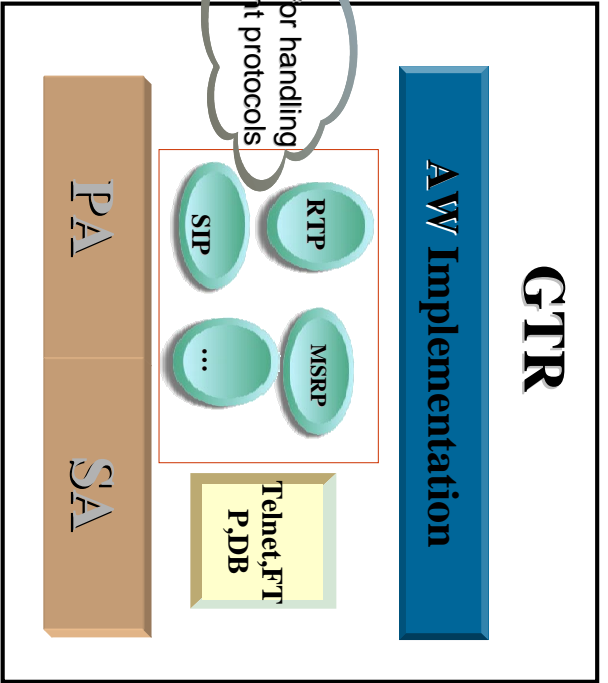
CaseStep

Preshell



## TTCN-3 & AW solution – Execution Layer

- ◆ **Action Word Implementation**  
implements AW function with TTCN-3
- ◆ **Protocol library**  
implements protocol stacks like SIP stack, with TTCN-3
- ◆ **Common library**  
implements common operations on database, file, telnet, ftp, etc, with TTCN-3 / C++ (PA) / TCL



# TTCN-3 & AW solution – Application Area

- ◆ Adapt to service testing
- ◆ Not recommended for protocol testing/ API testing
- ◆ Good to test service that is :  
driven by many protocols  
stable, less than 10% changing would be perfect  
(so that the AW test cases can be inherited )

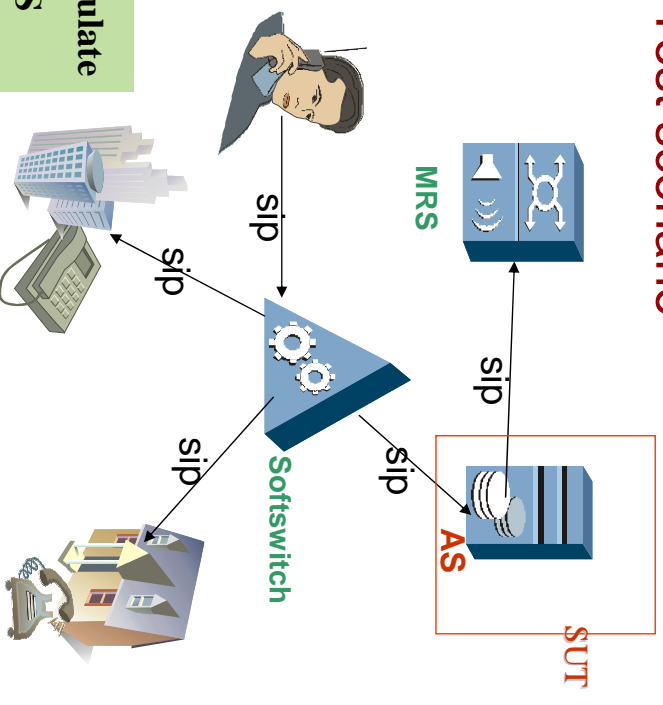
## Contents

- ◆ Service Testing
- ◆ TTCN-3 & AW solution
- ◆ **Typical application**
- ◆ Benefits
- ◆ Future work



# Typical application – Test scenario

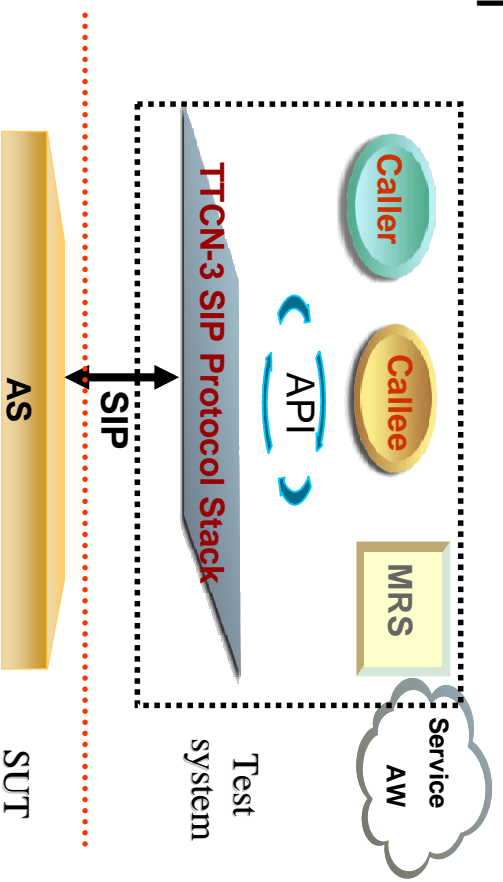
- ◆ A User dialed a virtual number
- ◆ The AS called the office phone and the home telephone at the same time
- ◆ The home telephone is picked up



◆ Here , we use TTCN-3 & AW to simulate Softswitch and MRS to test the AS

# Typical application – Test system details

- ◆ TTCN-3 SIP Protocol Stack handles sip message and SIP transaction, dialog, etc.
- ◆ Service AW are TTCN-3 functions that simulate the Softswitch (caller, callee) and MRS actions



# Typical application – Demo Testcase

The screenshot displays the 'AutotestIM View - newTest' interface. It is divided into several sections: 'Database', 'Action', and 'Parameter'. The 'Database' section lists 'Table data delete', 'DB operation', 'DB data check', 'Service data', 'Initial environment', 'Initial charge data', and 'Initial service data'. The 'Action' section lists 'Initial environment', 'DB operation', 'DB operation', 'Refresh memory data', and 'Memory tables=ALL'. The 'Parameter' section lists 'Import DB type=SDU/SCDU:', 'operation=Ipddate,DB type=SDU/SCDU, tab...', 'operation=Ipddate,DB type=SDU/SCDU, tab...', and 'Memory tables=ALL'. Below these sections is a table with columns 'No', 'Memo', 'Action', and 'Parameter'. The table contains three rows: '10 Caller', '20 Caller', and '30 Mrs'. To the right of the table is a 'Call flow' diagram showing the sequence of messages between 'Caller', 'AS', 'MRS', and 'Callee'. The sequence is: Caller sends 'invite' to AS; AS sends 'invite' to MRS; MRS sends 'invite' to Callee; Callee sends '200' to MRS; MRS sends 'ack' to AS; AS sends '200' to Caller; Caller sends 'ack' to AS. A cloud labeled 'Call flow' is positioned above the diagram.

## Contents

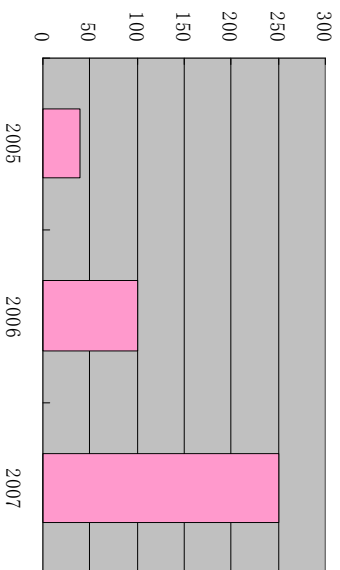
- ◆ Service Testing
- ◆ TTCN-3 & AW solution
- ◆ Typical application
- ◆ **Benefits**
- ◆ Future work



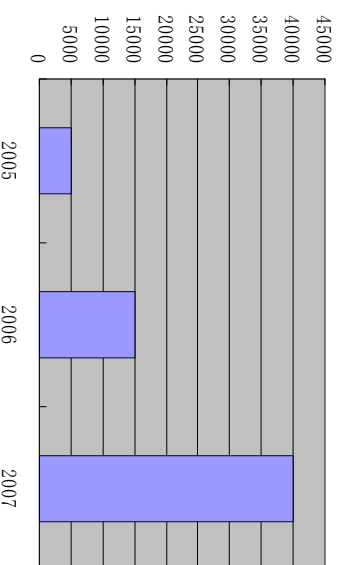
# Benefits



Total users



Total test cases



# Contents

- ◆ Service Testing
- ◆ TTCN-3 & AW solution
- ◆ Typical application
- ◆ Benefits
- ◆ **Future work**



# Future work

- ◆ **AW design with layers**  
High level AW can be implemented by lower level  
AWs
- ◆ **Object oriented AW design**  
With object oriented design, it is similar to real  
entity
- ◆ **Integrate AW implemented in C++/ TCL/ java**  
Share AW pool largely



# Thank you!