TESTING MICRO SERVICES

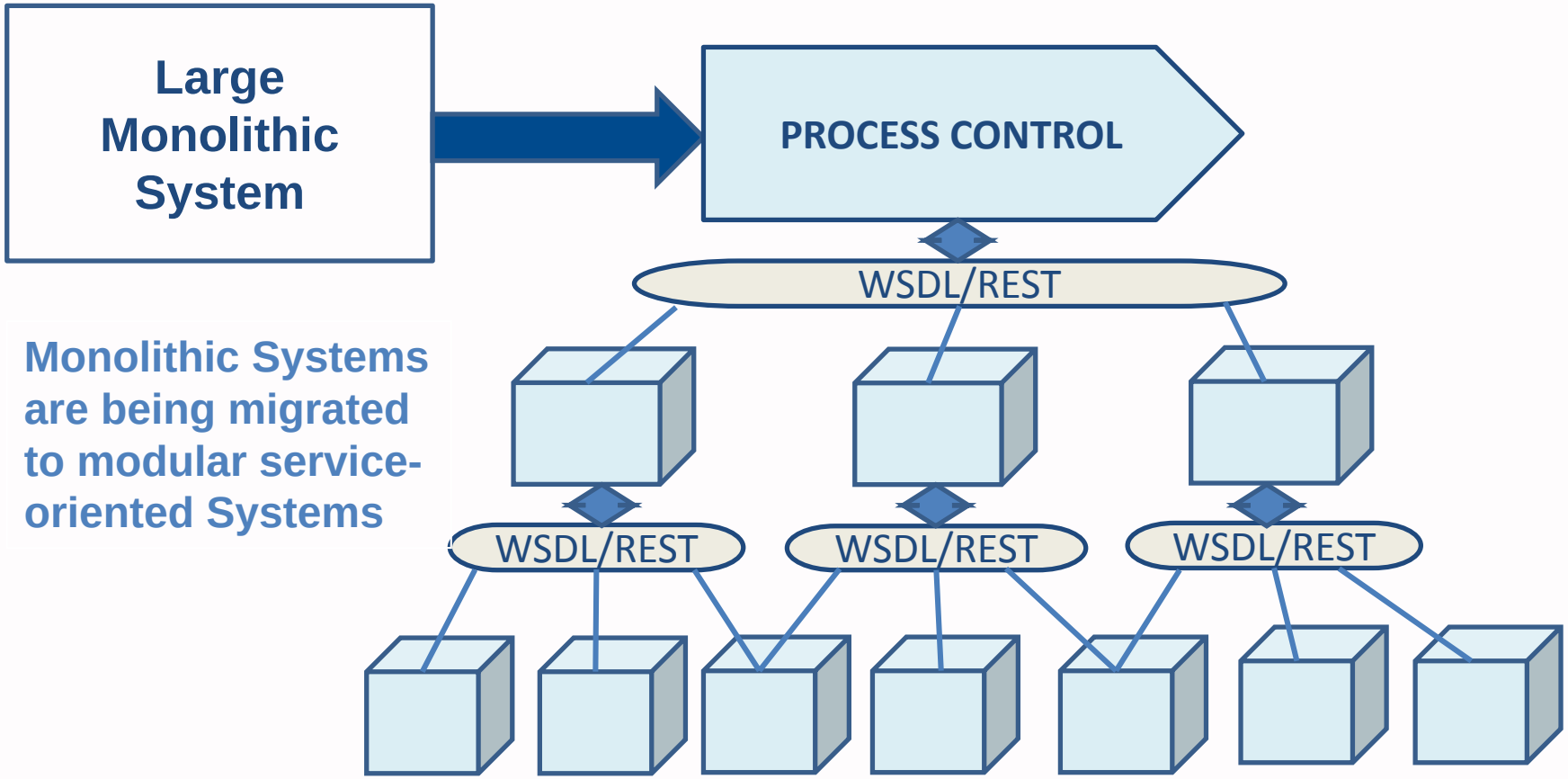
Presented by Harry M. Sneed, SoRing Kft.

DEFINING MICRO SERVICES?

DESIGNING FLEXIBLE, EXTENDABLE SYSTEMS

SoDing

FROM MONOLITHIC TO MODULAR SYSTEMS

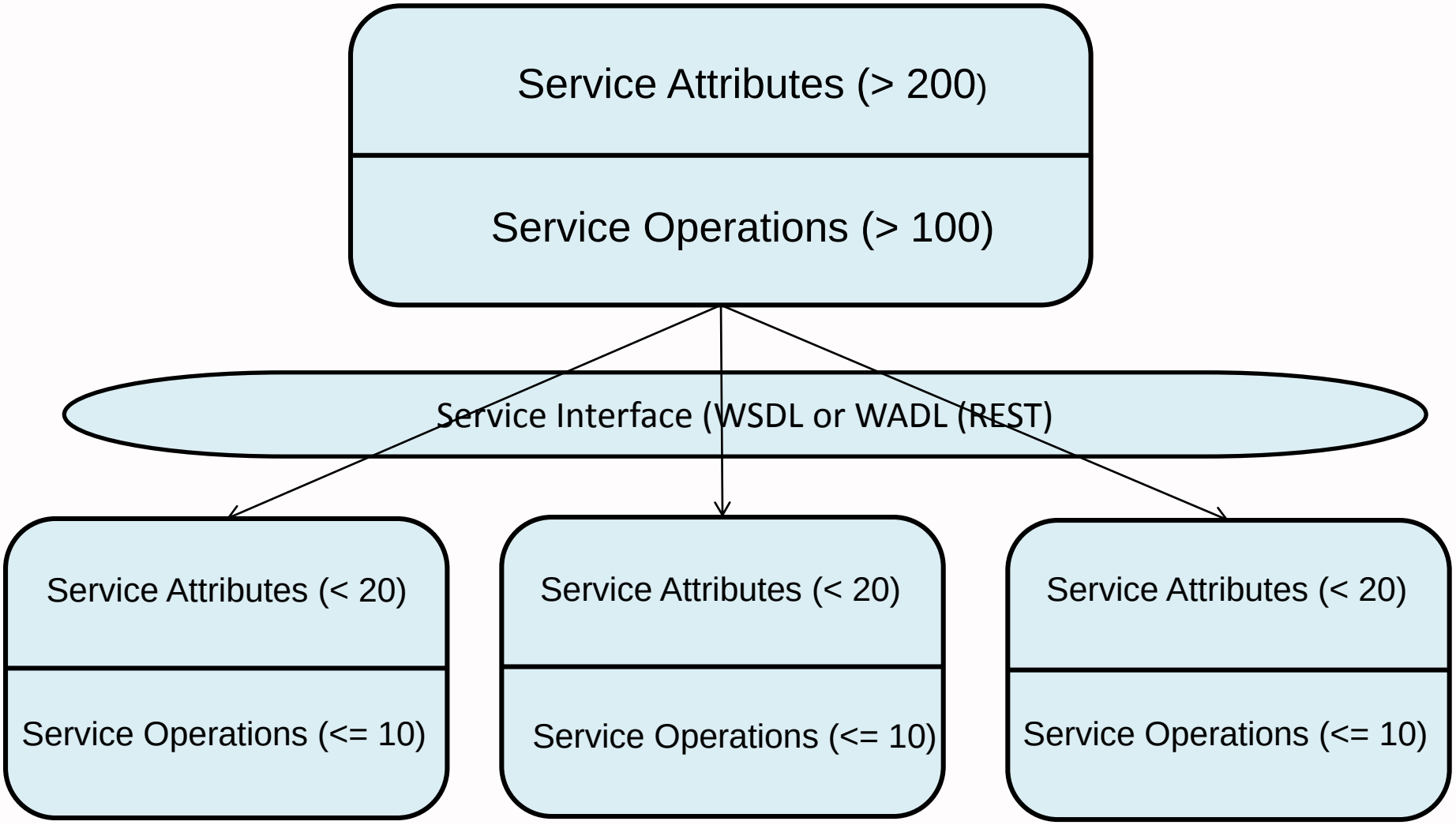


Monolithic Systems are being migrated to modular service-oriented Systems

CREATING TESTABLE SERVICES

- The goal is to be able to develop and test a service within a week
- Since testing makes up for at least 50% of the costs, a micro service should be testable with 3 days.
- Based on the current test productivity and the desired level of test coverage, no service should require more than n test cases.
- The number of test cases is determined by the number of parameters and operations. Within one service they must be limited.

LIMITING THE SIZE OF SERVICES



5th UCAAT User Conference on Advanced Automated Testing

ETSI

TESTING MICRO SERVICES

CHALLENGE OF SERVICE TESTING

DEGREES OF MICRO SERVICE COVERAGE

TESTING A MONOLITHIC WEB SERVICE

TESTING A SET OF MICRO SERVICES

MICRO SERVICE TEST PROCESS

SoDing

CHALLENGES FOR THE SERVICE TEST

- Testing Levels
  - Micro-Services are minute self-contained software components with a standard message-oriented interface (REST or WSDL) which allows them to be connected with one another.
  - They have to first be tested independently via their interface with selected parameters.
  - They are then tested in interconnected groups – components via an integration interface
  - Finally, they are tested as a whole as part of a larger system via the user interface

DEGREES OF MICRO SERVICE COVERAGE

- testing each individual operation in a service (non-modal testing)
- testing all combinations of operations in a service, i.e. all paths (semi-modal testing)
- testing each individual operation with all relevant states (quasi-modal testing)
- testing all sequences of all operations with all combinations of relevant states (modal-testing).

Robert Binder in Object-oriented Testing [Binder1995]:

TESTING A MONOLITHIC WEB SERVICE

A Java micro service for managing an account contains

- 3495 statements
- 51 classes
- 202 methods
- 28 interfaces.

The static analysis of this service reveals

- 1309 branches
- 644 paths.

The Test with Selenium requires

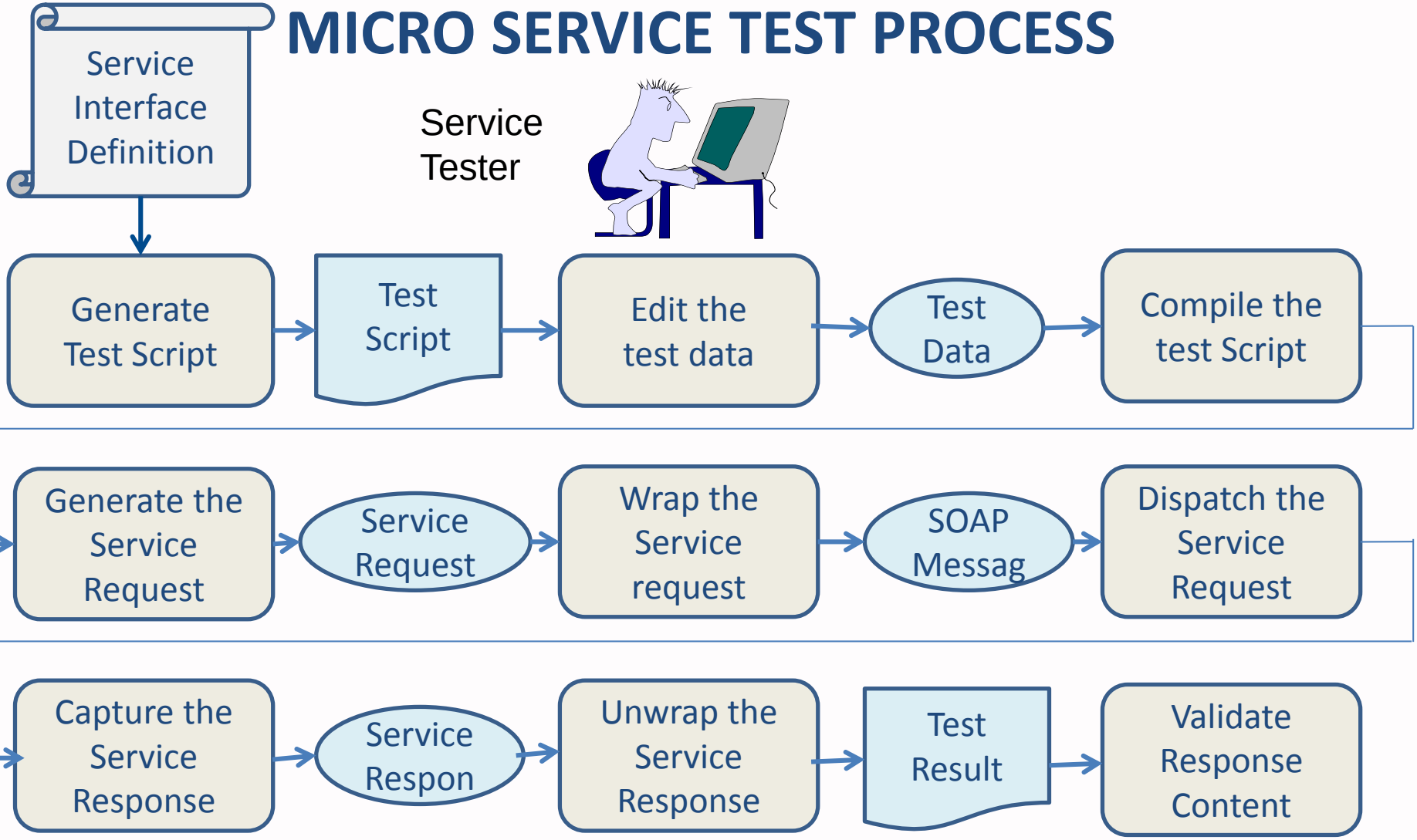
- 16 Person days
- for the 640 test cases

TESTING A SET OF MICRO SERVICES

The same Java Web Service is broken down into

- 5 Micro Services, each with <= 700 statements
- It requires an average of 144 test cases per service
- Each service can be tested within 3,5 days
- With Selenium it takes 17,5 person days to test the whole
- With ServTest it takes less than 2 days to test each service
- All 720 test cases can be tested within 10 days or 2 Weeks
- That corresponds to 72 test cases per tester day
- With Selenium the test productivity was 40 test cases per day.
- The difference is due to the automated simulation of the service interfaces.

MICRO SERVICE TEST PROCESS



5th UCAAT User Conference on Advanced Automated Testing

ETSI

TESTING MICRO SERVICES WITH TESTSERV

STARTING WITH THE SERVICE INTERFACE DEFINITION

SERVICE TEST DATA FLOW

SERVICE TEST SCRIPT

SERVICE TEST RESULT

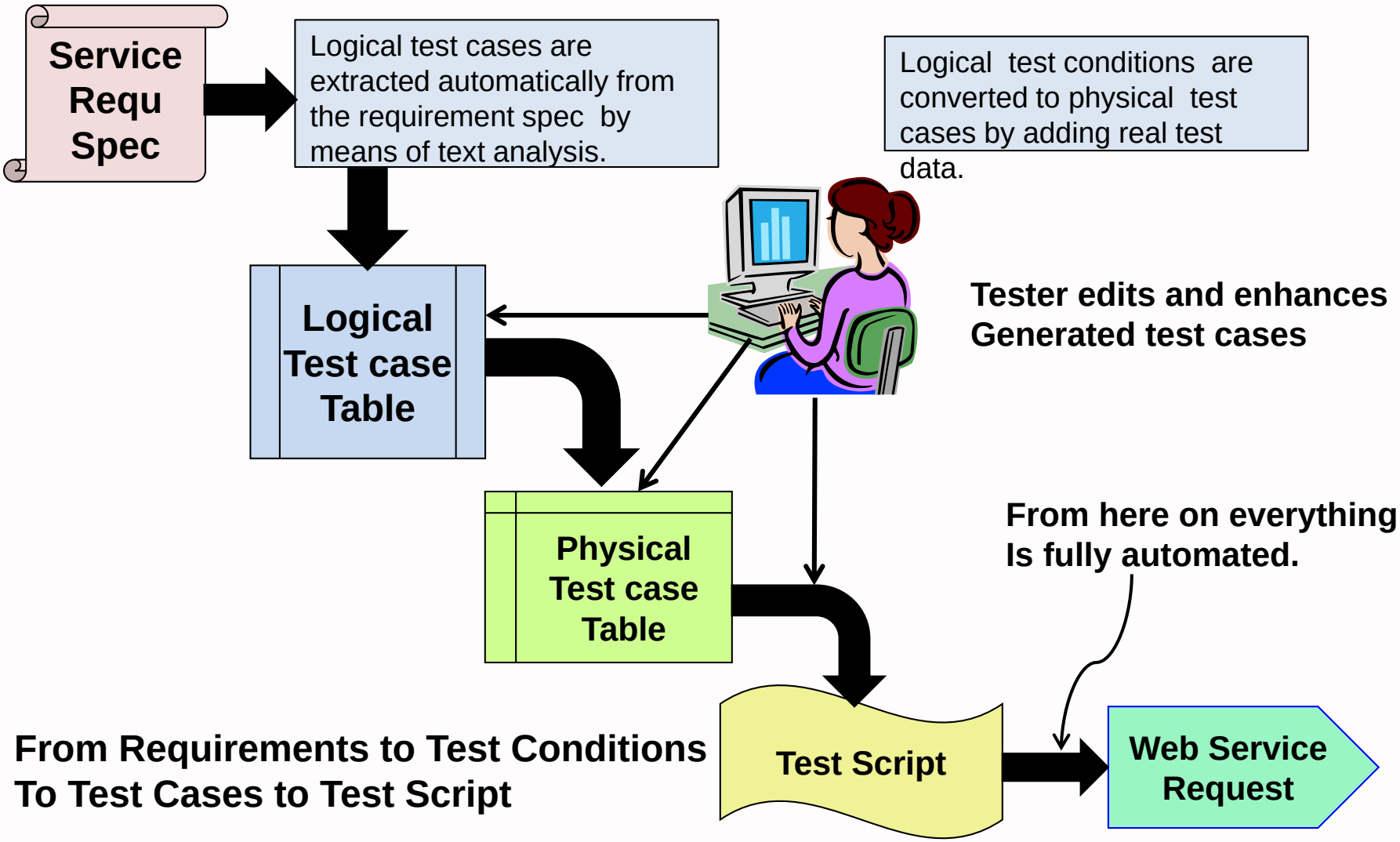
SERVICE TEST DASHBOARD

SoDing

SERVICE INTERFACE DEFINITION

```
<complexType name="ArticleItem">
                <sequence>
                    <element name="ResponseArtNo" type="xs:string"/>
                    <element name="ResponseArtType" type="xs:string"/>
                    <element name="ResponseArtName" type="xs:string"/>
                    <element name="ResponseArtPrice" type="xs:double"/>
                </sequence>
            </complexType>
            <complexType name="Order">
                <sequence>
                    <element name="CustNo" type="xs:string"/>
                    <element name="OrderNo" type="xs:int"/>
                    <element name="DeliveryMode" type="xs:string"/>
                    <element name="PaymentMode" type="xs:string"/>
                    <element name="Orders" type="ns1:ArrayofOrderItem"/>
                </sequence>
            </complexType>
            <complexType name="OrderItem">
                <sequence>
                    <element name="OrderArtNo" type="xs:string"/>
                    <element name="OrderArtType" type="xs:string"/>
                    <element name="OrderArtName" type="xs:string"/>
                    <element name="OrderArtAmount" type="xs:int"/>
                </sequence>
            </complexType>
        </schema>
    </types>
```

SERVICE TEST DATA FLOW



From Requirements to Test Conditions To Test Cases to Test Script

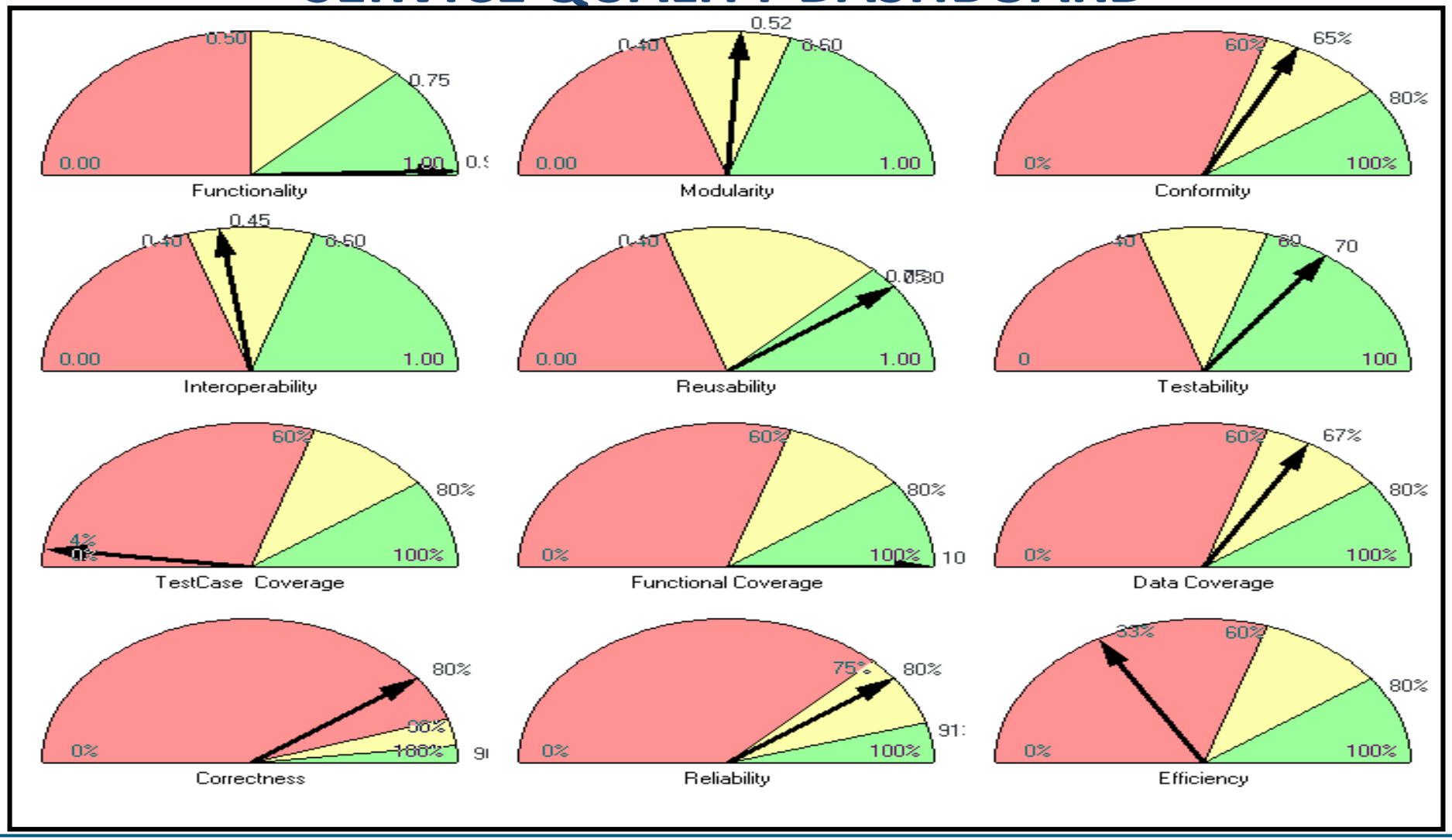
MICRO SERVICE TEST SCRIPT

```
//  First Request to Frontend to order Articles
      if ( operation =   "GetTypes");
         if ( request  =   "GetTypes1Request");
            assert inp.GetTypes1Request_DummyParam  = "?";
         endRequest ;
         if ( response  =   "GetTypes1Response");
            assert out.$ResponseTime < "1000";
            if ( object = "return" occurs = "2");
               assert out.item[1]  = old.item[1];
           endObject;
         endResponse ;
      endOperation;
      if ( operation =   "QueryAvailableProducts");
         if ( request  =   "QueryAvailableProducts2Request");
            assert inp.CustNo  = "009999";
            assert inp.ArtType  = "BOOK";
         endRequest ;
         if ( response  =   "QueryAvailableProducts2Response");
           assert out.$ResponseTime < "1000";
```

SERVICE TEST RESULT

Non-Matching Params	Non-Matching Values
Resp-Id: GetTypes1Response_003	
Ist: $ResponseTime	2059
Soll:Asserted_Value	<1000
Resp-Id: GetTypes1Response_003_return	
Ist: item[2]	BOOK
Soll:Asserted_Value	=NEWS
Ist: item[3]	0
Soll:Asserted_Value	=BOOK
Resp-Id: QueryAvailableProducts2Response_003	
Ist: $ResponseTime	2580
Soll:Asserted_Value	<1000
Resp-Id: QueryAvailableProducts2Response_003_return_item[1]	
Ist: ResponseArtNo	9999
Soll:Asserted_Value	=4711

SERVICE QUALITY DASHBOARD



SUMMARY

- MICRO SERVICES IS THE CURRENT TREND IN SOFTWARE DEVELOPMENT
- MICRO SERVICES MUST BE DESIGNED TO BE TESTABLE
- THE SERVICES MUST BE KEPT SMALL AND SIMPLE
- THE SERVICE TEST MUST BE AUTOMATED
- SERVTEST IS AVAILABLE FREE OF COST ON THE SORING WEBSITE