The Standards People

EG 203 647:
Methods for Testing and Specification (MTS);
Methodology for RESTful APIs specifications and testing

Presented by: **Martti Käärik**
Philip Makedonski
ETSI STF 576

For: **oneM2M TP#47**

28.09.2020

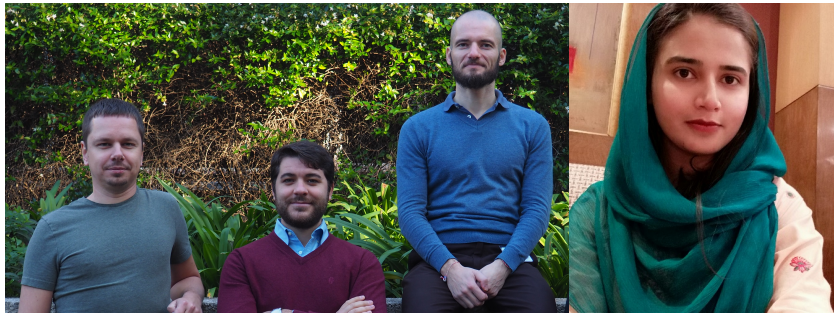
Agenda

- Motivation
- Overview of EG 203 647
- REST API specification methodology
- REST API testing methodology
- Examples, tooling
- Conclusions and lessons learned
- Next steps

TC MTS and STF 576

- TC MTS
  - Chair: Dirk Tepelmann
  - Developing methodology, languages and ecosystem for Testing
  - TTCN-3, TDL
  - TDL Open Source project (https://labs.etsi.org/rep/top)

- STF 576
  - Martti Käärik, Elvior
  - Philip Makedonski, University Göttingen
  - Sana Zulfiqar, xFlow Research
  - Michele Carignani, ETSI Technical Expert



https://portal.etsi.org/TB-SiteMap/MTS/Summary

https://portal.etsi.org/STF/STFs/STF-HomePages/STF576

Background

Motivation

- REST API specifications more and more relevant in ETSI groups
  - ISG Multi-access Edge Computing, ISG Network Function Virtualisation, ISG Zero-touch Service Management, ISG Cross-cutting Information Management, 3GPP CT & SA, OneM2M
- Guidelines, methodologies and conventions needed
  - Led to development of MEC 009, NFV-SOL 013 & 015, …

Objective

- EG 203 647: Methods for Testing and Specification (MTS); Methodology for RESTful APIs specifications and testing

  - Scope: guidance and methodology for RESTful API specification and testing
  - Benefits: Consolidation of efforts, consistency and quality of standards
  - See WI description for further information

Overview of the EG

Specification methodology

- Introduction on RESTful Interfaces
- API Specification Process
- Common Patterns and Naming Conventions
- Versioning and Implementation Details

Testing methodology

- Testing Frameworks and Methodologies
- Conformance and Interoperability Testing
- Test Specification Development
- Test Deployment and Execution
- Test Maintenance and Evolution

Tooling Recommendations

- Design and Drafting
- Coordination and Collaboration
- Validation and Quality Checking
- Post Processing

Working Examples

- OpenAPI
- TDL / TDL-TO
- Word Documents
- TTCN-3
- Robot Framework

Survey of Activities

- ETSI and Beyond
- Review of 14 Base Documents
- API Adoption Survey

Overview of the EG

Specification methodology

- Introduction on RESTful Interfaces
- API Specification Process
- Common Patterns and Naming Conventions
- Versioning and Implementation Details

Testing methodology

- Testing Frameworks and Methodologies
- Conformance and Interoperability Testing
- Test Specification Development
- Test Deployment and Execution
- Test Maintenance and Evolution

Tooling Recommendations

- Design and Drafting
- Coordination and Collaboration
- Validation and Quality Checking
- Post Processing

Working Examples

- OpenAPI
- TDL / TDL-TO
- Word Documents
- TTCN-3
- Robot Framework

Survey of Activities

- ETSI and Beyond
- Review of 14 Base Documents
- API Adoption Survey

API Specification Process

- Present challenges in standardisation of APIs
  - Duplication, redundancy, inconsistencies
  - Maintenance overhead (manual processes)
- Benefits of a machine-readable first approach
  - Well-known specification format (e.g. OpenAPI)
  - Reduced overhead, single source of truth
  - Syntax correctness and semantic validation
- ETSI specific information using extensions
- Linked with the standardized document


```yaml
/service:
  get:
    # Info excluded
    operationId: getService
    # Support for GET request is mandatory for API provider
    x-etsi-provision: mandatory
    parameters:
      - name: 'circuitswitching'
        in: query
        required: false
        schema:
          type: string
        x-etsi-capabilities:
          # Parameter only applies to "3G" capability
          - 3G
    responses:
      200:
        description: 'The requested service'
        content:
          application/json:
            schema:
              type: object
              properties:
                speed:
                  type: string
                  enum:
                    - fast
                    - superfast
                x-etsi-enum:
                  # Enum value "superfast" is optional and
                  # only applies to "4G" and "5G" capabilities
                  superfast:
                    required: false
                    x-etsi-capabilities:
                      - 4G
                      - 5G
```

Figure 4.3.2.9-1: Example of usage of extensions

Common Patterns and Conventions

- Collection from standards and industry:
  - Filtering Patterns
  - URI Creation
  - Update Conflict resolution
  - Authorization and Authentication
  - Non-CRUD operations
  - Naming conventions

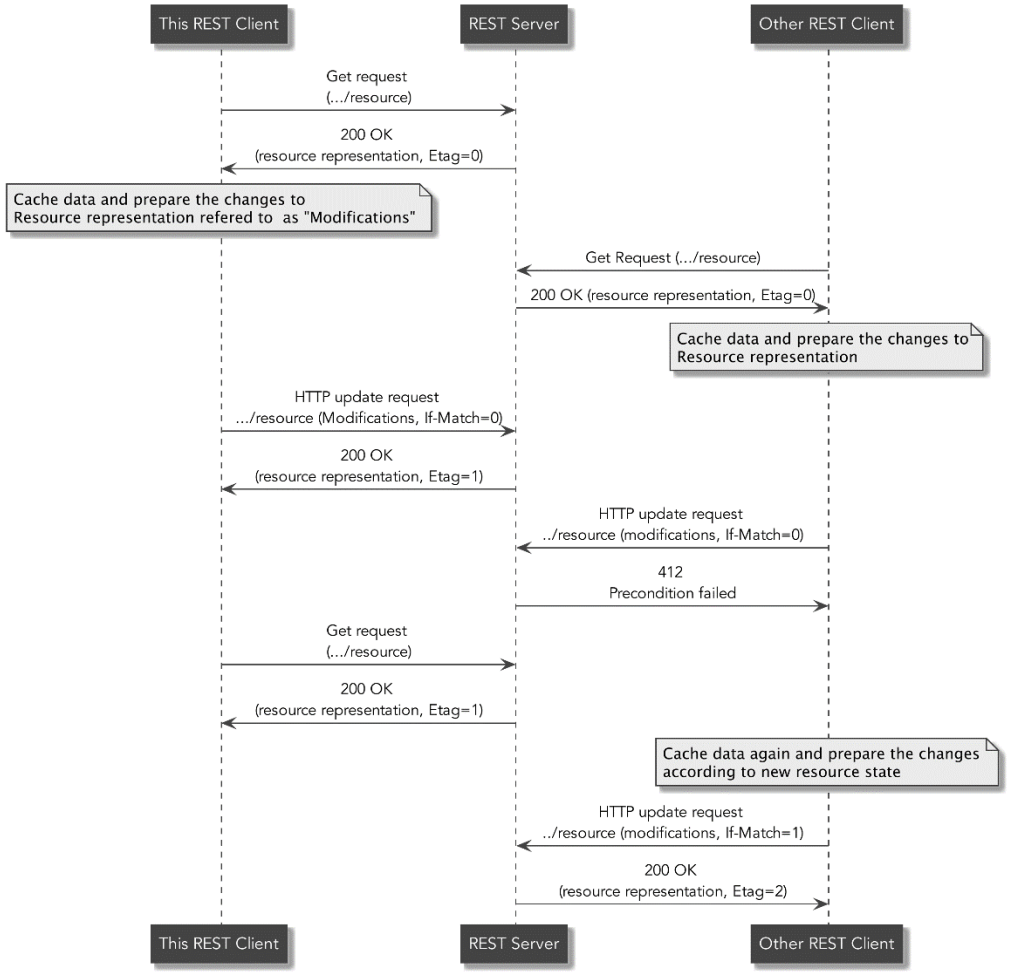


Figure 4.4.3.1-1: Flow of concurrent update of a Resource

Versioning and Implementation Details

- Versioning
  - Align document and OpenAPIs versions
  - Modelling versioning information

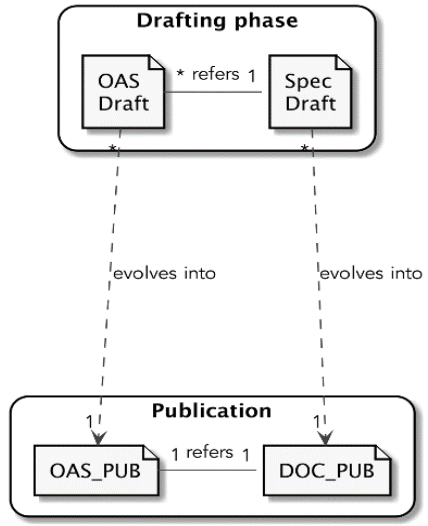


Figure 4.6.1-1: Versioning model

- Implementation details
  - Vendor specific versioning
  - Proprietary features and extensions

```
/res1: # Standardized API feature
  # Content excluded
/res2: # Proprietary API feature
  description: ...
  x-etsi-proprietary-capability:
  - ACMEAuthSystem
  # Content excluded
```

Figure 4.7-1: Usage example for the proprietary capability extension

Overview of the EG

Specification methodology

- Introduction on RESTful Interfaces
- API Specification Process
- Common Patterns and Naming Conventions
- Versioning and Implementation Details

Testing methodology

- Testing Frameworks and Methodologies
- Conformance and Interoperability Testing
- Test Specification Development
- Test Deployment and Execution
- Test Maintenance and Evolution

Tooling Recommendations

- Design and Drafting
- Coordination and Collaboration
- Validation and Quality Checking
- Post Processing

Working Examples

- OpenAPI
- TDL / TDL-TO
- Word Documents
- TTCN-3
- Robot Framework

Survey of Activities

- ETSI and Beyond
- Review of 14 Base Documents
- API Adoption Survey

Test Specification Development in General (EG 203 130)

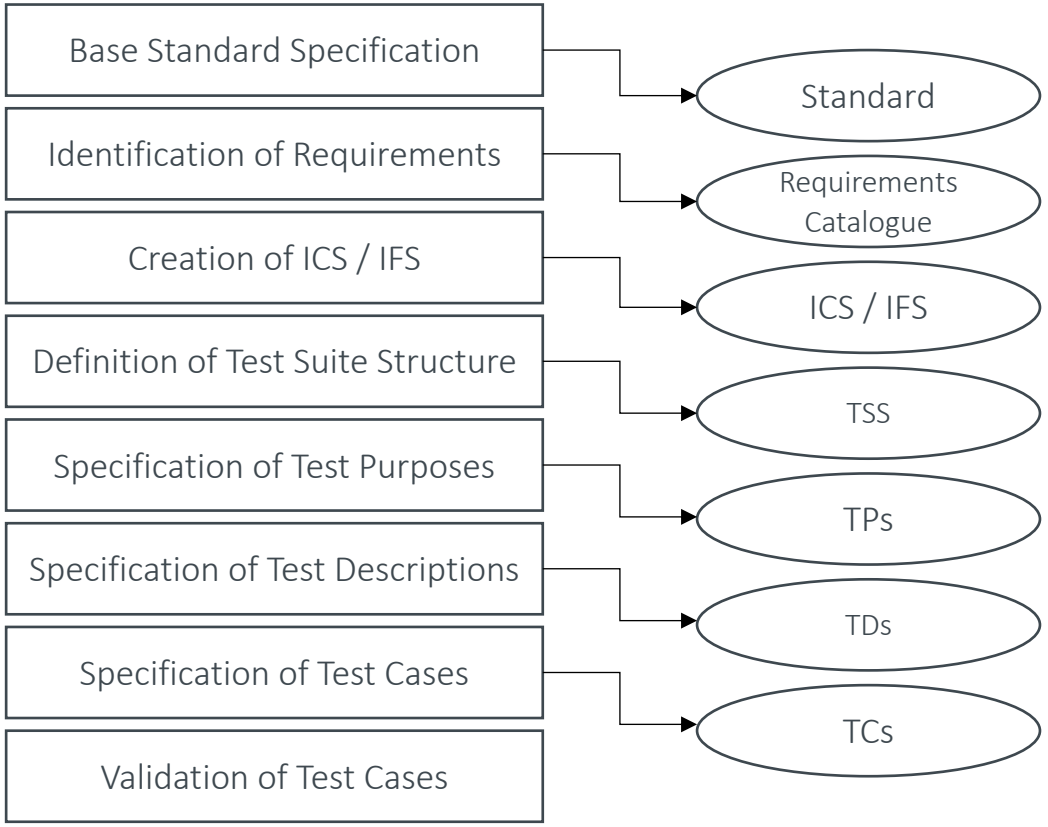


Figure 5.4.1-1: ETSI Test Specification Process (based on ETSI EG 203 130 [i.4])

Test Specification Development based on OpenAPIs

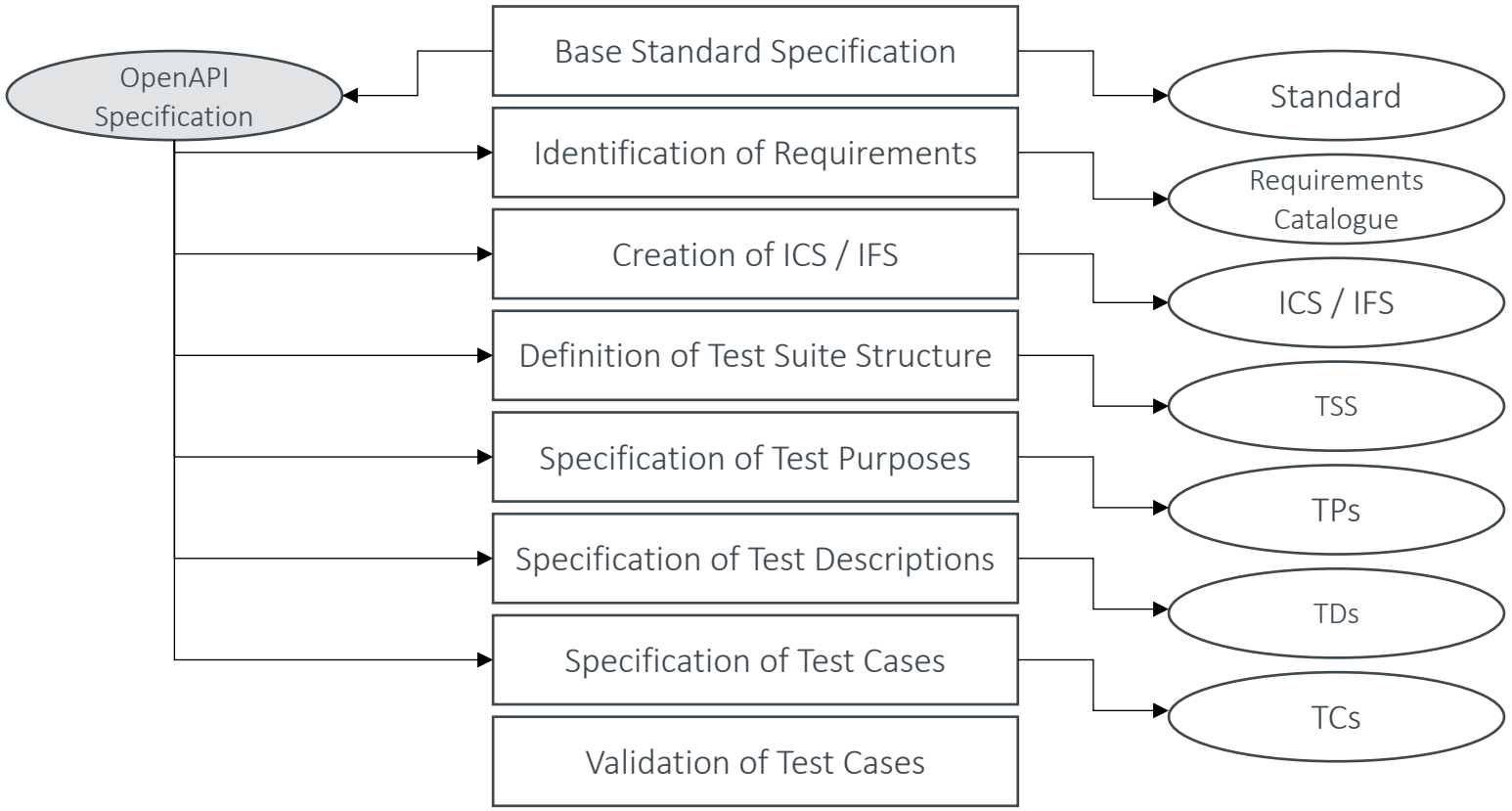


Figure 5.4.2-1: RESTful API Test Specification Process with OpenAPI

Test Specification Development based on OpenAPIs

```yaml
paths:
  '/resource/{id}':
    get:
      # Request and parameters excluded
      responses:
        # Response code
        200:
          description: 'The requested resource'
          # Custom headers
          headers:
            ETag:
              # Reference to (reusable) header definition
              $ref: '#/components/headers/ETag'
          # Response body
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/ResourceData'
        401:
          # Reference to (reusable) response definition
          $ref: '#/components/responses/401'
        404:
          $ref: '#/components/responses/404'
components:
  responses:
    # Common responses with response code as identifier
    204:
      description: 'No content'
    401:
      description: 'Unauthenticated'
    404:
      description: 'Not found'
  headers:
    # Definition of ETag header
    ETag:
      description: 'Identifier for a specific version of a resource'
      schema:
        type: string
```

Figure 4.3.2.6-1: Examples of response definitions

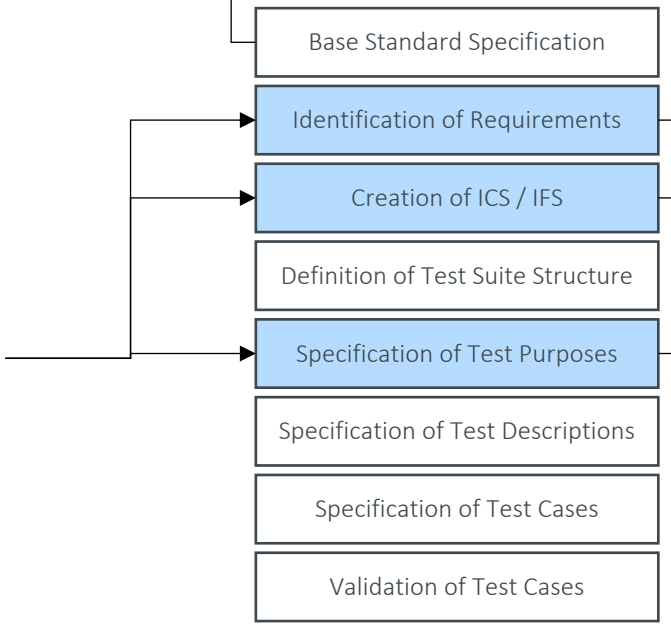


Table 5.4.2-1: Extracted requirements

Identifier	Reference	Type	Applicability	Requirement	Context
RQ_RESOURCE_GET_001	RESTful API Guide, clause 4.3.2.4; /resource/{id}	M	RESOURCE	Read full contents of a resource with specific ID (description from OpenAPI™)	(relevant OpenAPI™ specification fragment)
RQ_RESOURCE_POST_001	RESTful API Guide, clause 4.3.2.5; /resource	M	RESOURCE	Create new resource (description from OpenAPI™)	(relevant OpenAPI™ specification fragment)
RQ_RESOURCE_PUT_001	RESTful API Guide, clause 4.3.2.5; /resource/{id}/file	M	RESOURCE	Upload a file for a resource (description from OpenAPI™)	(relevant OpenAPI™ specification fragment)

Table 5.4.2-3: Extracted higher (method) level ICSs

ID	Resource	Reference	Method	Type	Request	Response
M1	/resource/{id}	Clause 4.3.2.6	GET	M	(request resource with id)	200, 401, 404
M2	/resource	Clause 4.3.2.5	POST	M	(create new resource with id and version)	201, 400
M3	/resource	Clause 4.3.2.5	PUT	M	(update a resource with id)	200, 201, 204, 400

TP Id	TP_RESOURCE_GET
Test Objective	Read full contents of a resource with an ID
Reference	Clause 4.3.2.4 Clause 4.3.2.6
Expected Behaviour	

```
ensure that {
    when {
        the Server receives a vGET request containing
            uri indicating value "/resource/",
            id set to ID
    }
    then {
        the Server sends a HTTP response containing
            status set to HTTP_STATUS
    }
}
```

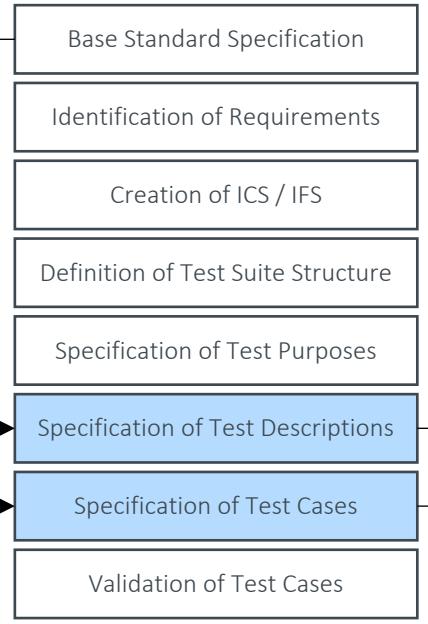
TP Id	Description	ID	HTTP_STATUS
TP_RESOURCE_GET_200v1	"Read full contents of a resource with a valid ID"	VALID_ID	"200 OK"
TP_RESOURCE_GET_404v2	"Read contents of a resource with a non-existent ID returns 404"	NONEXISTENT_ID	"404 Not found"

Figure 5.4.2-7: Tabular presentation of example TPs with variants

Test Specification Development based on OpenAPIs

```yaml
paths:
  '/resource/{id}':
    get:
      # Request and parameters excluded
      responses:
        # Response code
        200:
          description: 'The requested resource'
          # Custom headers
          headers:
            ETag:
              # Reference to (reusable) header definition
              $ref: '#/components/headers/ETag'
          # Response body
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/ResourceData'
        401:
          # Reference to (reusable) response definition
          $ref: '#/components/responses/401'
        404:
          $ref: '#/components/responses/404'
components:
  responses:
    # Common responses with response code as identifier
    204:
      description: 'No content'
    401:
      description: 'Unauthenticated'
    404:
      description: 'Not found'
  headers:
    # Definition of ETag header
    ETag:
      description: 'Identifier for a specific version of a resource'
      schema:
        type: string
```

Figure 4.3.2.6-1: Examples of response definitions



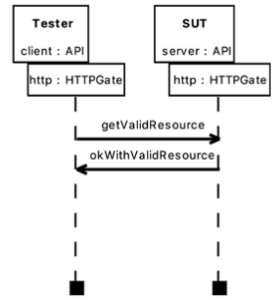


Figure 5.4.2-10: Graphical presentation of test description with valid ID

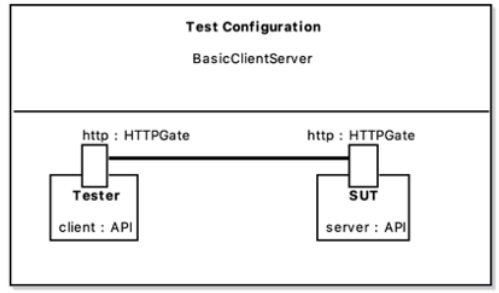


Figure 5.4.2-14: Graphical presentation of test configuration example

```
function setupTestConfiguration_BasicClientServer ( ) runs on MTC_BasicClientServer {
    client := API.create ;
    map ( client : http_to_server_http , system : server_http ) ;
}

function f_RESOURCE_GET_200 ( ) runs on MTC_BasicClientServer {
    client.start ( f_RESOURCE_GET_200_client_main ( ) ) ;
}

function f_RESOURCE_GET_200_client_main ( ) runs on API {
    http_to_server_http.send ( getValidResource ) ;
    http_to_server_http.receive ( okWithValidResource ) ;
}

testcase tc_RESOURCE_GET_200 ( ) runs on MTC_BasicClientServer system SYSTEM_BasicClientServer {
    setupTestConfiguration_BasicClientServer ( ) ;
    f_RESOURCE_GET_200 ( ) ;
    all component.done ;
}
```

Figure 5.4.2-15: Derived test case example and supporting functions for behaviour and configuration

Overview of the EG

Specification methodology

- Introduction on RESTful Interfaces
- API Specification Process
- Common Patterns and Naming Conventions
- Versioning and Implementation Details

Testing methodology

- Testing Frameworks and Methodologies
- Conformance and Interoperability Testing
- Test Specification Development
- Test Deployment and Execution
- Test Maintenance and Evolution

Tooling Recommendations

- Design and Drafting
- Coordination and Collaboration
- Validation and Quality Checking
- Post Processing

Working Examples

- OpenAPI
- TDL / TDL-TO
- Word Documents
- TTCN-3
- Robot Framework

Survey of Activities

- ETSI and Beyond
- Review of 14 Base Documents
- API Adoption Survey

Overview of the EG

Specification methodology

- Introduction on RESTful Interfaces
- API Specification Process
- Common Patterns and Naming Conventions
- Versioning and Implementation Details

Testing methodology

- Testing Frameworks and Methodologies
- Conformance and Interoperability Testing
- Test Specification Development
- Test Deployment and Execution
- Test Maintenance and Evolution

Tooling Recommendations

- Design and Drafting
- Coordination and Collaboration
- Validation and Quality Checking
- Post Processing

Working Examples

- OpenAPI
- TDL / TDL-TO
- Word Documents
- TTCN-3
- Robot Framework

Survey of Activities

- ETSI and Beyond
- Review of 14 Base Documents
- API Adoption Survey

Examples

- Extensive set of examples at ETSI Forge
  - OpenAPI
  - Requirements, ICSs, Test Suite Structure
  - TDL-TO Test Purposes / TDL Test Descriptions
  - Generated Word documents
  - TTCN-3 test cases
  - Robot Framework tests
- Two flavours
  - Manually derived (more detailed)
  - Automatically generated (prototype)

Name	Last commit
OpenAPI	Examples by STF 576
Robot	Added Robot example.
TC	* updated generated TTCN-3
TD	Examples by STF 576
TP	Examples by STF 576
LICENSE	Licensing information
README.md	Examples by STF 576
RQ-ICS-TSS.md	Examples by STF 576
rq-ics-tss.docx	Examples by STF 576

Overview of the EG

Specification methodology

- Introduction on RESTful Interfaces
- API Specification Process
- Common Patterns and Naming Conventions
- Versioning and Implementation Details

Testing methodology

- Testing Frameworks and Methodologies
- Conformance and Interoperability Testing
- Test Specification Development
- Test Deployment and Execution
- Test Maintenance and Evolution

Tooling Recommendations

- Design and Drafting
- Coordination and Collaboration
- Validation and Quality Checking
- Post Processing

Working Examples

- OpenAPI
- TDL / TDL-TO
- Word Documents
- TTCN-3
- Robot Framework

Survey of Activities

- ETSI and Beyond
- Review of 14 Base Documents
- API Adoption Survey

Survey of Activities

- Topics highlighted in the survey
  - Major challenges in the specification of REST API
    - Inconsistency of and between documents
    - Growing size of documents
  - Important Topics to be covered in RESTful API Guide
    - Validation of API specifications
    - Selection and usage of tools
    - Examples of standard deliverables (TPs, TDs, ATS)
  - Importance of Interoperability and Conformance testing
- Base documents reviewed
  - ETSI: MEC 009 (V2.1.1), MEC-DEC 025 (V2.1.1), MEC-DEC 032-1 (V0.0.3), CIM 009 (V1.2.1), QKD 014 (V1.1.1), TS 129 501 (V15.3.0), NFV-SOL 013 (V2.7.1), NFV-SOL 015 (V1.1.1), NFV-TST 010 (V2.4.1), TS 118 115 (V2.0.0)
  - Other SDOs: oneM2M TS-0018 (V3.2.0), TM Forum Open APIs initiative, OMG hData RESTful Transport, OASIS OData v4.01

Conclusions, Lessons Learned

- Conclusions
  - A lot of work has been done and we could benefit from it
  - The goal is for standardization processes to benefit from consolidation
- Collaboration is welcome
  - Put the guide in practice
  - Coordinate on future work and goals
- How to give feedback
  - Subscribe to the public REST mailing list (Link)
  - File issues and requests at the ETSI Forge Bug tracker (Link)
  - Join MTS meetings (Link)

Next Steps

- Membership Voting until Nov 6 (Link – EOL account required)

- Stay tuned for a possible feedback survey (TBC)
- Possible normative work (e.g. specification of OpenAPI extensions)
- Future developments of the Guide to include topics such as:
  - Detailed process of specification development
  - Testing languages (TDL, TTCN-3) development to better accommodate REST testing
  - Your suggestions!

Thanks!

Philip Makedonski

REST@list.etsi.org