Model Writer

Progress on the Industrial Use Cases

UC-FR-01 - Synchronization between models and documents

Yvan Iussaud
OBEO

ITEA3

UC-FR-01

Synchronization between models and documents

Sirius

Development of Sirius

Problematic

Concern

Open source Eclipse project

Domain Specific language

Creates graphical modelers

View point approach

Easy to use (Diagram, table, tree, …)

Development of Sirius

Artifacts

- Graphical Mapping DSL
- Source code
- Documentation (specifier, user)

Life cycle

- Release train Eclipse
- Dependencies

Team

- Core team
- External contributors

Methodology
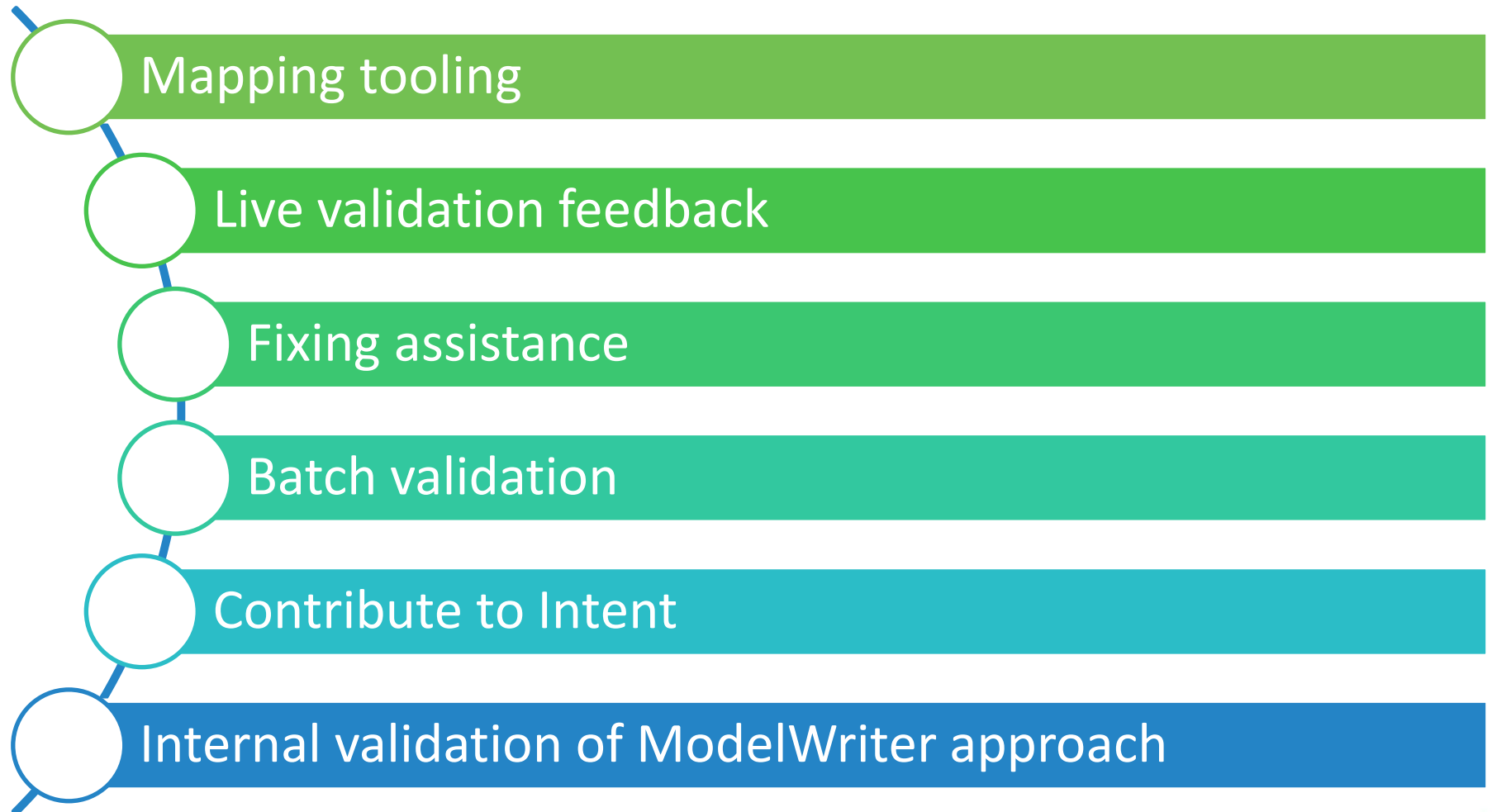
- Code review
- Plan manual synchronization of artifacts

Problematic

Synchronization gap

- Development
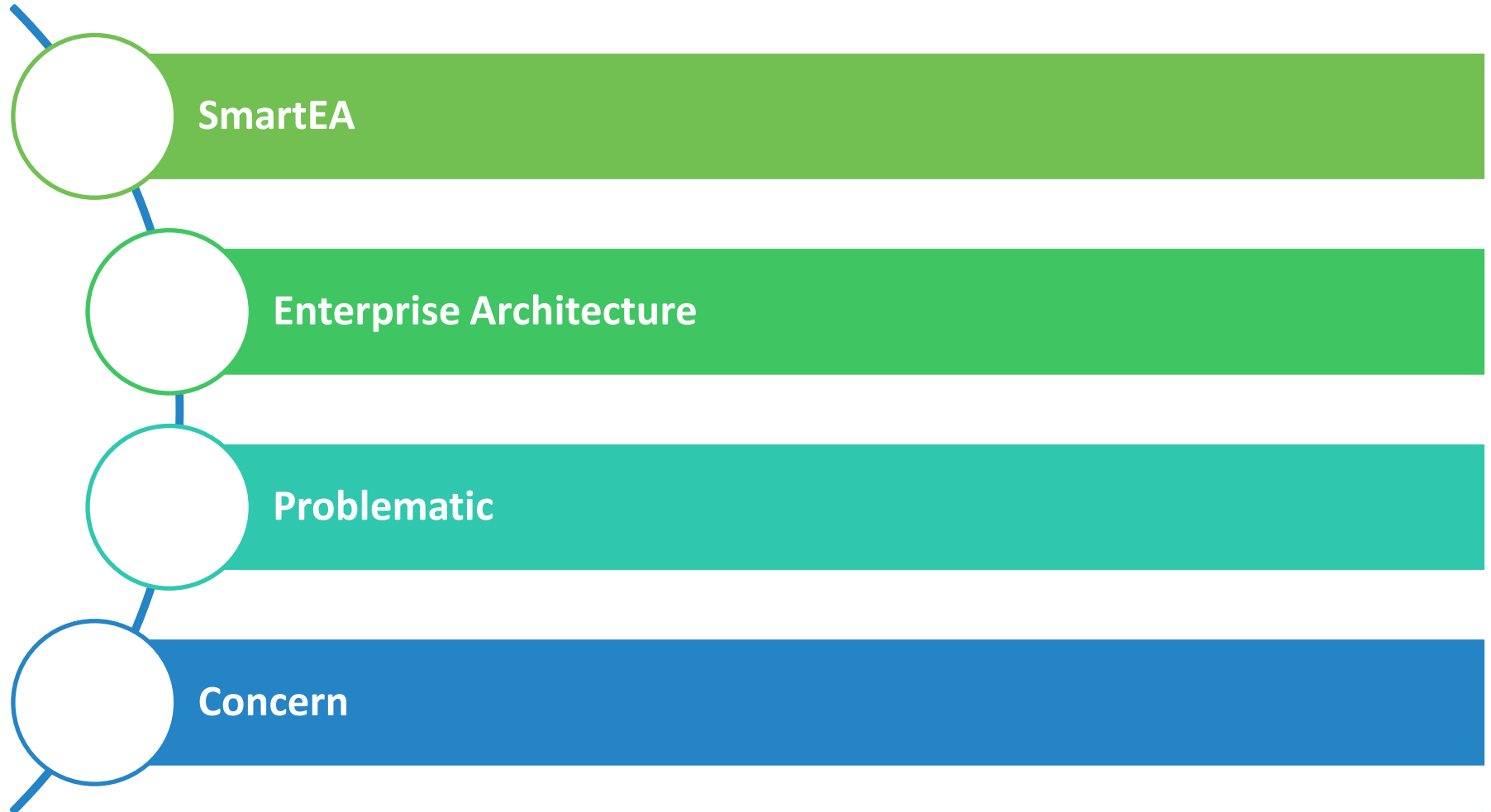- Artifacts knowledge is volatile

Validation

- Release
- Quality

- Mapping tooling
- Live validation feedback
- Fixing assistance
- Batch validation
- Contribute to Intent
- Internal validation of ModelWriter approach

UC-FR-02 - Enterprise Architecture

Yvan Iussaud
OBEO

ITEA3

Enterprise Architecture

SmartEA

Enterprise Architecture

Problematic

Concern

Model the company

- Centralize information
- Ease communication in the company
- View point approach

Support changes in the company

- Business opportunities, business changes, interoperability, new technologies
- Gap analysis
- Impact analysis

Technical information

- Web portal
- Eclipse and Sirius editors
- Internal model can be changed (TOGAF9, BPMN, …)

Define As-Is state

- Import documents
- Consolidate the company model

Define To-Be state

- Identify possible scenario
- Evaluate possible scenario
- Modify the company model

Define trajectories

- Gap analysis
- Impact analysis
- Define milestones
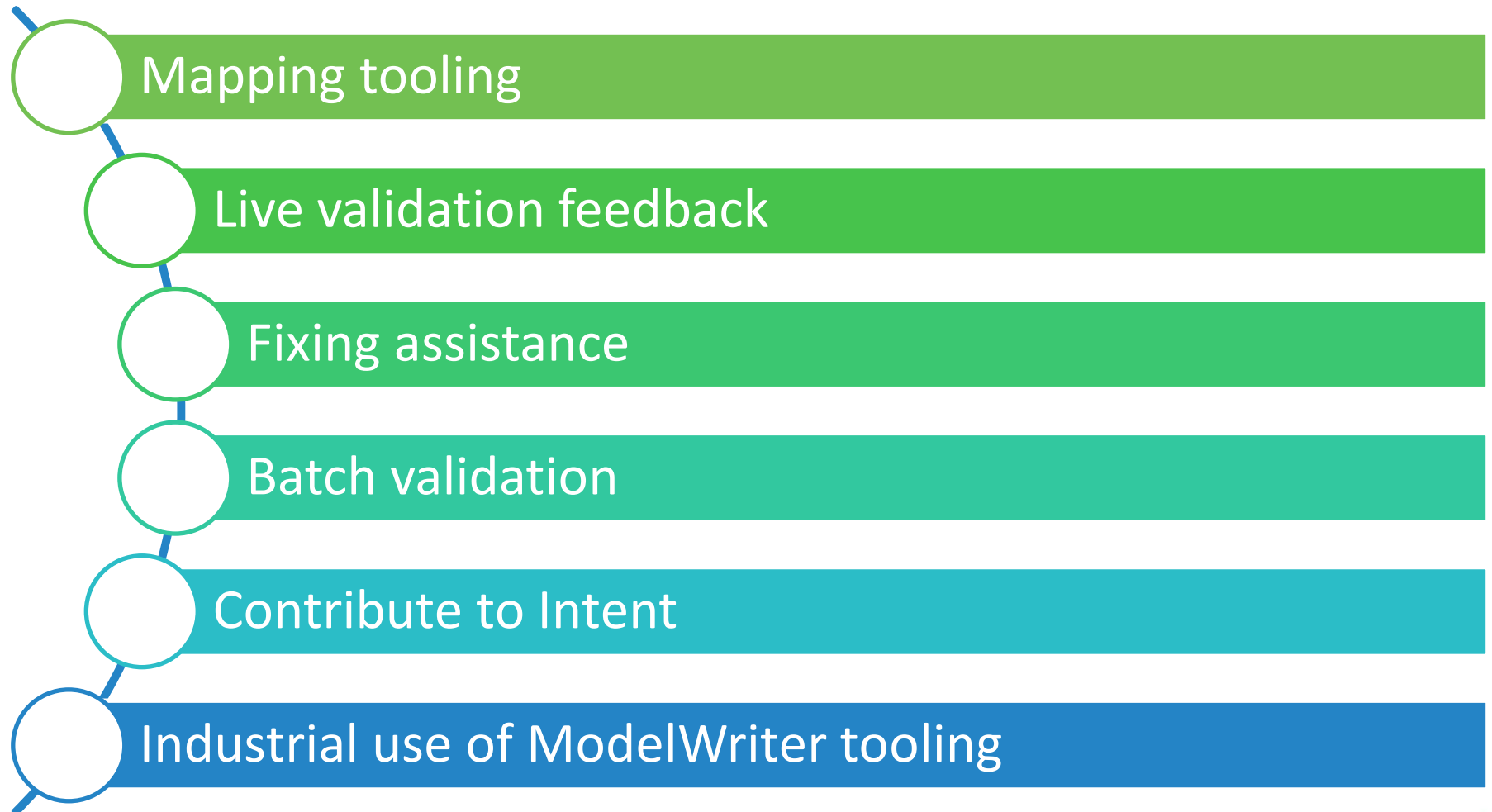
Problematic

Documentation

- Keep track of sourced documents
- Keep representation and portal up-to-date

Synchronization gap

- Organizational changes
- Artifacts knowledge is volatile

Validation

- Produce documentation of organizational changes
- Quality

Concern

- Mapping tooling
- Live validation feedback
- Fixing assistance
- Batch validation
- Contribute to Intent
- Industrial use of ModelWriter tooling

UC-FR-03 - Synchronization of Regulation Documentation with a Design Rule Repository
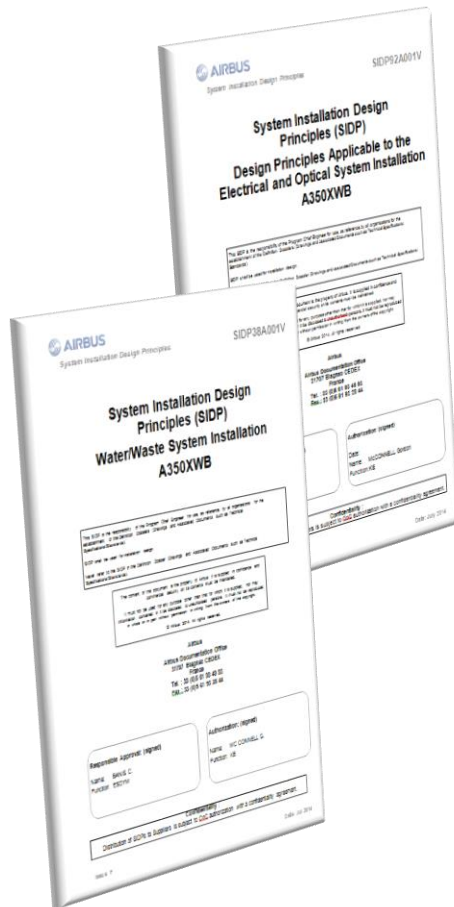
Anne MONCEAUX
Airbus Group Innovation
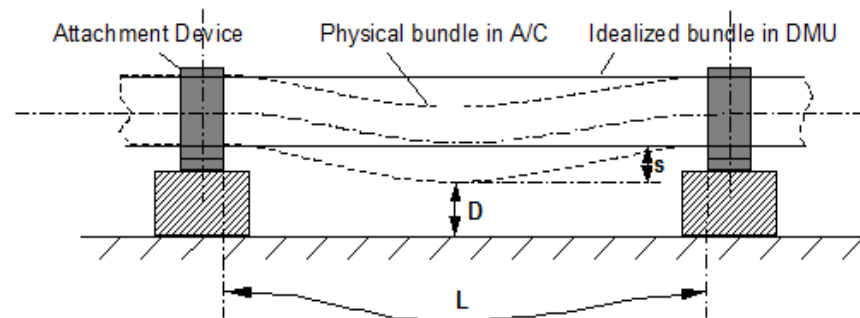*Louis ROUCH, Ayhan MOHOVIC*
Airbus SAS

ITEA3

SIDP: "System Installation Design Principles"

SIDP92A001V-A-784

For installation of optical and electrical harnesses additional clearance for sagging (s) shall be provided as detailed below:

Attachment Device   Physical bundle in A/C   Idealized bundle in DMU

s...Sagging of bundle (real behavior of physical bundle in A/C due to gravity, ageing, etc.)
D...Required Distance
L...Actual length of a bundle segment between two Attachment Points (as designed in DMU)

Figure 6: Sagging of bundles between attachment points

Note: Unless the bundle has a straight routing, L is bigger than the pitch between the Attachment Points.

15

Context and problem

- SIDP documents explain how to install the aircraft systems and attach them to the structure. They capitalize the best practices & proven technical solutions.

- SIDP are defined per ATA chapter (~functional domain) to be applied for each given A/C project: for example "A350 Electrical installation"

- SIPD are open to Extended Enterprise: installation tasks are performed by risk sharing partners.

- SIDP are living documents: during the aircraft development any new DP allowing to satisfy all targets/constraints can be added, assuming it is validated by Airbus dedicated committee.
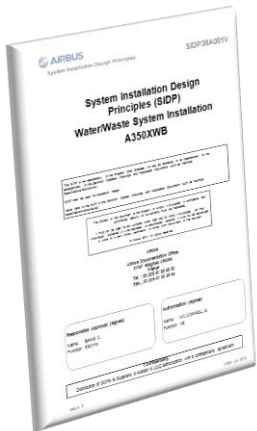
Industrial high level needs

To improve SIDP creation, maintenance and consultation in order to:

- save costs by supporting DP consultation / retrieval in accordance to the needs

- avoid non-compliance of design caused by DP updating lead times

- keep traceability with upstream regulations and requirements and downstream design models

Synchronization of regulation documentation with a design rule repository

ITEA3

UC-FR-03 Synchronization of regulation documentation with a design rule repository

Goal: synchronizing the SIDP database content with documentation

- Create links between text fragments & model elements (manual annotation?, semi-automatic?…)
- Manage consistent synchronization (manage changes)
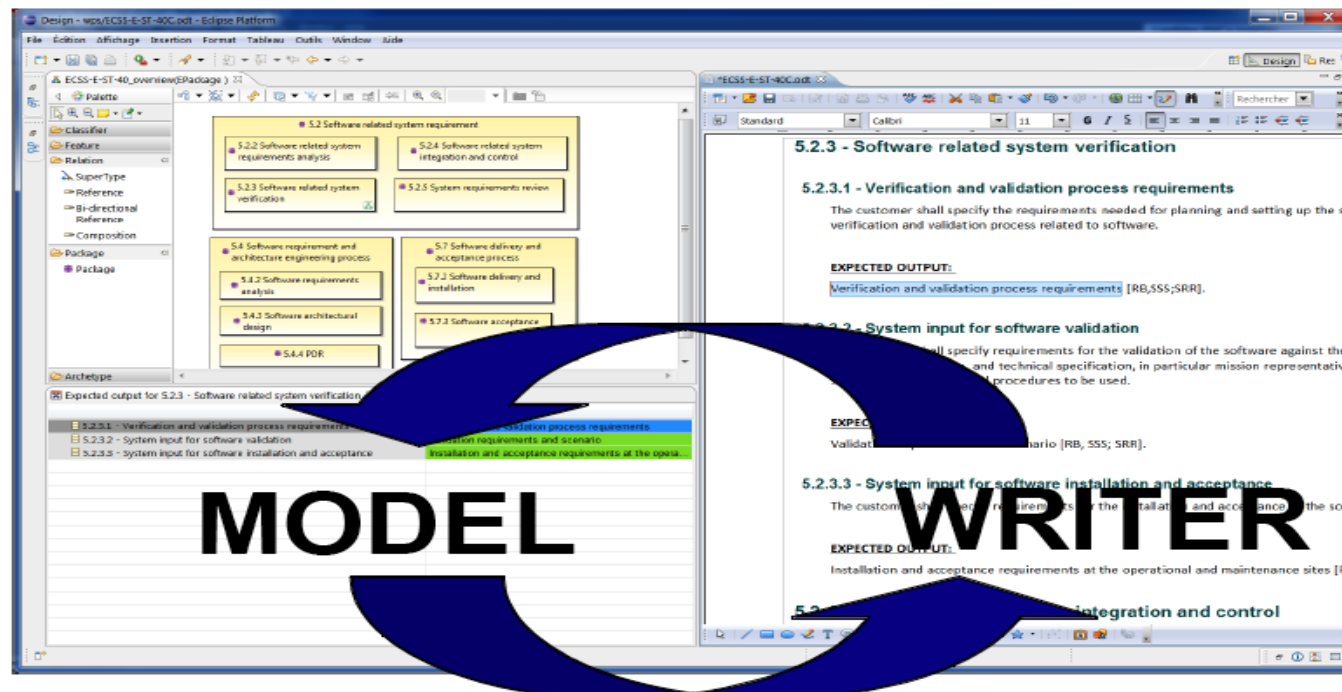
UC-FR-03 Synchronization of regulation documentation with a design rule repository

- Approach: limited case on Electrical Installation functional domain
- (Confidential Data - Non Disclosure Agreement finalized in June 2015).

  - Text
    - 1 document: SIDP ATA 92
    - In our industrial context SIDP are edited using MSWord

  - Models
    - An OWL model is built that reflects the DB schema: "Rule ontology" (30 classes, 35 object properties and 54 data properties)

    - Automatic population mechanism of the model from a csv BD export produces the KB (45781 triples)

UC-FR-03 Synchronization of regulation documentation with a design rule repository

Status 1st iteration: synchronizing the SIDP KB content and text part
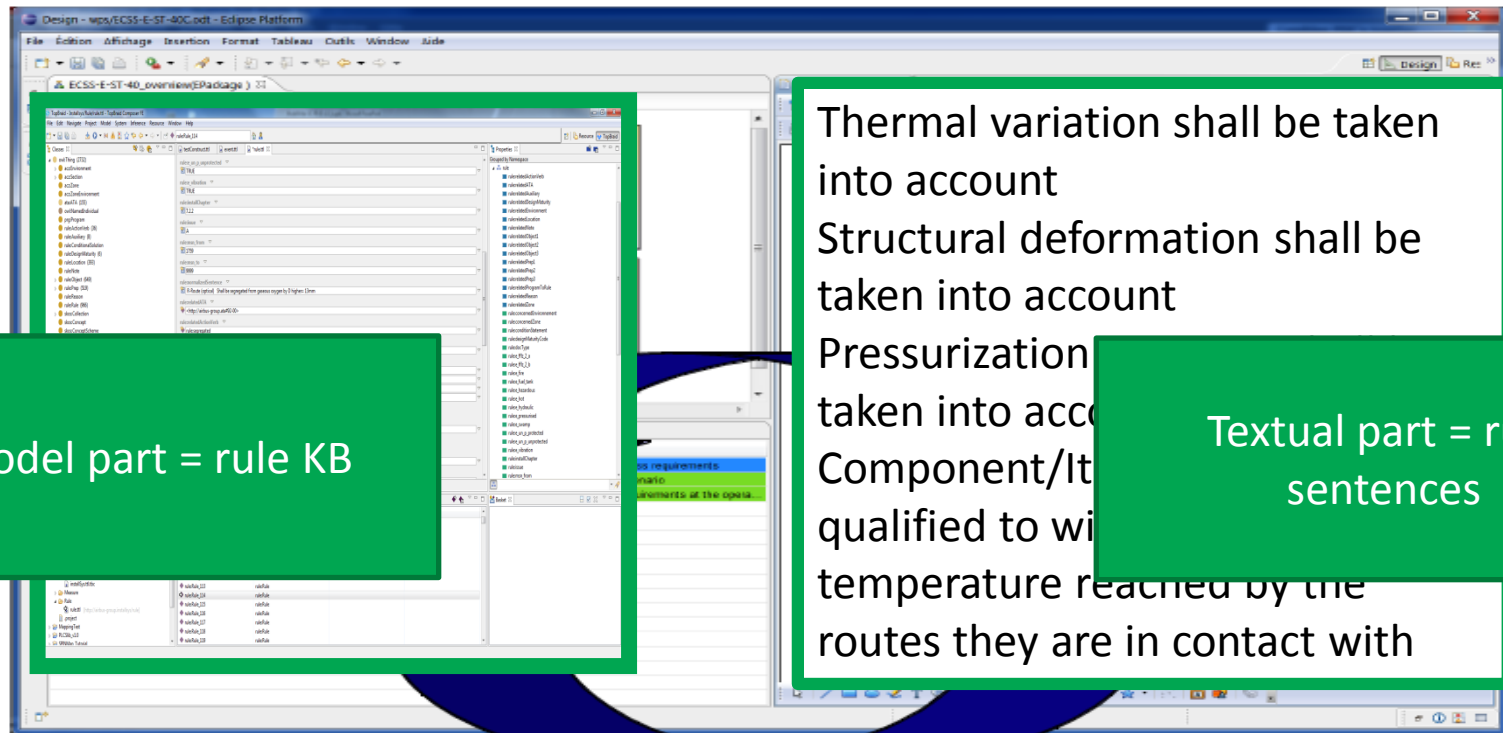
Model part = rule KB

Thermal variation shall be taken into account
Structural deformation shall be taken into account
Pressurization
taken into acc
Component/It
qualified to wi
temperature reached by the
routes they are in contact with

Textual part = rule sentences
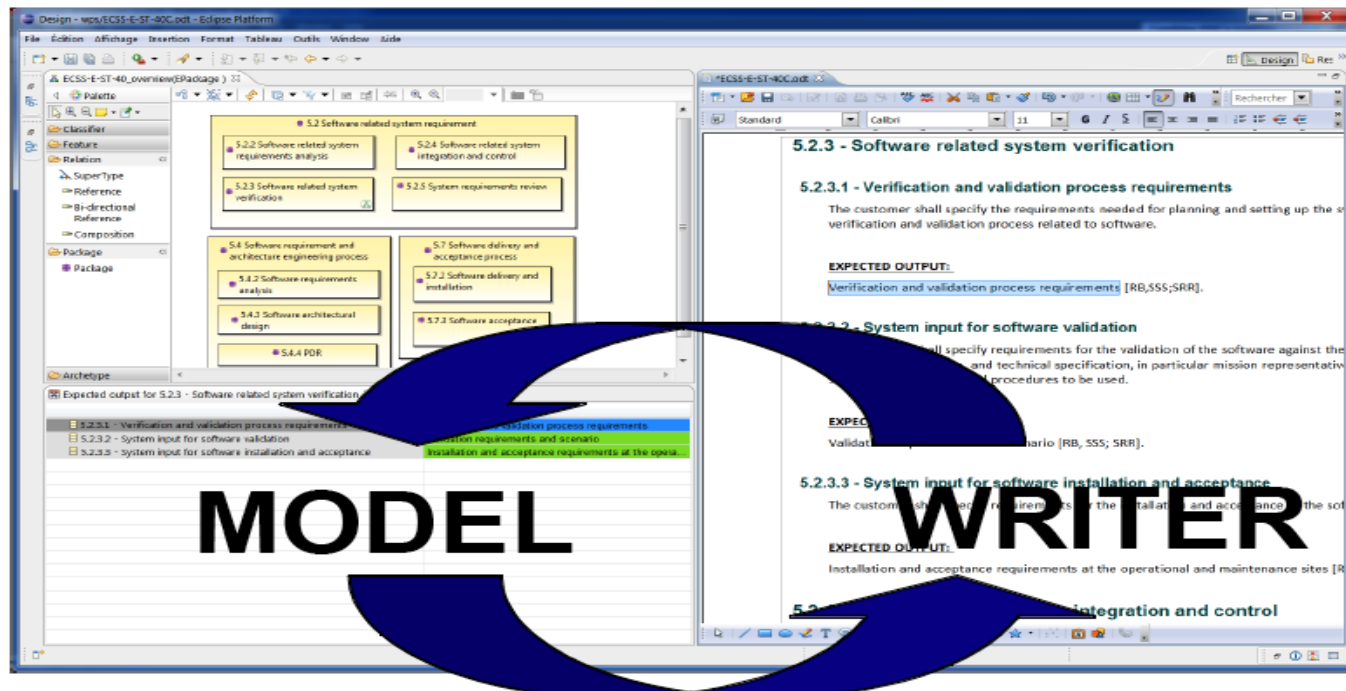
UC-FR-04 - Production of a Context Specific Design Document

Anne MONCEAUX
Airbus Group Innovation
Louis ROUCH, Ayhan MOHOVIC
Airbus SAS

ITEA3

UC-FR-04 Production of a context specific design document

Goal: producing document according to usage "needs"

- Produce "filtered" document with subset of Design Principle textual elements according to usage "needs"

UC-FR-04 Production of a context specific design document

- Approach:
  - use case Electrical Installation functional domain
  - Confidential Data - Non Disclosure Agreement finalized in June 2015
  - Use model elements to retrieve relevant Design Principles

- Text
  - 1 document: SIDP ATA 92
  - In our industrial context SIDP are edited using MSWord

- Models
  - The previous Rule KB (populated Rule ontology)
  - Component ontology (476 classes, 21 ObjectProperties and 35 DataProperties)

Component classes taxonomy

"Structural_Element"

"Bracket"

"Fastener"

"Pclamp"

"Assembly"

ObjectProperties

Component		Assembly

subClassOf

Physical component		Attachment device

Clamp

AnnotationProperties

Rdfs:label

Skos:prefLabel

P-Clamp
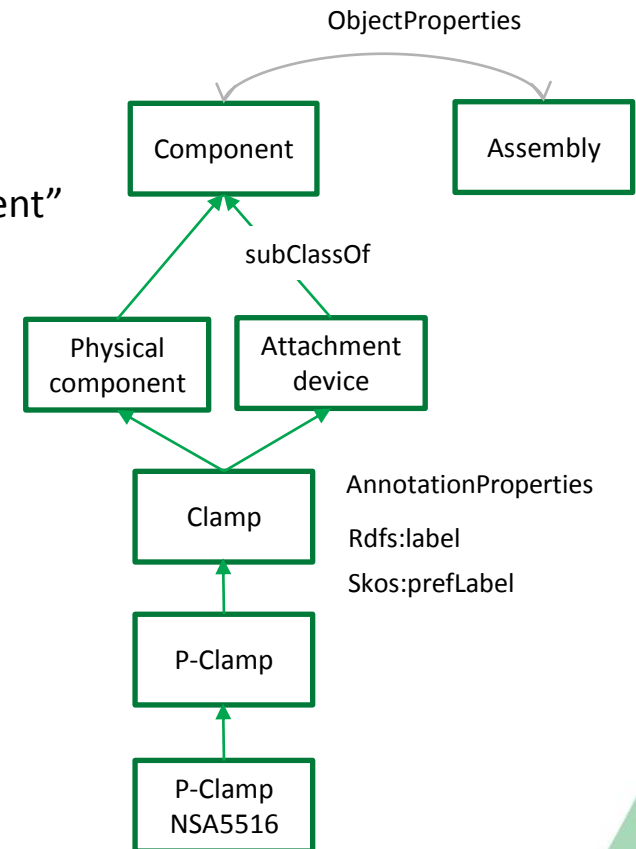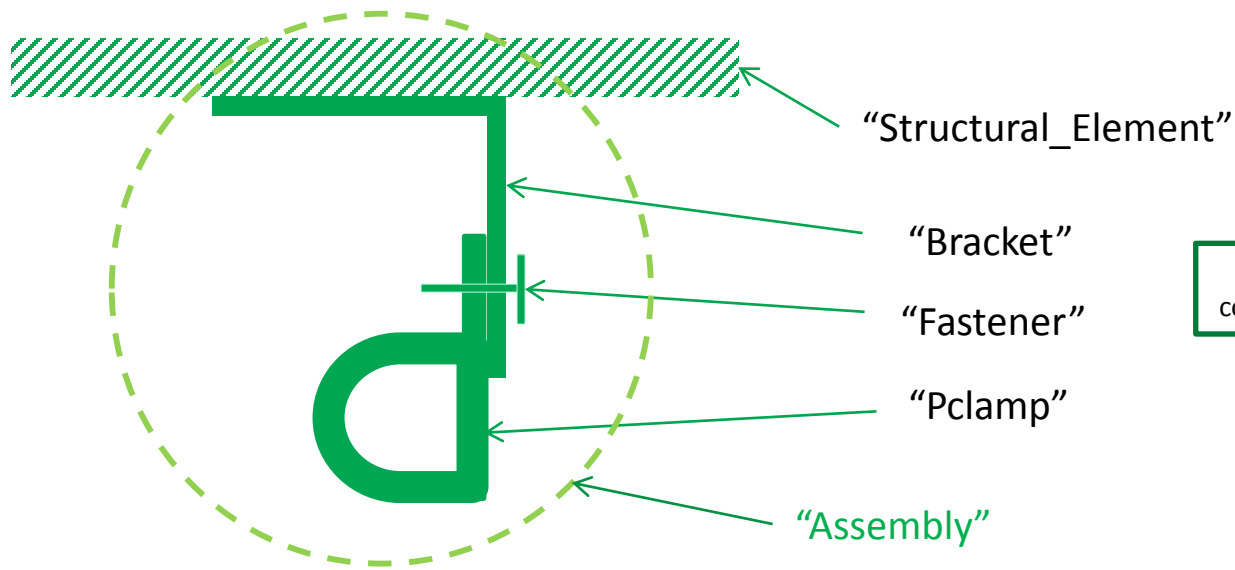
P-Clamp NSA5516

Component classes taxonomy

"P-clamp   NSA5516   can be fixed on X with Y"

"Physical component"   "Standard reference"
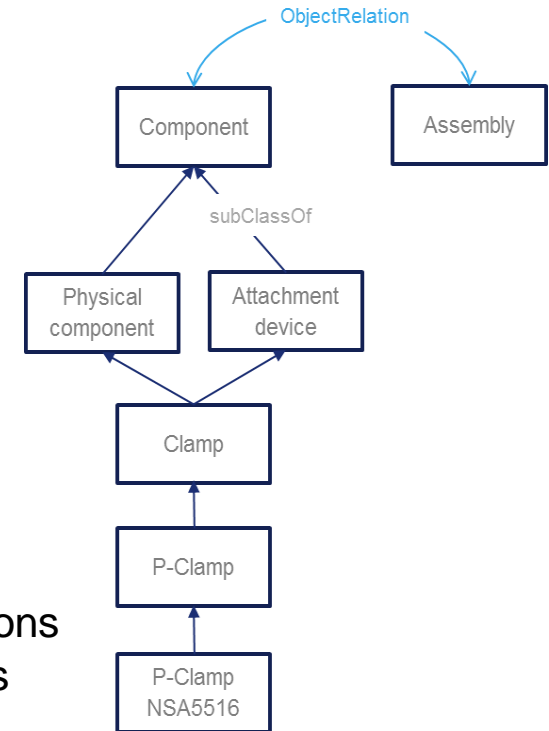
- NLP Parsing uses this taxonomy. Labels + assumptions such as a physical component may be referred to using its name or its reference or both concatenated

- Inference rule: a rule applying to a component type (Attachment device) applies to its subtypes (P-clamp)

- Document display : when searching rules applying to a component type (P-Camp) → retrieve and display rules applying to super-types

SPARQL queries

Status : model based queries specification

- Preliminary study with Loria (Text to RDF)
- 1 internship on RDFizer and Query management
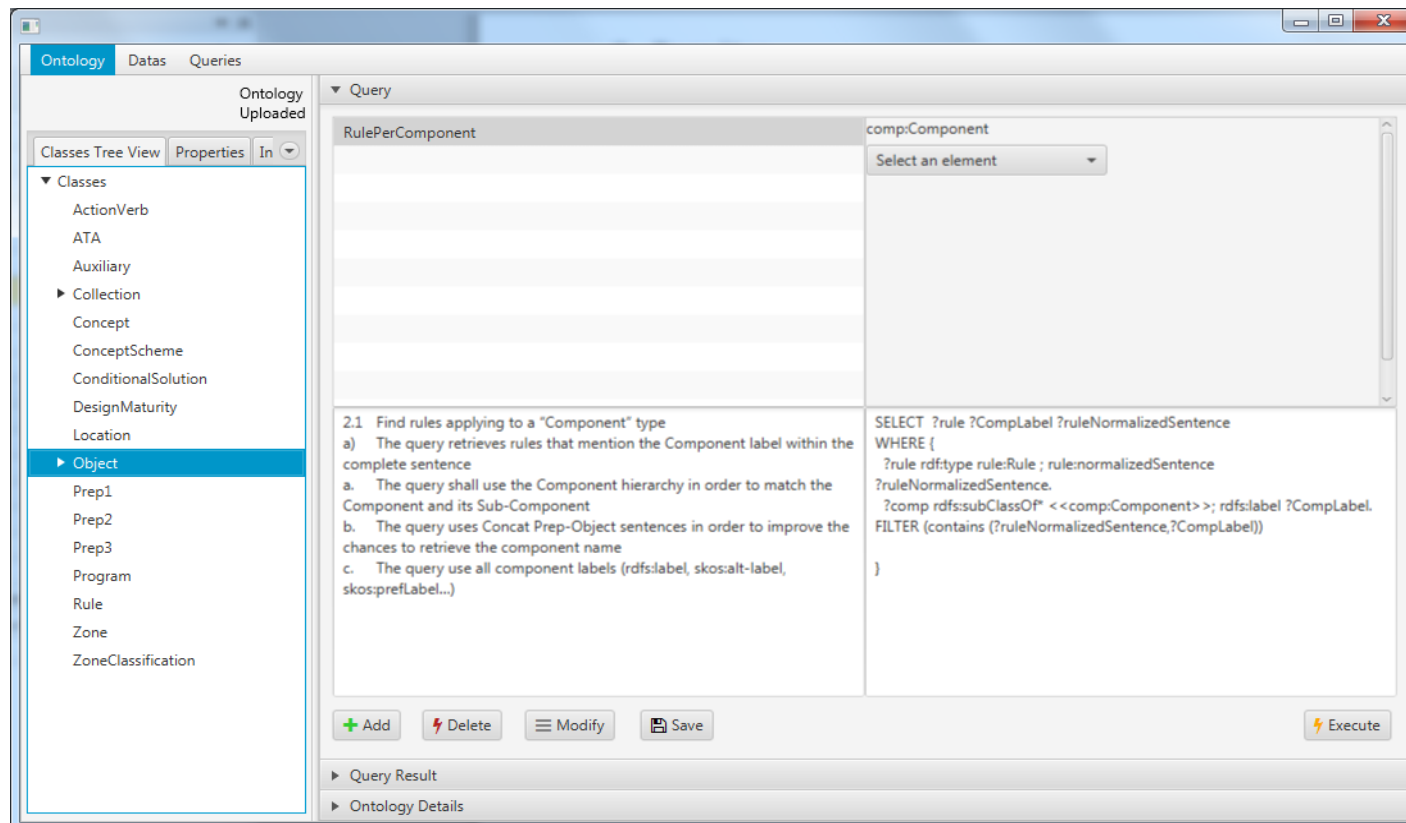
UC-TR-01 - Documents of Quality Assurance Department

Ersan GÜRDOĞAN

HISBIM

UC-TR-01 Documents of Quality Assurance Department

UC-TR-01	Documents of Quality Assurance Department
Version	V1.0.0 dated 15-Nov-2014
Description	To create faster and more accurate the forms that are used in quality control progress and trigger related forms (re-work form triggering, revision needed, approved, rejected etc.)
Actors	Quality Managers, Quality Measurement Specialists, Quality Control Personnel, Quality Auditors, Production Crews
Assumptions	Quality control measurement units are standard Rejected and Approved products forms are standard Quality Certification standards are always applied
Steps	Products information forms are created according to product information comes from ModelWriter system Products are measured by quality control department authorized personnel's according to standards To decide product is standard or not according to measurement report Product would be sent rejected products section if its measures are out of standards ModelWriter system is created Rejected Product Quality Form Else Approved Product Quality Form is created by ModelWriter System Approved products are sent to warehouse as accepted products
Alternatives	If rejected products' measurement out of range in rework standards, it means cannot be applied rework on this product, it returns to scrap. Then it should be sent wasteland.
Issues	Standardizing form designs is hard because of all customers have their own unique reporting tools/formats

UC-TR-02 - Non-Disclosure Agreements

Ersan GÜRDOĞAN
HISBIM

UC-TR-01 Non-Disclosure Agreements

UC-TR-02	Non-Disclosure Agreements
Version	V1.0.0 dated 15-Nov-2014
Description	Non-Disclosure Agreement (NDA), also known as a confidentiality agreement (CA), confidential disclosure agreement (CDA), proprietary information agreement (PIA), or secrecy agreement (SA), is a legal contract between at least two parties that outlines confidential material, knowledge, or information that the parties wish to share with one another for certain purposes, but wish to restrict access to or by third parties. It is a contract through which the parties agree not to disclose information covered by the agreement. An NDA creates a confidential relationship between the parties to protect any type of confidential and proprietary information or trade secrets. As such, an NDA protects nonpublic business information. NDAs are commonly signed when two companies, individuals, or other entities (such as partnerships, societies, etc.) are considering doing business and need to understand the processes used in each other's business for the purpose of evaluating the potential business relationship. NDAs can be "mutual", meaning both parties are restricted in their use of the materials provided, or they can restrict the use of material by a single party. It is also possible for an employee to sign an NDA or NDA-like agreement with an employer. In fact, some employment agreements will include a clause restricting employees' use and dissemination of company-owned confidential information.
Actors	Responsible/Authorized personnel in both parties.
Assumptions	Agreements are prepared and written according to European Business Law.
Steps	To define both partied who would sig the NDA To write items of agreement according to scope of NDA Reviewing agreement by decision maker then getting approval Sharing NDA each other Send feedback to ModelWriter system if any change apply on NDA Sharpen final version of NDA then signing by both parties
Alternatives	After sharing NDA, both parties sign without any change and no feedback. Cancellation of NDA
Issues	Different laws could be applied in out of European Union

UC-TR-03 - Synchronization of ReqIF models from requirement specifications

Ferhat Erata
UNIT

ITEA3

UC-TR-03 - Synchronization of ReqIF models from requirement specifications

- Technical documents are usually long and have complex structure
  - For example requirement or specification documents

- These documents keep changing in the time frame and need to be consistent with the other artifacts
  - For example, with a ReqIF model

- In this use case we aim to keep these documents and models synchronized
  - This will include bidirectional transformation of documents and models

- At the current state of this UC:
  - The transformation is done in one way (left to right)
  - The synchronization is done only between DocModel and ReqIF models

UC-TR-03 - Synchronization of ReqIF models from requirement specifications

Sample Applications:

- Airbus SIDP templates ⇔ ReqIF

- Havelsan Requirement doc templates ⇔ ReqIF

- University Management System docs ⇔ ReqIF

- Eclipse RMF specifications ⇔ ReqIF

- The initial implemented version of this use case will be presented in the demonstration session.

AIRBUS

Design Principles Applicable to the xx System
Installation - Program

SIDPREF1

7 Design Principles

7.1 Design Principles Applicable to the Entire Design

General

SIDP92A001V-A-269
The effects of thermal variations, structural deformation, pressurization variation, etc., shall be taken into account.

SIDP92A001V-A-280
Each item in direct contact with ATA92 bundles shall be qualified to withstand conditions detailed in Table 3 below.

Type of Route	All areas except in high temperature zones		High temperature zones	
	Peak condition *	Continuous operating condition **	Peak condition *	Continuous operating condition **
G routes	170°C	170°C	260°C	260°C
P, X routes	150°C	95°C	260°C	200°C
S, R, T, U, V routes	85°C	85°C	130°C	130°C
Others types (M, S, Q,...)	135°C	95°C	260°C	200°C
* duration of 100 hours				
** duration A/C life				

Table 3. Operating conditions for items in direct contact with ATA92 bundles
Attachment devices placed inside boxes, which contain power cables, shall withstand.

- a minimum of 150°C for peak condition and

- a minimum of 110°C for continuous operating condition.

platform:/resource/Demo/Airbus.docmodel
- Document
  - Paragraph Introduction
    - Paragraph Purpose
    - Paragraph Definitions
    - Paragraph Nomenclature/Abbreviation
    - Paragraph Document Precedence
  - Paragraph Objectives
  - Paragraph Reference Regulations/Documents
    - Paragraph Airworthiness Regulations
      - Paragraph 1
      - Paragraph 2
    - Paragraph Others
      - Paragraph 1
  - Paragraph Responsibilities
    - Paragraph
    - Paragraph
  - Paragraph Structures/Systems Configuration
    - Paragraph
  - Paragraph Application Domain
    - Paragraph
  - Paragraph Design Principles
    - Paragraph Design Principles Applicable to the Entire Design
      - Paragraph General
      - Paragraph SIDP92A001V-A-269
      - Paragraph SIDP92A001V-A-280
      - Paragraph SIDP92A001V-A-356
      - Paragraph SIDP92A001V-A-413
      - Paragraph SIDP92A001V-A-424
      - Paragraph SIDP92A001V-A-3763
      - Paragraph SIDP92A001V-A-472
      - Paragraph Positioning of Bundles in the Aircraft Considering Environmental Constraints
      - Paragraph Segregation or Clearances of Bundles to A/C Structure, other Systems or Betw
    - Paragraph Installation of Connection Elements and Feed Through Seals
    - Paragraph Installation of Electrical Equipment
  - Paragraph Approval Sheet
  - Paragraph Record of Revisions

ID	Description
1	Introduction
1.1	Purpose
1.2	Definitions
1.3	Nomenclature/Abbreviation
1.3.1	A/C Aircraft
1.3.2	ATA Air Transport Association
1.4	Document Precedence
2	Objectives
3	Reference Regulations/Documents
3.1	Airworthiness Regulations
3.1.1	JAR 25.607 Fasteners
3.1.2	REF Title
3.2	Others
3.2.1	A350XWB_SIDP V&V Policy PL0901917
4	Responsibilities
5	Structures/Systems Configuration
6	Application Domain
7	Design Principles
7.1	Design Principles Applicable to the Entire Design
7.1.1	General
7.1.3	SIDP92A001V-A-280
7.1.4	SIDP92A001V-A-356
7.1.4.1	Locking Of Bolted Fastenings
7.1.2	SIDP92A001V-A-269
7.1.5	SIDP92A001V-A-413
7.1.6	SIDP92A001V-A-424
7.1.6.1	Installation In Ceiling Area
7.1.7	SIDP92A001V-A-3763
7.1.7.1	Installation In Fuel Tanks
7.1.8	SIDP92A001V-A-472
7.1.9	Positioning of Bundles in the Aircraft Considering Environmental (
7.1.9.1	General
7.1.9.2	SIDP92A001V-A-557
7.1.9.3	SIDP92A001V-A-579
7.1.10	Segregation or Clearances of Bundles to A/C Structure, other Syste
7.1.10.1	General Applications
7.1.10.1.1	Sagging (s)
7.1.10.1.2	SIDP92A001V-A-784

ITEA3

Screenshot 1 (document):

HAVELSAN OZEL

HAVELSAN YGO PROJESİ
YAZILIM KONFİGÜRASYON
YÖNETİMİ
TEKNİK ŞARTNAMESİ

Doküman No : HVL-YGO-TS-003
Yayın No : 1.0
Yayın Tarihi : Ağustos 2011

HAVELSAN

1. İSTEK VE ÖZELLİKLER

1.1 GENEL ÖZELLİKLER

1.1.1 Yönetim ve Yapılandırma

1.1.1.1 Bütün HAVELSAN Birimlerinin ve Projelerinin merkezi ve tek bir kurulum üzerinde çalışmasına olanak sağlanacaktır.

1.1.1.2 İşletim sistemlerinden bağımsız olarak grafik arayüz ile erişime olanak veren istemci sağlanacaktır.

1.1.1.3 Konfigürasyon yönetim sistemi sunucularına bağlı olmadan çalışılmasına olanak sağlanacaktır.

1.1.1.4 Çoklu kullanıcı desteği sağlanacaktır.

1.1.1.5 İşletim sistemlerinden bağımsız olarak ve en az görüntüleme amaçlı örün (Web) tabanlı çalışılmasına olanak sağlanacaktır.

1.1.1.6 Yeni kullanıcı tanımlanmasına, var olan kullanıcıların güncellenmesine ve silinmesine olanak sağlanacaktır.

1.1.1.7 Kullanıcıların gruplara atanmasına ve gruplardan çıkartılmasına olanak sağlanacaktır.

1.1.1.8 Kullanıcı profiline ve proje yapısına göre var olan deponun genişletilmesine, yeni depo tanımlanmasına olanak sağlanacaktır.

1.1.2 Yetkilendirme ve Güvenlik

1.1.2.1 Kendi veritabanındaki kullanıcı bilgilerini kullanarak kullanıcı kimlik denetimi yapabilecektir.

1.1.2.2 Aktif Dizin'de (Active Directory) tanımlı kullanıcı bilgilerini kullanarak kullanıcı kimlik denetimi yapabilecektir.

Screenshot 2 (doc model tree):

platform:/resource/Demo/Havelsan.docmodel
- Document
  - Paragraph İSTEK VE ÖZELLİKLER
    - Paragraph GENEL ÖZELLİKLER
      - Paragraph Yönetim ve Yapılandırma
        - Paragraph Bütün HAVELSAN Birimlerinin ve Projelerinin merkezi ve tek bir kuru...
        - Paragraph İşletim sistemlerinden bağımsız olarak grafik arayüz ile erişime olana...
        - Paragraph Konfigürasyon yönetim sistemi sunucularına bağlı olmadan çalışılm...
        - Paragraph Çoklu kullanıcı desteği sağlanacaktır.
        - Paragraph İşletim sistemlerinden bağımsız olarak ve en az görüntüleme amaçlı...
        - Paragraph Yeni kullanıcı tanımlanmasına, var olan kullanıcıların güncellenmesi...
        - Paragraph Kullanıcıların gruplara atanmasına ve gruplardan çıkartılmasına olan...
        - Paragraph Kullanıcı profiline ve proje yapısına göre var olan deponun genişletil...
      - Paragraph Yetkilendirme ve Güvenlik
      - Paragraph İşlevsellik
      - Paragraph İzleme ve Rapor
      - Paragraph Entegrasyon
        - Paragraph Dış sistemlerle tümleştirme için Uygumala Geliştirme Arayüzü sağlar...
        - Paragraph Depodaki bir konfigürasyon elemanından başlanarak bu öğenin altın...
        - Paragraph Dosya sistemindeki bir dizin ve altının özyineli olarak içeri alınmasın...
        - Paragraph LDAP veya Aktif Dizin bağlantısı için gerekli olan parametrelerin giril...
      - Paragraph Ürün Kılavuzları
        - Paragraph Satıcı/Yüklenici ürünle ilgili Tablo 3'de listelenen kılavuzları geçici ka...
        - Paragraph Kullanıcı kılavuzları Türkçe veya İngilizce olarak sağlanacaktır.
        - Paragraph Ürün için çevrimiçi yardım ekranları sağlanacaktır.
      - Paragraph Eğitimler
        - Paragraph Sistemlerin kullanıma alınması ve yaygınlaştırılması için Tablo 4'de li...
        - Paragraph Eğitim süreleri yüklenici önerisi ve HAVELSAN onayı ile değiştirilebil...
        - Paragraph Satıcı/Yüklenici tarafından katılımcı sayısı kadar basılı eğitim mater...
        - Paragraph Eğitim yeri HAVELSAN/ANKARA tesisleridir.
        - Paragraph Satıcı/Yüklenici eğitimden bir hafta önce eğitimci özgeçmişini, eğiti...

Screenshot 3 (ID / Description table):

ID	Description
1	İSTEK VE ÖZELLİKLER
1.1	İSTEK VE ÖZELLİKLER
1.1	GENEL ÖZELLİKLER
1.1.1	Yönetim ve Yapılandırma
1.1.1.1	Bütün HAVELSAN Birimlerinin ve Projelerinin merkezi ve tek bir kurulum üzerinde çalış
1.1.1.2	İşletim sistemlerinden bağımsız olarak grafik arayüz ile erişime olanak veren istemci sa
1.1.1.3	Konfigürasyon yönetim sistemi sunucularına bağlı olmadan çalışılmasına olanak sağla
1.1.1.4	Çoklu kullanıcı desteği sağlanacaktır.
1.1.1.5	İşletim sistemlerinden bağımsız olarak ve en az görüntüleme amaçlı örün (Web) taban
1.1.1.6	Yeni kullanıcı tanımlanmasına, var olan kullanıcıların güncellenmesine ve silinmesine
1.1.1.7	Kullanıcıların gruplara atanmasına ve gruplardan çıkartılmasına olanak sağlanacaktır.
1.1.1.8	Kullanıcı profiline ve proje yapısına göre var olan deponun genişletilmesine, yeni depo
1.1.2	Yetkilendirme ve Güvenlik
1.1.2.1	Kendi veritabanındaki kullanıcı bilgilerini kullanarak kullanıcı kimlik denetimi yapabile
1.1.2.2	Aktif Dizin'de (Active Directory) tanımlı kullanıcı bilgilerini kullanarak kullanıcı kimlik d
1.1.2.3	Aktif Dizin'den elde edilmiş kullanıcı kimliğini, Tek Giriş (Single Sign On - SSO) ilkesine
1.1.2.4	Kimliği doğrulanmış kullanıcılar için yetkilendirme yapılacaktır.
1.1.2.5	Nesne erişimlerinin yetkilendirilmesi için, en az "Ekleme", "Silme", "Düzenleme", "Görü
1.1.2.6	Proje yönetim ve bakım işlevlerinin yetkilendirilmesi için "Yönetim" yetkisi sağlanacak
1.1.2.7	Proje yönetim ve bakım işlevleri ile nesneler üzerinde yapılabilen tüm iş ve işlemleri ya
1.1.2.8	Nesneler üzerinde yapılabilen tüm iş ve işlemleri yapma yetkisine sahip "Düzenleyici" y
1.1.2.9	Nesneleri görüntüleme ve raporlama yetkisine sahip "Görüntüleyici" yetki profili sağla
1.1.2.10	Yetki profillerinin kullanıcılara/gruplara atanmasına olanak sağlanacaktır.
1.1.2.11	Kullanıcıların/grupların dosya/dizinler üzerinde seçilebilecek yetkiler ile yetkilendirilme
1.1.3	İşlevsellik
1.1.3.1	Konfigürasyon elemanlarının çalışma kopyasının yaratılmasına olanak sağlanacaktır.
1.1.3.2	Çalışma kopyasında değişiklik yapılan konfigürasyon elemanlarının depoya gönderilm
1.1.3.3	Konfigürasyon elemanı üzerinde yapılan değişikliğin depoya gönderilmesi sırasında yo
1.1.3.4	Komut satırından çalışma olanağı sağlanacaktır.
1.1.3.5	Depoya gönderme işleminin parçalanamaz şekilde (atomic commit) gerçekleşmesi sa
1.1.3.6	Yan dal (branch) açmaya olanak sağlanacaktır.

Left Panel

1 USE CASE UC1: SIGN IN

Primary Actor: Student, Lecturer.

Stakeholders and Interests:

- Student: Wants simple user interface, fast response, no system errors.
- Lecturer: ?

Preconditions:

- Student is registered.

Success Guarantee (Postcondition):

- Student is logged in.

Main Success Scenario (or Basic Flow):

1. Student visits system home page.
2. System shows home page with login form and sign up button.
3. Student enters his/her username and password then click login button.
4. System shows Student's home page.

Extensions (or Alternative Flows):

*a. At any time, system fails, to support recovery, ensure all transaction sensitive state
be recovered from any step of the scenario.
   1. Student restarts System and requests recovery of prior state.
   2. System reconstructs prior state.
      2a. System detects anomalies preventing recovery.
         1. System signals error to the Student, records the error, and
state.
3a. Student enters invalid username or password.
   1. System shows errors and request to Student to retry.
   2. Student enters his/her username and password.
4a. System detects failure to communicate with server.
   1. System signals error and rejects the request.

Middle Panel

- platform:/resource/Demo/UniversityManagementSystem.docmodel
  - Document
    - Paragraph Use Case UC1: Sign In
      - Paragraph Primary Actor
        - Part Student
        - Part Lecturer.
      - Paragraph Stakeholders and Interests
        - Paragraph 1
          - Paragraph Student
        - Paragraph 2
          - Paragraph Lecturer
      - Paragraph Preconditions
      - Paragraph Success Guarantee (Postcondition)
      - Paragraph Main Success Scenario (or Basic Flow)
        - Paragraph 1
        - Paragraph 2
        - Paragraph 3
        - Paragraph 4
      - Paragraph Extensions (or Alternative Flows)
        - Paragraph *a
        - Paragraph 3a
        - Paragraph 4a
        - Paragraph
    - Paragraph Use Case UC2: File Download
      - Paragraph Primary Actor
      - Paragraph Stakeholders and Interests
      - Paragraph Preconditions
      - Paragraph Success Guarantee (Postcondition)
      - Paragraph Main Success Scenario (or Basic Flow)
      - Paragraph Extensions (or Alternative Flows)
    - Paragraph Use Case UC3: File Upload
    - Paragraph Use Case UC4: Course Analysis Report
    - Paragraph Use Case UC5: Adding Lecture
    - Paragraph Use Case UC6: Deleting Lecture
    - Paragraph Use Case UC7: Lecture Listing
    - Paragraph Use Case UC8: Enrolling to a Lecture
    - Paragraph Use Case UC9: Unenrolling a Lecture
    - Paragraph Use Case UC10: New Forum Post
    - Paragraph Use Case UC11: Reply Forum Post

Right Panel

ID	Description
1	Use Case UC1: Sign In
1.1	Primary Actor
1.1.1	Student
1.1.2	Lecturer.
1.2	Stakeholders and Interests
1.2.1	Student
1.2.1.1	Wants simple user interface, fast response, no system errors.
1.2.1	Lecturer
1.2.1.1	?
1.3	Preconditions
1.3.1	Student is registered.
1.4	Success Guarantee (Postcondition)
1.4.1	Student is logged in.
1.5	Main Success Scenario (or Basic Flow)
1.5.1	Student visits system home page
1.5.2	System shows home page with login form and sign up button
1.5.3	Student enters his/her username and password then click login b
1.5.4	System shows Student's home page
1.6	Extensions (or Alternative Flows)
1.6.1	At any time, system fails, to support recovery, ensure all transacti state and events can be recovered from any step of the scenario
1.6.1.1	Student restarts System and requests recovery of prior state
1.6.1.2	System reconstructs prior state
1.6.1.2.1	System detects anomalies preventing recovery
1.6.1.2.1.1	System signals error to the Student, records the error, and enters a
1.6.2	Student enters invalid username or password
1.6.2.1	System shows errors and request to Student to retry
1.6.2.2	Student enters his/her username and password
1.6.3	System detects failure to communicate with server
1.6.3.1	System signals error and rejects the request
2	Use Case UC2: File Download
2.1	Primary Actor
2.1.1	Student
2.2	Stakeholders and Interests
2.2.1	Student

Left panel

1 USE CASE UC1: CREATE A NEW SPECOBJECT

Preconditions:

- ReqIF model exists and is open.

Main Success Scenario (or Basic Flow):

1. We assume that a Specification exists and is open (not required for alternative scenario)
2. Double click on the cell in the Specification Editor to be edited.
3. Select the Child or Sibling submenu.
4. Select the desired Spec Object Type (or none) from the submenu.
5. Note that some cells may not be editable, in which case nothing will happen.

Alternative 1 Create in Outline:

*a. The same workflow works for elements that are shown underneath "Specifications" in th
2a. It is also possible to create children of the "SpecObjects" folder in the outline, but in this SpecHirarchy will be created.

Alternative 2 Keyboard Shortcut:

*a. The keyboard shortcut Ctrl-Enter will create a SpecHierarchy sibling to the currently sele element and immediately go into edit mode in the currently selected column.

2 USE CASE UC2: EDIT SPECOBJECT

Preconditions:

- A ReqIF model exists, is open and at least one SpecObject exisits.

Main Success Scenario (or Basic Flow):

1. We assume that a Specification exists and is open (not required for alternative scenario)
2. Open a row's context menu (or in the empty editor space)

Middle panel

- platform:/resource/Demo/CustomerRequirementSpecification.docmodel
  - Document
    - Paragraph Use Case UC1: Create a new SpecObject
      - Paragraph Preconditions
        - Paragraph 1
      - Paragraph Main Success Scenario (or Basic Flow)
        - Paragraph 1
        - Paragraph 2
        - Paragraph 3
        - Paragraph 4
        - Paragraph 5
      - Paragraph Alternative 1 Create in Outline
        - Paragraph *a
        - Paragraph 2a
      - Paragraph Alternative 2 Keyboard Shortcut
        - Paragraph *a
    - Paragraph Use Case UC2: Edit SpecObject
      - Paragraph Preconditions
        - Paragraph 1
      - Paragraph Main Success Scenario (or Basic Flow)
        - Paragraph 1
        - Paragraph 2
        - Paragraph 3
        - Paragraph 4
        - Paragraph 5
      - Paragraph Alternative 1 Edit in Properties View
        - Paragraph *a
    - Paragraph Use Case UC3: Delete SpecObject
      - Paragraph Preconditions
        - Paragraph 1
      - Paragraph Main Success Scenario (or Basic Flow)
        - Paragraph 1
      - Paragraph Alternative 1 Delete in SpecObjects folder in Outline
        - Paragraph *a

Right panel

ID	Description
1	Use Case UC1: Create a new SpecObject
1.1	Preconditions
1.1.1	ReqIF model exists and is open.
1.2	Main Success Scenario (or Basic Flow)
1.2.1	We assume that a Specification exists and is open (not required for alternative scenario)
1.2.2	Double click on the cell in the Specification Editor to be edited
1.2.3	Select the Child or Sibling submenu
1.2.4	Select the desired Spec Object Type (or none) from the submenu
1.2.5	Note that some cells may not be editable, in which case nothing will happen
1.3	Alternative 1 Create in Outline
1.3.1	The same workflow works for elements that are shown underneath "Specifications" in the outilne
1.3.2	It is also possible to create children of the "SpecObjects" folder in the outline, but in this case, no SpecH
1.4	Alternative 2 Keyboard Shortcut
1.4.1	The keyboard shortcut Ctrl-Enter will create a SpecHierarchy sibling to the currently selected element an
2	Use Case UC2: Edit SpecObject
2.1	Preconditions
2.1.1	A ReqIF model exists, is open and at least one SpecObject exisits.
2.2	Main Success Scenario (or Basic Flow)
2.2.1	We assume that a Specification exists and is open (not required for alternative scenario)
2.2.2	Open a row's context menu (or in the empty editor space)
2.2.3	Alternatively, hit enter or space in that cell
2.2.4	In both cases, the double-clicked / selected cell will switch to edit mode
2.2.5	This results in a new SpecHierarchy being created that is linked to a newly created SpecObject with the
2.3	Alternative 1 Edit in Properties View
2.3.1	A selected element (no matter whether in Specification Editor or Outline or elsewhere) will be shown in
3	Use Case UC3: Delete SpecObject
3.1	Preconditions
3.1.1	A ReqIF model exists, is open and at least one SpecObject exists.
3.2	Main Success Scenario (or Basic Flow)
3.2.1	If an element is deleted from the specification (so essentially a SpecHierarchy), and no references to the
3.3	Alternative 1 Delete in SpecObjects folder in Outline
3.3.1	If the SpecObject is deleted from the SpecObjects folder in the ouline, it will be removed, no matter wha

- Next steps:

    - ReqIF model transformation to documents (bi-directional transformation)
    - Synchronization between a document and Doc-Model (fully synchronization)

UC-TR-04 - Integration with Application Lifecycle Management (ALM) Tools

Eray Tüzün
Yagup MACİT

HAVELSAN

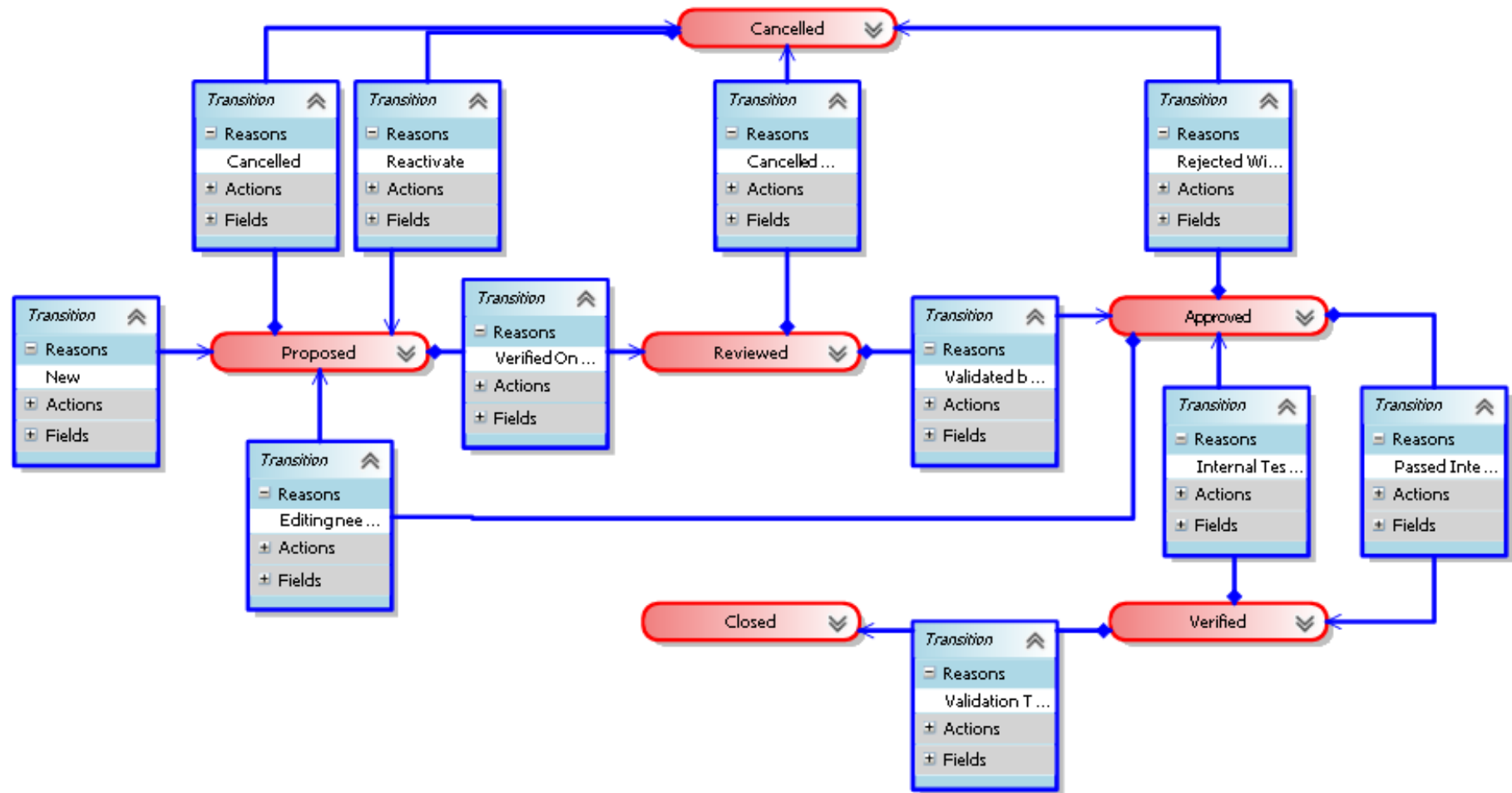# UC-TR-04 - Integration with Application Lifecycle Management (ALM) Tools

UC-TR-03	Integration with ALM tools
Versioning Info	V1.0.0 dated 28-Apr-2015
Description	Show that the ModelWriter can extract required elements from structured requirement objects in ALM tool to generate automatic design model. Show that the ModelWriter's can generate requirements specification document from structured requirement objects in ALM tool.
Actors	Requirements Engineer/Manager, System/Software Architect
Assumptions	• All the traceability information would continue to be followed from the structured requirement object via ALM tool. (ModelWriter does not need to have any traceability information) • Textual representation of requirements are stored in MS Word documents. • Design models are stored in Sparx Systems Enterprise Architect. • Structured requirement objects are stored in Microsoft Team Foundation Server.

Steps	Scenario1:
	
	    o From a set of structured requirement objects in the ALM tool, ModelWriter would create a natural-language text requirement document.
	
	Scenario2:
	
	    o From a set of structured requirement objects in the ALM tool, ModelWriter would create/synch a design model.
	
	    o From a design model, ModelWriter would create/synch to a set of structured requirement objects.

Variations (optional)	Scenario1: From a natural-language text requirement document, ModelWriter would create/synch to a set of requirement objects.
Non-functional (optional)	The system should have a Word plugin for natural-language text transformation. The natural-language support should be in both English and Turkish
Issues	A common format between ALM platforms and ModelWriter may need to be implemented.

Requirement Work Item

Customized Form

Attributes

WorkItem Number Patterns

History

Discussion

Requirement LifeCycle

Modeling Artifacts in ALM

Requirements in ALM

- Traceability with other artifacts is key
  - Requirements to other requirements
    - Customer/System/Software/Hardware..
    - Dependency relation between requirements
  - Requirements to tasks (Project management)
  - Requirements to Test Cases
  - Requirements to Design elements
  - Requirements to generated documents
  - Requirements to source code
  - Requirements to Build
  - Requirements to bugs
  - Requirements to risks
  - …

23.9.2015

Havelsan Ext - Document Generation

Preview

Word Extension

Support for Templates

23.9.2015
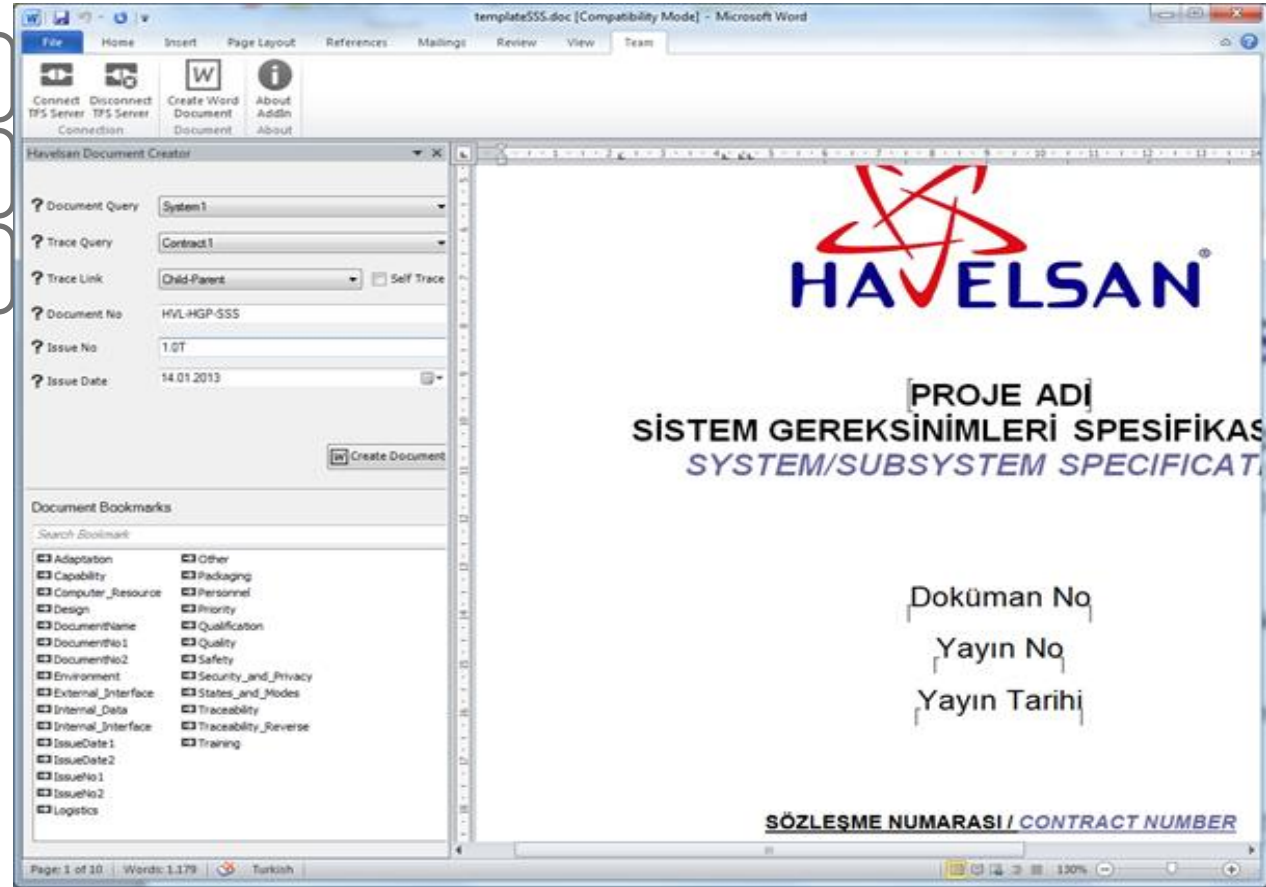
Requirements - Design Model Traceability

- Requirements would be imported to the Design tool for traceability.
- Code generation from design
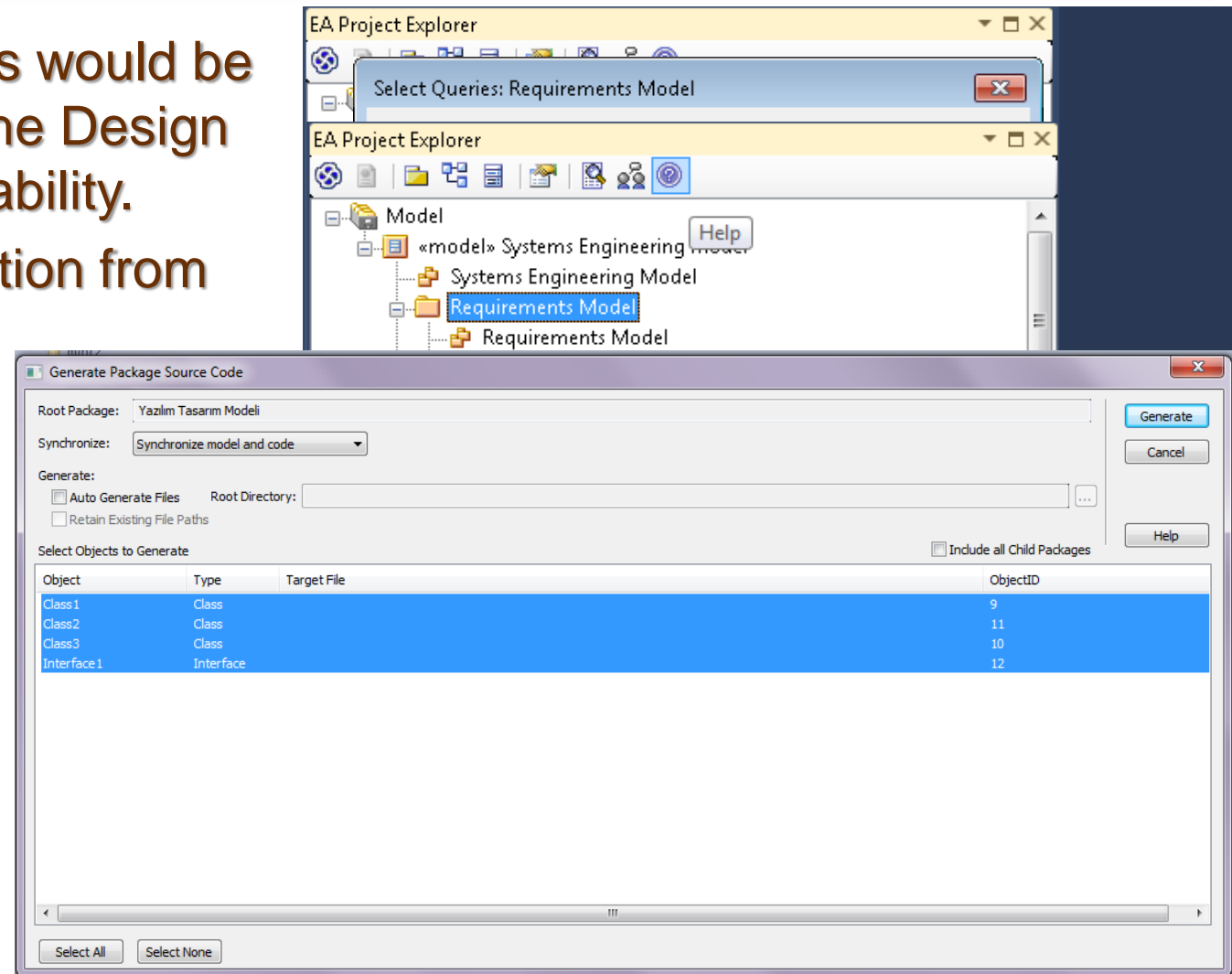
HAVELSAN Use case for ModelWriter

Requirement Document

Model

1

Requirement Document

2

Structured Requirement Object

3

Model

4
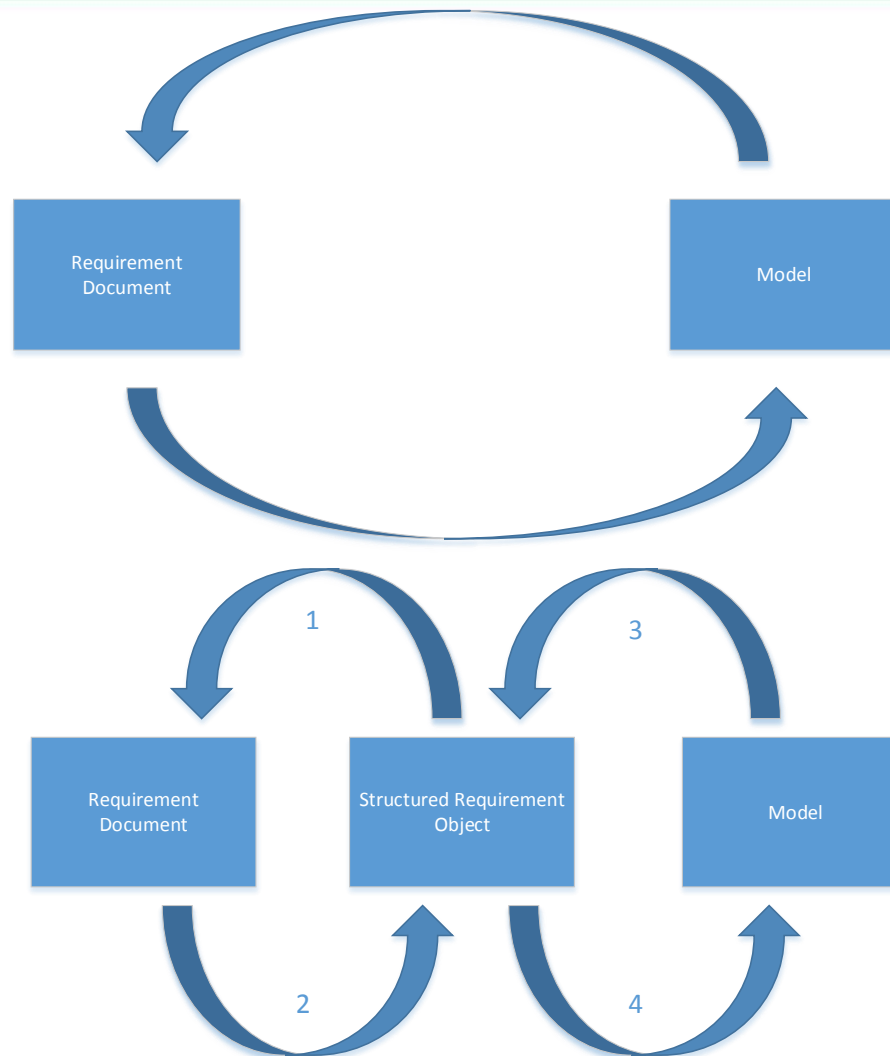
Currently we support scenario #1 and #4, and interested in Scenario #3, #2

23.9.2015

UC-TR-05 - Synchronous Business Process Design with Use Cases

Geylani Kardaş

KoçSistem

- Use cases are one of the main approaches to represent the requirements.
  - A use case is a list of actions or event steps, typically defining the interactions between a role (a.k.a actor in the UML).

- BPMN provides a graphical notation for specifying the processes in a diagram based on a flowcharting technique (similar to activity diagrams in UML).
  - The aim is to support business process management, for both technical and business users, by providing a notation which enables to represent complex process semantics.

- The transformation between use cases and BPMN models, and synchronization of them are addressed in this UC.
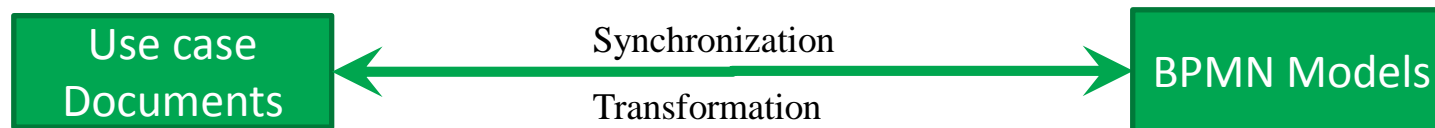
- Applications:
  - University Management System Use cases docs ⇔ BPMN
  - Eclipse RMF use case specifications ⇔ BPMN

- At the current state of this UC:
  - The transformations are done in one way (left to right)

3 USE CASE UC3: FILE UPLOAD

Primary Actor: Student

Stakeholders and Interests:

- Student: Wants simple user interface, fast response, no system errors.

Preconditions: Student is identified and authenticated.

Success Guarantee (Postcondition): File is uploaded.

Main Success Scenario (or Basic Flow):

1. Student visits file upload page.
2. System opens file browser dialog.
3. Student chooses the file that she/he is wanted to upload.
4. System starts the upload process.

Extensions (or Alternative Flows):

*a. At any time system fails, to support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

    1. Student restarts System and requests recovery of prior state.

    2. System reconstructs prior state.

2a. System detects anomalies preventing recovery.

    1. System signals error to the Student, records the error, and enters a clean state.

3a. Invalid file.

    1. System shows the error and returns the file upload page.

- ▾ ☒ platform:/resource/Demo/TestDocument.xmi
  - ▾ ◆ Specification UseCaseSpecification
    - ▾ ◇ Use Case Use Case UC1: Create a new SpecObject
      - › ◆ Process Main Success Scenario (or Basic Flow)
      - ▾ ◆ Process Preconditions
        - › ◆ Start Event
    - ▾ ◇ Use Case Use Case UC2: Edit SpecObject
      - ▾ ◆ Process Main Success Scenario (or Basic Flow)
        - ◆ Start Event
        - ◆ Sequence Flow
        - › ◆ Activity 1
        - ◆ Sequence Flow
        - › ◆ Activity 2
        - ◆ Sequence Flow
        - › ◆ Activity 3
        - ◆ Sequence Flow
        - › ◆ Activity 4
        - ◆ Sequence Flow
        - › ◆ Activity 5
        - ◆ Sequence Flow
        - ◆ End Event
      - › ◆ Process Preconditions
    - ▾ ◇ Use Case Use Case UC3: Delete SpecObject
      - ▾ ◆ Process Main Success Scenario (or Basic Flow)
        - ◆ Start Event
        - ◆ Sequence Flow
        - › ◆ Activity 1
        - ◆ Sequence Flow
        - ◆ End Event
      - ▾ ◆ Process Preconditions
        - ▾ ◆ Start Event
          - ◆ Documentation A ReqIF model exists, is open and at least one SpecObject exists.

1   USE CASE UC1: CREATE A NEW SPECOBJECT

Preconditions:

▪ ReqIF model exists and is open.

Main Success Scenario (or Basic Flow):

1. We assume that a Specification exists and is open (not required for alternative scenario)
2. Double click on the cell in the Specification Editor to be edited.
3. Select the Child or Sibling submenu.
4. Select the desired Spec Object Type (or none) from the submenu.
5. Note that some cells may not be editable, in which case nothing will happen.

Alternative 1 Create in Outline:

*a. The same workflow works for elements that are shown underneath "Specifications" in the outilne.
2a. It is also possible to create children of the "SpecObjects" folder in the outline, but in this case, no SpecHirarchy will be created.
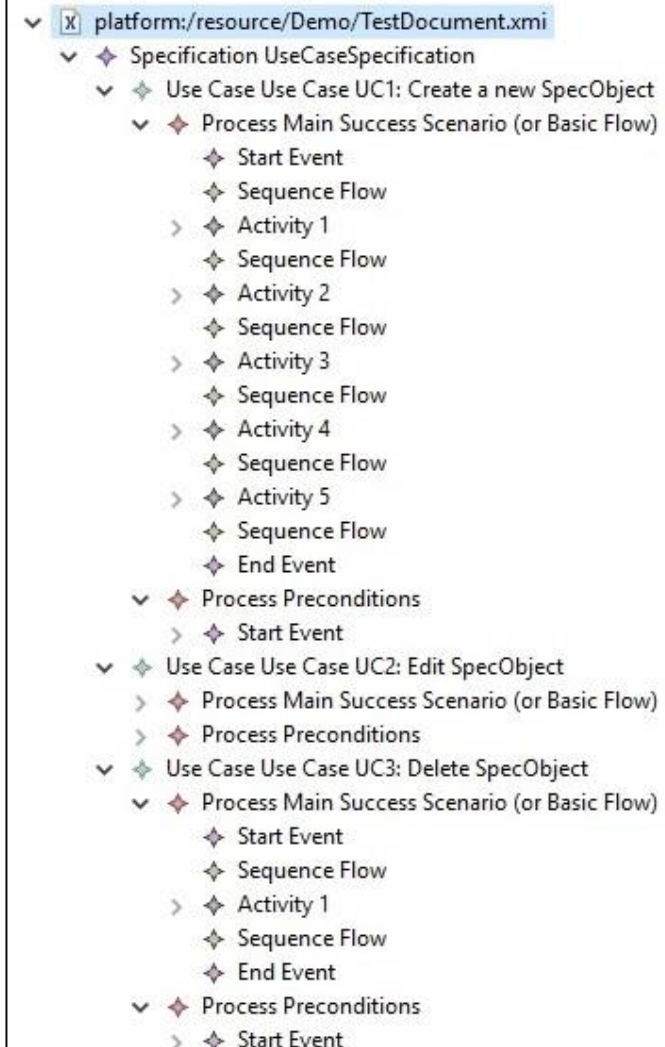
Alternative 2 Keyboard Shortcut:

*a. The keyboard shortcut Ctrl-Enter will create a SpecHierarchy sibling to the currently selected element and immediately go into edit mode in the currently selected column.

2   USE CASE UC2: EDIT SPECOBJECT

Preconditions:

▪ A ReqIF model exists, is open and at least one SpecObject exisits.

- ⌄ ☒ platform:/resource/Demo/TestDocument.xmi
  - ⌄ ◆ Specification UseCaseSpecification
    - ⌄ ◆ Use Case Use Case UC1: Create a new SpecObject
      - ⌄ ◆ Process Main Success Scenario (or Basic Flow)
        - ◆ Start Event
        - ◆ Sequence Flow
        - › ◆ Activity 1
        - ◆ Sequence Flow
        - › ◆ Activity 2
        - ◆ Sequence Flow
        - › ◆ Activity 3
        - ◆ Sequence Flow
        - › ◆ Activity 4
        - ◆ Sequence Flow
        - › ◆ Activity 5
        - ◆ Sequence Flow
        - ◆ End Event
      - ⌄ ◆ Process Preconditions
        - › ◆ Start Event
    - ⌄ ◆ Use Case Use Case UC2: Edit SpecObject
      - › ◆ Process Main Success Scenario (or Basic Flow)
      - › ◆ Process Preconditions
    - ⌄ ◆ Use Case Use Case UC3: Delete SpecObject
      - ⌄ ◆ Process Main Success Scenario (or Basic Flow)
        - ◆ Start Event
        - ◆ Sequence Flow
        - › ◆ Activity 1
        - ◆ Sequence Flow
        - ◆ End Event
      - ⌄ ◆ Process Preconditions
        - › ◆ Start Event

- **Demonstration:**
  - The initial implemented version of this use case will be presented in the demonstration session.

- **Next steps:**

  - BPMN model transformation to Use case documents (bi-directional transformation)
  - Synchronization between documents and BPMN Models
  - Using technique documents of Ford-Otosan;

Thank you for your attention!
Any question ?