Model-Driven Approach for Metadata Specifications

Joachim Wackerow

COSMOS Conference 2024, Paris

DATA DOCUMENTATION INITIATIVE

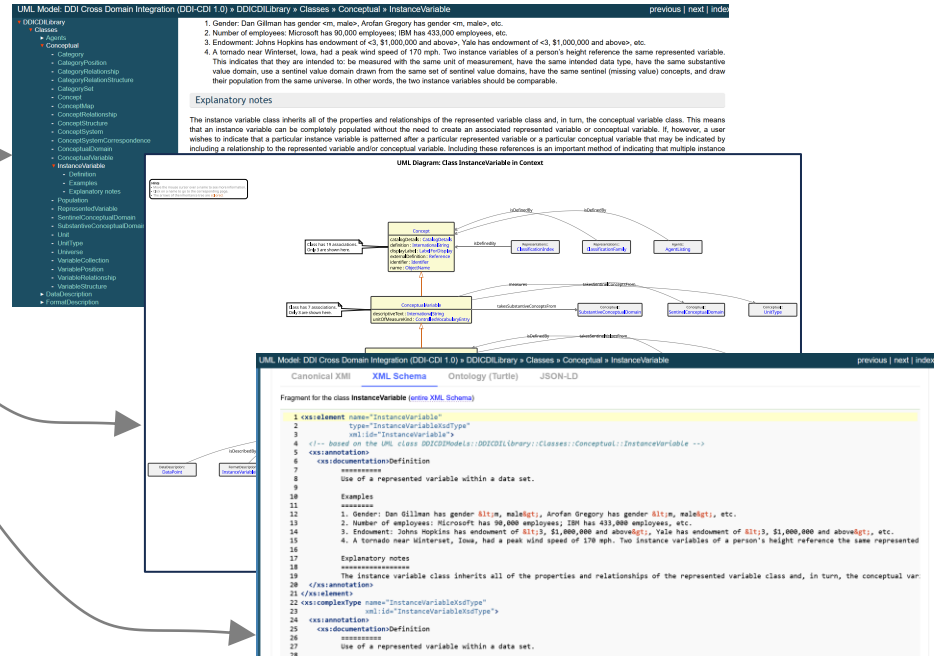
Overview

Single source of truth

UML Editor

Metadata Specification as UML Class Model

Documentation
Diagrams
Encodings



Issues and Resolution

Issues

- UML models are often not interoperable between different UML tools
  - Not all UML class diagram features are implemented in every UML tool
  - The XML Metadata Interchange (XMI) format used is often proprietary

Resolution

- Careful use of UML to achieve interoperability
  - Restriction to a subset of UML class diagram items
  - Usage of Canonical XMI which restricts the choices of the XMI generation rules

UML Class Model Interoperable Subset (UCMIS)

The **objectives** are to have a UML class model …

- which is the **single source of truth** for class-level documentation and derived target languages (syntax representations/encodings),
- which provides consistency over time,
- can be further processed in UML tools,
- which ensures the consistency across the target languages, resulting in interoperability on this level,
- which can be used for future target languages.

UCMIS (cont.)

- UCMIS, a **subset of UML class diagram items**, is intended for data modeling
- It focuses on core items that are familiar from object-oriented programming
- The subset focuses on items that describe classes, describe their relationships to each other, and their attributes
- The subset ensures structural interoperability between UML tools

Git repository: https://bitbucket.org/ddi-alliance/ucmis/

Interoperability

UCMIS models as Canonical XMI ensure interoperability on the structural and syntactic level between UML tools.

Canonical XMI

- Canonical XMI (see Appendix B of the OMG XMI 2.5.1 specification) constitutes a specific constrained format of XMI that minimizes variability, provides more predictable identification and ordering, and ensures syntactic interoperability
- UCMIS class models as Canonical XMI can be imported into many UML tools (but no tool exports as Canonical XMI)

Model-Driven Products

- Field-level documentation: one page per class and data type
- Syntax representations: XML Schema, RDF (ontology in Turtle, JSON-LD, in the works: SHACL and ShEx)
- Further model processing in UML tools

Example page: https://tinyurl.com/ddicdiexample

Model-Driven Products: Documentation per Class

1. Gender: Dan Gillman has gender <m, male>, Arofan Gregory has gender <m, male>, etc.
2. Number of employees: Microsoft has 90,000 employees; IBM has 433,000 employees, etc.
3. Endowment: Johns Hopkins has endowment of <3, $1,000,000 and above>, Yale has endowment of <3, $1,000,000 and above>, etc.
4. A tornado near Winterset, Iowa, had a peak wind speed of 170 mph. Two instance variables of a person's height reference the same represented variable. This indicates that they are intended to: be measured with the same unit of measurement, have the same intended data type, have the same substantive value domain, use a sentinel value domain drawn from the same set of sentinel value domains, have the same sentinel (missing value) concepts, and draw their population from the same universe. In other words, the two instance variables should be comparable.

Explanatory notes

The instance variable class inherits all of the properties and relationships of the represented variable class and, in turn, the conceptual variable class. This means that an instance variable can be completely populated without the need to create an associated represented variable or conceptual variable. If, however, a user wishes to indicate that a particular instance variable is patterned after a particular represented variable or a particular conceptual variable that may be indicated by including a relationship to the represented variable and/or conceptual variable. Including these references is an important method of indicating that multiple instance variables have the same representation, measure the same concept, and are drawn from the same universe. If two instance variables of a person's height reference the same represented variable. This indicates that they are intended to: be measured with the same unit of measurement, have the same intended data type, have the same substantive value domain, use a sentinel value domain drawn from the same set of sentinel value domains, have the same sentinel (missing value) concepts, and draw their population from the same universe. In other words, the two instance variables should be comparable. The instance variable describes actual instances of data that have been collected.

Diagram

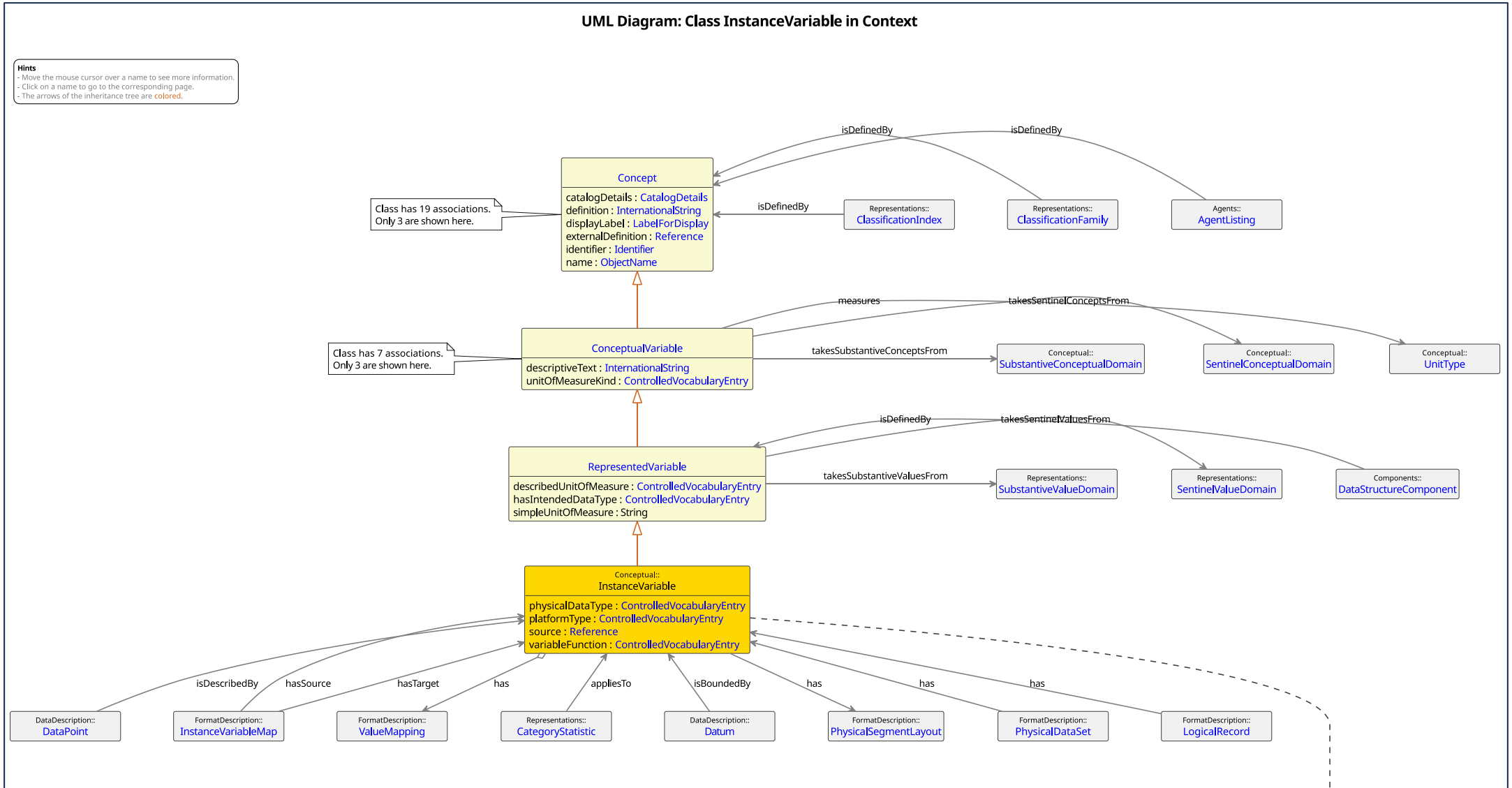
Inheritance

Attributes

Associations

Syntax representations / encodings

Model-Driven Products: Diagram per Class



Model-Driven Products: Encodings per Class

UML Model: DDI Cross Domain Integration (DDI-CDI 1.0) » DDICDILibrary » Classes » Conceptual » InstanceVariable previous | next | index

Canonical XMI | **XML Schema** | Ontology (Turtle) | JSON-LD

Fragment for the class **InstanceVariable** (entire XML Schema)

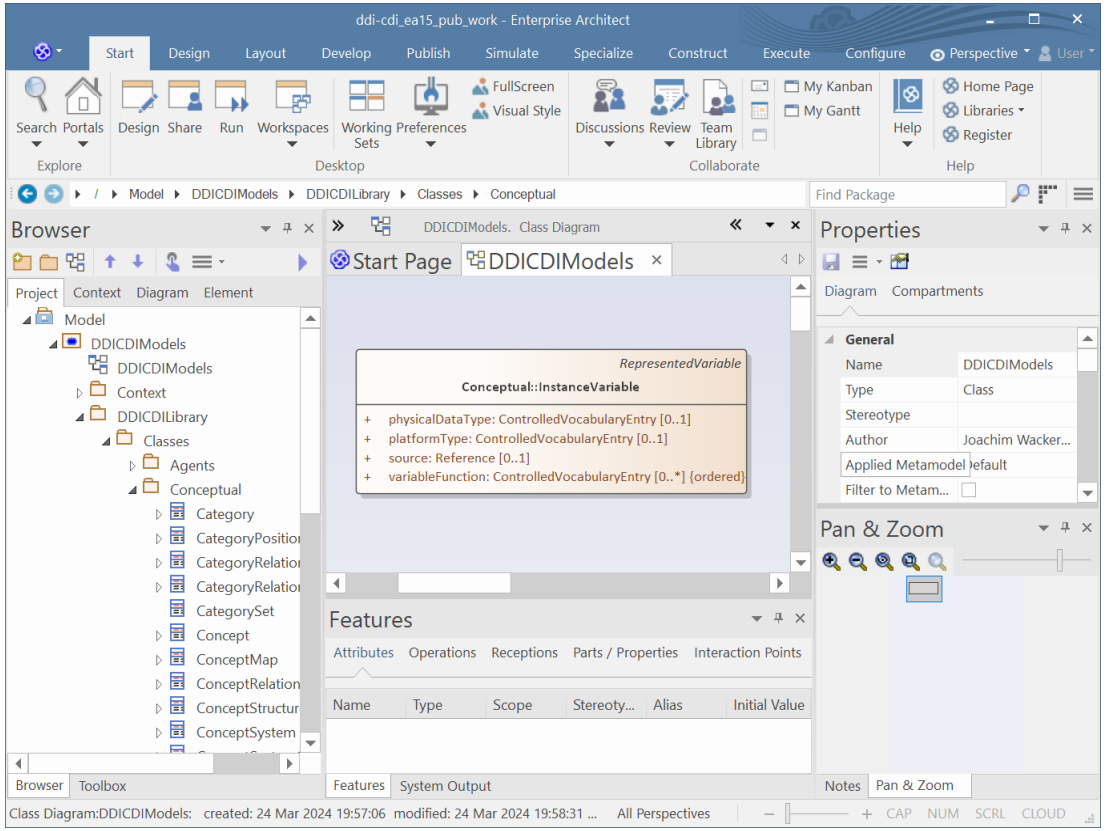
```
1 <xs:element name="InstanceVariable"
2             type="InstanceVariableXsdType"
3             xml:id="InstanceVariable">
4   <!-- based on the UML class DDICDIModels::DDICDILibrary::Classes::Conceptual::InstanceVariable -->
5   <xs:annotation>
6     <xs:documentation>Definition
7           ==========
8           Use of a represented variable within a data set.
9
10          Examples
11          ========
12          1. Gender: Dan Gillman has gender &lt;m, male&gt;, Arofan Gregory has gender &lt;m, male&gt;, etc.
13          2. Number of employees: Microsoft has 90,000 employees; IBM has 433,000 employees, etc.
14          3. Endowment: Johns Hopkins has endowment of &lt;3, $1,000,000 and above&gt;, Yale has endowment of &lt;3, $1,000,000 and above&gt;, etc.
15          4. A tornado near Winterset, Iowa, had a peak wind speed of 170 mph. Two instance variables of a person's height reference the same represented
16
17          Explanatory notes
18          =================
19          The instance variable class inherits all of the properties and relationships of the represented variable class and, in turn, the conceptual var
20  </xs:annotation>
21 </xs:element>
22 <xs:complexType name="InstanceVariableXsdType"
23                 xml:id="InstanceVariableXsdType">
24   <xs:annotation>
25     <xs:documentation>Definition
26           ==========
27           Use of a represented variable within a data set.
28
```

UML Model Creation

Model editing in a UML tool like Enterprise Architect
Using only items of UCMIS
Exporting to XMI (often proprietary flavour)

Includes …

- the conceptual structure of a metadata specification,
- the documentation of all individual elements such as classes, data types, and class relationships.



Transformation of Model as XMI

Transformation from proprietary XMI to Canonical XMI

Software tool: to-canonical-xmi (set of XSLTs)

- Intensively tested for Enterprise Architect XMI flavour
- Basic tests for flavors other major UML editing tools
- Output is Canonical XMI which can be imported into many UML tools

XML Metadata Interchange (XMI) is an Object Management Group (OMG) standard for exchanging metadata.

Git repository: https://bitbucket.org/wackerow/to-canonical-xmi/

Model-driven Documentation and Encodings

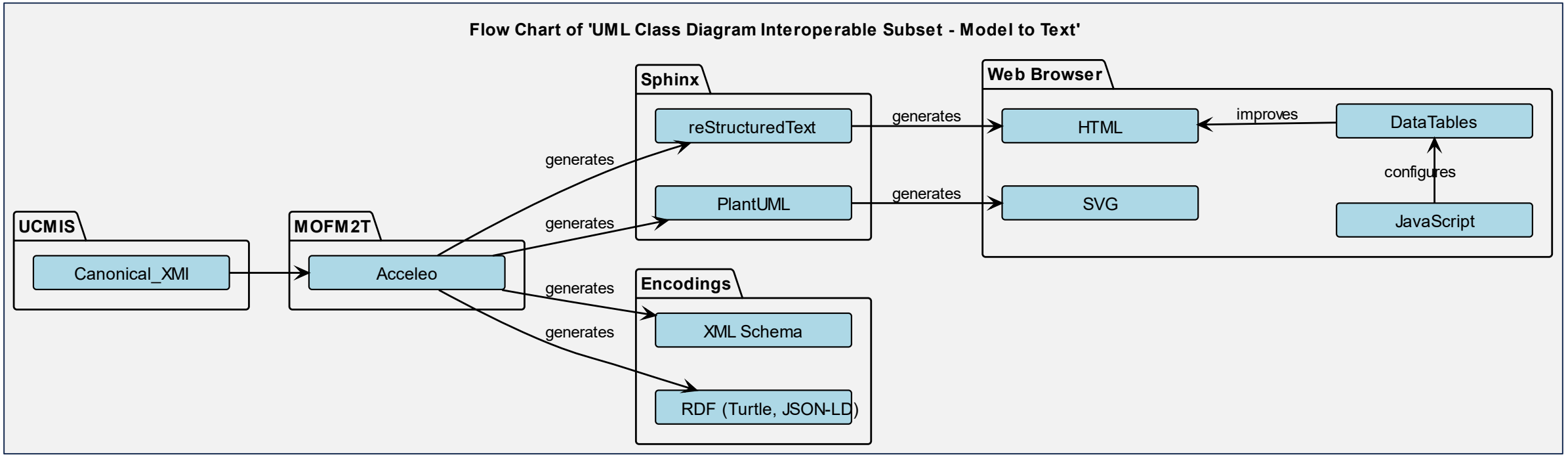
Transformation from UCMIS model as Canonical XMI to documentation and encodings

Software tool: UCMIS Model to Text (UCMIS.M2T)

- UCMIS.M2T is a tool for the generation of the classifier documentation (including UML diagrams) and syntax representations of a model confirming to UCMIS
- It uses the Eclipse Acceleo implementation of the OMG standard MOF Model to Text Transformation Language (MOFM2T™)

Git repository: https://bitbucket.org/wackerow/ucmis.m2t/

UCMIS Model to Text – Components



UCMIS Used for the DDI-CDI Model

- *The **overall concept** is used for the new specification **DDI Cross Domain Integration (DDI-CDI)** forthcoming publication in 2024*

See: https://ddialliance.org/Specification/DDI-CDI/

Credits

- UCMIS is developed by the DDI-CDI working group of the DDI Alliance and planned for publication in 2024.
- The software tools are developed by Joachim Wackerow and contributors with some support of the DDI Alliance.
- Poster author: Joachim Wackerow (joachim.wackerow@posteo.de).