



# D1.2 AIRM Compliance Validator

## Deliverable 1.2

**BEST**

**Grant:**

**699298**

**Call:**

**H2020-SESAR-2015-1**

**Topic:**

**Sesar-03-2015**

**Information Management in ATM**

**Consortium coordinator:**

**SINTEF**

**Dissemination Level:**

**PU**

**Edition date:**

**[30 April 2018]**

**Edition:**

**[00.01.00]**

Founding Members



## Authoring & Approval

### Authors of the document

Name/Beneficiary	Position/Title	Date
<b>Audun Vennesland (SINTEF)</b>	Project Member	18.04.2018
<b>Bernd Neumayr (LINZ)</b>	Project Member	18.04.2018
<b>Christoph Schuetz (LINZ)</b>	Project Member	18.04.2018
<b>Eduard Gringinger (FRQ)</b>	Project Member	18.04.2018

### Reviewers internal to the project

Name/Beneficiary	Position/Title	Date
Scott Wilson (ECTRL)	Project Member	23.04.2018

### Approved for submission to the SJU By — Representatives of beneficiaries involved in the project

Name/Beneficiary	Position/Title	Date
Approved by consortium in accordance with procedures defined in Project Handbook.		30.4.2018

### Rejected By - Representatives of beneficiaries involved in the project

Name/Beneficiary	Position/Title	Date
------------------	----------------	------

### Document History

Edition	Date	Status	Author	Justification
00.00.01	23.03.2017	Document created	Audun Vennesland	First draft
00.00.02	31.03.2017	Planned Content and Structure (PCOS)	Audun Vennesland	Completed PCOS for internal review after comments from co-authors
00.00.03	29.05.2017	Intermediate Proposed	Audun Vennesland	Added background, requirements, principles and suggested approach for ontology matching,

				evaluation, and references
00.00.04	11.04.2018	For input/review by co-authors	Audun Vennesland	Final draft submitted to co-authors for review/comments
	18.04.2018	External Proposed	Audun Vennesland	Version sent to internal review
00.01.00	30.4.2018	Released	Joe Gorman	Updated administrative information on cover pages, ready for release.



## Achieving the **BE**nefits of **SWIM** by making smart use of **Semantic Technologies**

This deliverable is part of a project that has received funding from the SESAR Joint Undertaking under grant agreement No 699298 under the European Union's Horizon 2020 research and innovation programme.

### Executive Summary

---

This report describes the development, evaluation and prototype of the AIRM Compliance Validator, a proof-of-concept application aiming to provide application support for verifying compliance and interoperability between Air Traffic Management (ATM) ontologies. The ontologies used in this work originate from the ATM Information Reference Model (AIRM), the Aeronautical Information Exchange Model (AIXM) and the ICAO Weather Information Exchange Model (IWXXM).

This work presented in this report supports two main use cases:

1. *Semantic interoperability in the development of new ATM information models and services.* By suggesting semantic correspondences between models under development and the AIRM this encourages re-use of standardised information elements rather than the development of new ones.
2. *Compliance Assessment.* Once information models are developed they undergo a process to assure that they are compliant with the AIRM. The AIRM Compliance Validator supports the compliance assessment process as it through an automated process suggests semantic correspondence between elements in the information models under assessment and the AIRM.

The application is developed using principles from ontology matching research. Ontology matching is a process where the aim is to (more or less) automatically identify semantic correspondence between concepts from different ontologies. The components of the application include a set of metrics used for profiling ontologies to be matched, a set of matching algorithms that produce an alignment as a set of semantic correspondences, and strategies that combine the alignments in an optimal manner.

An experimental evaluation show that the AIRM Compliance Validator is able to identify equivalence relations fairly well, and in most cases at a higher quality than two state-of-the-art ontology matching systems used as baseline. Other semantic relations are more challenging. This is both due to the lack of "clues" that can be exploited by the matchers to infer those relations, and the fact that other semantic relations encompass a variety of different relation types, making it difficult to implement generic rules for their identification.

## Table of Contents

Executive Summary .....	4
<b>1 Introduction: About this document .....</b>	<b>7</b>
1.1 Purpose .....	7
1.2 Intended Readership .....	8
1.3 Relationship to other deliverables.....	8
1.4 Acronyms and terminology.....	8
<b>2 Background knowledge .....</b>	<b>10</b>
2.1 Defining compliance with AIRM .....	10
2.2 Ontologies and ontology matching.....	11
2.2.1 A definition of ontology .....	11
2.2.2 The need for ontology matching.....	13
2.2.3 Ontology matching techniques and matchers.....	13
2.2.4 Alignments as results from ontology matching operations.....	14
<b>3 AIRM Compliance Validator .....</b>	<b>15</b>
3.1 Functional requirements .....	15
3.1.1 Overall process flow and requirements mapping .....	19
3.2 Processes of the AIRM Compliance Validator .....	21
3.2.1 Ontology Profiling .....	21
3.2.2 Matcher Selection and Configuration .....	24
3.2.3 Matcher Combination .....	28
<b>4 Experimental Evaluation .....</b>	<b>31</b>
4.1 Experimental Setup .....	31
4.1.1 Datasets.....	31
4.1.2 Reference Alignments .....	32
4.1.3 Evaluating the quality of the prototype .....	32
4.1.4 Experimental Procedure.....	33
4.2 Experimental Results and Findings .....	34
4.2.1 Dataset 1 AIXM AirportHeliport – AIRM AerodromeInfrastructure.....	34
4.2.2 Dataset 2 IWXXM Common – AIRM .....	36
4.2.3 Dataset 3 IWXXM METAR – AIRM.....	38
4.2.4 Dataset 4 AIXM Shared – AIRM .....	39
4.2.5 Dataset 5 AIXM Geometry – AIRM.....	40
4.2.6 Dataset 6 AIXM Obstacle – AIRM.....	42
4.2.7 Dataset 7 AIXM Organisation – AIRM .....	43
4.2.8 Average alignment quality – Equivalence.....	44
4.2.9 Average alignment quality – other relations .....	45
4.2.10 Alignment combination results .....	45

4.2.11	Conclusions from the experimental evaluation.....	46
<b>5</b>	<b><i>User interface for the AIRM Compliance Validator .....</i></b>	<b>48</b>
<b>5.1</b>	<b>Import ontologies .....</b>	<b>48</b>
<b>5.2</b>	<b>Match ontologies.....</b>	<b>48</b>
5.2.1	Select matching strategy .....	48
5.2.2	Matcher configuration .....	49
<b>5.3</b>	<b>Report identified semantic correspondences .....</b>	<b>50</b>
<b>6</b>	<b><i>Conclusions and future work .....</i></b>	<b>52</b>
<b>6.1</b>	<b>Conclusions.....</b>	<b>52</b>
<b>6.2</b>	<b>Further work.....</b>	<b>53</b>
<b>7</b>	<b><i>References.....</i></b>	<b>54</b>
	<b><i>Annex A: Requirements from SWIM Compliance Specifications.....</i></b>	<b>56</b>

# 1 Introduction: About this document<sup>1</sup>

---

## 1.1 Purpose

The Grant Agreement describes the content of this deliverable as follows:

***This deliverable will be provided as a prototype software application that will identify correspondences among an AIRM Ontology developed as a representation of the AIRM Information Models and the ontology modules developed in BEST.***

Ensuring compliance with standardised reference information models is important for interoperability, information quality, efficiency and safety in Air Traffic Management. In this domain the standardised reference model is the ATM Information Reference Model (AIRM), and all other information models targeting information exchange in ATM, should be compliant with this model. However, ensuring such compliance requires significant effort. Both during the model development when modellers investigate potentially re-usable elements in the reference model, and after completion of the model, when its compliance with the AIRM has to be assured and maintained for governance purposes.

Several initiatives in the realm of aviation have investigated the feasibility of introducing semantic technologies as means for improving information management. In BEST we look at ATM information management and represent the AIRM model as well as other information models (for expressing aeronautical information and weather information) as OWL ontologies. Within ontology engineering research, ontology matching as a sub-discipline investigates techniques for (semi) automatic identification of semantic relations between ontologies. Our assumption is that ontology matching techniques lend themselves well to provide automated support for compliance verification, and can reduce much of the human effort that is currently required for compliance management in ATM. Furthermore, such automated support can motivate re-use of standardised information elements in ATM, preventing interoperability threats and unnecessary use of development resources.

While the quality of ontology matching systems has improved over the last years, there is no superior system or technique that performs the best in all contexts and settings. A system or technique has to be adapted and tuned according to its context. Our context is the ATM domain, and in this report, we outline an approach for measuring the semantic correspondence between different ATM-related models and the AIRM. The approach is materialised in a proof-of-concept application called the AIRM Compliance Validator.

---

<sup>1</sup> The opinions expressed herein reflect the author's view only. Under no circumstances shall the SESAR Joint Undertaking be responsible for any use that may be made of the information contained herein.

## 1.2 Intended Readership

This document is targeted towards people having an interest in:

- Development of ATM software
- Application of semantic technologies in ATM
- Application of ontology matching techniques
- SWIM (System-wide Information Management)
- Compliance Assessment

## 1.3 Relationship to other deliverables

Deliverable	Relationship
D1.1 Experimental ontology modules formalising concept definition of ATM data	D1.1 delivers the ontology infrastructure in BEST that will be used for describing and supporting retrieval of relevant aeronautical data by applications developed in other work packages of the project.
D4.4 Tutorial for Software Developers	The principles and techniques described in this deliverable will be included in the tutorial developed in D4.4.
D5.2 Ontology Modularisation Guidelines for SWIM	D1.2 (this deliverable) defines requirements with regards to what type of information needs to be included in the ontology modules in order to ensure that compliance validation can be performed correctly.

## 1.4 Acronyms and terminology

Definition	Explanation
AIRM	ATM Information Reference Model
AIXM	Aeronautical Information Exchange Model
IRI	Internationalized Resource Identifier
IWXXM	ICAO Meteorological Information Exchange Model
Matcher	A matching algorithm that operationalises a technique for identifying alignment between two (or more) ontologies.

Definition	Explanation
NLP	Natural Language Processing
OuA	Object Under Assessment
OWL	Web Ontology Language
RDF	Resource Description Framework
SESAR	Single European Sky ATM Research
SWIM	System-wide Information Management
UML	Unified Modeling Language
WP	Work Package

## 2 Background knowledge

---

We begin this chapter by trying to establish an understanding of what it means to be compliant with the AIRM before we in the remaining part of the chapter describe fundamental aspects of ontologies and ontology matching.

### 2.1 Defining compliance with AIRM

This section includes a formal definition of compliance as it is specified in the AIRM Compliance Framework [1], the AIRM Compliance Handbook [2], the AIRM Foundation Handbook [3], and the SWIM Information Definition Specification [4].

The Compliance Framework inherits the definition of compliance from ISO/IEC 17000 that states:

***“Compliance is the demonstration that specified requirements relating to a product, process, system, person, or body are fulfilled.”***

It is further stated in the AIRM Foundation Handbook:

***“Rule 117: Compliance with the AIRM shall measure the degree of semantic correspondence between the object under assessment and the AIRM.”***

Here, the measurable degree of semantic correspondence is defined by the following values:

- Exact copy: Definition of source and target are exact copy of each other.
- Syntactically equal: Syntax corrections (grammar, spelling)
- Rewritten: The definition has been rewritten for improved quality. The meaning is the same, i.e. the definition still describes exactly the same entity as the target definition.
- Specialised: Source definition is a special case of the target definition.
- Generalised: Source definition is a generalised case of the target definition.

The SWIM Information Definition Specification [4] contains general requirements for information definitions and requirements for semantic correspondence to AIRM. It generalises the AIRM Compliance Framework and the AIRM Foundation Rulebook, and contains in addition to the requirements concrete examples on how the requirements should be addressed. While the AIRM Foundation Rulebook and the AIRM Compliance Handbook specifies the correspondence levels as described in chapter **Feil! Fant ikke referansekinden.**, the SWIM Information Definition Specification specifies that:

***“The mapping of an information concept shall contain a trace from the information concept in the information definition to the AIRM concept that has an equivalent or wider meaning.”***

The AIRM Compliance Framework [1] defines 3 compliance levels that an Object under Assessment (OuA) can claim compliance with (level 3 is the strictest level and each level builds on the level below):

- Level 1 (AIRM Ready)
  - Requires that you map only the higher-level entities of the OUA (e.g. only UML classes, not attributes).
- Level 2 (AIRM Compatible)
  - Requires that you map also the inner properties of OUA entities (UML classes and attributes)
- Level 3 (AIRM Compliant)
  - Requires that you map also the base types and constraints for all OUA constructs.

These documents also include requirements and recommendations that needs to be considered during the development of the AIRM Compliance Validator application:

The AIRM Foundation Rulebook [3] describes normative rules, and informative recommendations and principles for the development and maintenance of AIRM, many of which are relevant for the development and use of the AIRM Compliance Validator.

The AIRM Compliance Handbook [2] explains how to apply and implement the requirements expressed in the AIRM Compliance Framework and achieve compliance reports for the “objects under assessment” (OuA). A compliance report is a filled-in MS Word template and is accompanied with a mapping artefact that contains mappings between entities in the OuA and AIRM. The mapping artefact is typically an Excel file or a UML model holding the mappings. In the work presented in this report we utilise mappings between AIXM and AIRM, and IWXXM and AIRM to establish reference alignments, representing the ground truth to which the quality of the AIRM Compliance Validator is measured against.

The above specifications formulate important requirements that directly or indirectly bear relevance to the development and use of the AIRM Compliance Validator. In Table 2 on page 16 we provide a summary of the most concrete requirements, while a more comprehensive list following from the review of these compliance documents is included in Annex A.

## 2.2 Ontologies and ontology matching

### 2.2.1 A definition of ontology

An ontology is a formal definition of the concepts, properties and interrelationships of the entities that exist in some domain of discourse. It provides a shared vocabulary that can be used to model a domain. The objective of an ontology is to describe some domain, classifying and categorising the elements contained within it.

For the purposes of this report it is also useful to have a more formal and elaborate definition:

*An ontology  $O$  is defined as a tuple  $\langle C, H_C, R_C, H_R, I, R_I, i_C, i_R, A \rangle$  where ontology concepts  $C$  are arranged in a subsumption [specialisation] hierarchy  $H_C$ . Relations  $R_C$  exist between pairs of*

*concepts. The relations themselves can also be arranged in a [specialisation] hierarchy  $H_R$ . Instance data is constituted by individuals  $I$  of specific concepts, and these individuals are interconnected by relational instances  $R_i$ . Individuals and relational individuals are connected to concepts and relations by instantiations  $i_C$  and  $i_R$  respectively. Additionally, one can define axioms  $A$  which can be used to [further express constraints and] infer knowledge from the ontology structure and associated instances” [5].*

OWL (Web Ontology Language) [6] is a popular ontology language, and is the formalism we use in the BEST project. The main building blocks of OWL are entities and axioms. An entity is either a class, an object property, a data property or an individual, and is identified by an IRI (Internationalized Resource Identifier). Classes are lightweight objects that in themselves do not hold information about definitions that may apply to themselves; this information is taken care of by the ontology object through axioms relating to the class level. Object properties are binary associations between individuals (real instances) and compared to UML associations OWL object properties can have additional characteristics (e.g. that an object property expression is transitive, inverse, ir-/reflexive, a-/symmetric, and so on). Data properties relate an individual to a concrete data value (for example a value of type xsd:string). As with classes, both object properties and data properties can be organised hierarchically.

We distinguish between monolithic ontologies, which are typically characterised as ontologies large in size and complexity, and often spanning several different topics and knowledge areas, and ontology modules, which aim at providing ontology users with the knowledge they require, reducing the scope as much as possible to what is strictly necessary [7]. As mentioned earlier, an ontology consists of a set of axioms, i.e. logical statements, that holds some knowledge. An ontology module represents a particular subset of these axioms, and encapsulates a subset of the axioms compared to the “monolithic” ontology. For example, if we are interested in only the knowledge about the concept Aircraft in AIRM, we can represent this knowledge in an Aircraft ontology module, while disregarding other axioms from the AIRM ontology that are not relevant for expressing knowledge about an Aircraft.

AIRM is a reference model that addresses semantic interoperability through harmonised and agreed definitions of the information being exchanged in ATM [8]. In D1.1 [9] of the BEST project a monolithic AIRM OWL ontology was developed from a transformation from the original AIRM UML model. In total the AIRM ontology consists of 1177 classes, 3272 object properties, 1972 data properties and 3727 individuals.

In D1.1 [9] of the BEST project, a set of ontology modules were developed from the AIRM UML model, the AIXM UML model and the IWXXM UML model. Table 1 lists the ontology modules and some statistics associated with them. In D5.2 (Ontology Modularisation Guidelines) [10], the AIRM-BaseInfrastructure module was decomposed into five smaller and more specific modules.

*Table 1. Ontology Modules*

Ontology Module	Classes	Object properties	Data properties	Individuals
AIRM-Aircraft	93	84	33	182
AIRM-AerodromeInfrastructure	117	345	69	0
AIRM-NavigationInfrastructure	34	70	39	0
AIRM-SurveillanceInfrastructure	34	21	17	0

Ontology Module	Classes	Object properties	Data properties	Individuals
AIRM-Obstacle	12	27	8	0
AIRM-BaseInfrastructureCodelists	100	0	0	1574
AIRM-Meteorology	74	69	15	97
AIRM-Stakeholders	148	131	40	316
AIRM-Common	78	44	19	396
AIXM-AirportHeliport	196	312	133	569
AIXM-Obstacle	24	35	10	132
AIXM-Organisation	15	22	8	23
AIXM-Geometry	11	8	19	4
AIXM-Shared	33	39	36	103
IWXXM-METAR	56	70	53	25
IWXXM-TAF	38	56	32	28
IWXXM-Common	10	2	0	0

## 2.2.2 The need for ontology matching

With the introduction of the Semantic Web, a vast amount of more or less formalized ontologies have been and are currently being developed in different domains as well as within the same domain. This results in a situation where ontologies within the same domain end up having duplicate concepts and where semantically equivalent entities have different syntactic and semantic representations. The latter is called the heterogeneity problem, where different terms are being used for the same meaning or the same term is being used for different concepts [11]. Such inconsistency represents a major challenge for those that employ ontologies, be it human users or semantic-aware software components that depend on a consistent representation and interoperation of the knowledge they utilise.

Ontology matching is the process of automatically identifying alignment between heterogeneous ontologies. Being a mature field of research, there exists a large variety of techniques suggested that aim to discover both syntactic and semantic similarity between ontology entities in order to mitigate the heterogeneity problem stated above.

## 2.2.3 Ontology matching techniques and matchers

Euzenat and Shvaiko [3] distinguishes between element-level techniques and structure-level techniques. *Element-level techniques* focus on the ontology entities themselves while disregarding their relations with other entities. Examples of such techniques are terminological or string-based similarity measures (which might identify correspondences based on name similarity), lexical techniques (e.g. using Natural Language Processing (NLP) and lexical resources to capture conceptual similarity and hence correspondence between entities not necessarily having the same name) and informal or formal resource-based techniques which employ external sources, either formal ones such as ontologies or informal sources such as web sites or documents, to improve the matching operation. *Structure-level techniques* on the other hand analyse how entities (or their instances) appear together in a structure. Some examples of structure-level techniques are graph-based techniques (such as the

use of graph algorithms to identify similar neighbouring entities and relations and thereby calculate correspondence), model-based techniques (e.g. the use of description logic reasoning in order to identify correspondence on the basis of semantic interpretation) and instance-based techniques (for example using statistical methods to compare sets of class instances to identify correspondence between these classes). A matcher is an algorithm that operationalises one or more of the mentioned techniques. It takes two input ontologies as input and uses one or more of the abovementioned techniques to calculate a confidence measure (also called strength) for each identified correspondence (see Figure 1).

## 2.2.4 Alignments as results from ontology matching operations

The output of an ontology matching task is an alignment. An alignment contains a set of correspondences (also called cells) between an entity from the first input ontology and an entity of the second input ontology. Further the correspondence includes a relation type that defines what relation holds between the two entities and a confidence measure that states how much confidence the matching system has in the proposed correspondence. The relation type is normally = (equivalence), < (less than) or > (greater than), but other relations are also possible [12]. The confidence measure is typically a value between 0 and 1, but also Boolean values are possible (true or false). The Alignment Format [13], which is represented in RDF, is the de facto standard for representing alignments from ontology matching tasks. An example of a correspondence in an alignment is shown in Figure 1.

```
<map>
  <Cell>
    <entity1 rdf:resource="http://project-best.eu/owl/airm-mod/meteorology#RunwayVisualRange" />
    <entity2 rdf:resource="http://project-best.eu/owl/iwxm-mod/metar#AerodromeRunwayVisualRange" />
    <measure rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.8504</measure>
    <relation>=</relation>
  </Cell>
</map>
```

Figure 1. Example of a correspondence expressed in the Alignment Format

## 3 AIRM Compliance Validator

---

This chapter describes the functionality and inner workings of the AIRM Compliance Validator. We start by listing a set of functional requirements elicited from relevant SWIM and AIRM Compliance specifications. We then describe the processes of the AIRM Compliance Validator in detail.

### 3.1 Functional requirements

This section lists a set of requirements relevant for the development of the AIRM Compliance Validator and how they have been addressed. The requirements are classified into requirements originating from AIRM Compliance documentation and requirements related to required (and typical) ontology matching) functionality of the AIRM Compliance Validator. For the requirements originating from the compliance documentation we include a trace back to the compliance document(s) from which the requirement is elicited from. We refer to Annex A where we present a full list of requirements and recommendations.

Table 2. Functional requirements for the AIRM Compliance Validator

Requirement ID	Requirement	Requirement Source	How the requirement is addressed
<b>F-REC-1</b>	All ontology entities require definition	ACF-Req-4, ACF-Req-6, ACF-Req-11, ACF-Req-13, ACF-Req-14, ACF-Req-15, AFH-Rule 116, AFH-Rule 60, AFH-Rule 108, AFH-Rule 59, SWIM-INFO-001	For assessing the compliance between an OuA and AIRM also the definitions must be compliant. The definitions in the UML models have therefore been included in the transformation to OWL.
<b>F-REC-2</b>	Degree of correspondence should be provided to the end-user	SWIM-INFO-016, AFH-Rule 60, ACF-Req-16, ACF-Req-17, ACF-Req-18	We adhere to the requirement from the SWIM Information Definition Specification that states that the mapping should “contain a trace between the information concept in the information definition [interpreted as the model being developed/assessed] to the AIRM concept that has an equivalent or wider meaning. As a conclusion we categorise semantic correspondences as either: “Equivalent” or “Other semantic relation” where the latter includes specialisation/generalisation, meronymic relations (part-whole relations), and other relations other than “Equivalent”. With regards to the semantic correspondence degrees listed in section <b>Feil! Fant ikke referansekinden.</b> , several of these require human judgement (e.g. whether or not a concept definition is a re-written version of an AIRM concept definition).
<b>F-REC-3</b>	All entities need a unique identifier	ACF-Req-11, AFH-Principle 23, SWIM-INFO-008, SWIM-INFO-019	Each entity is uniquely identified by a IRI (Internationalized Resource Identifier). SWIM-INFO-019 requires that a semantic trace towards an AIRM element, should include an AIRM unique identifier. This could be accomplished by including this URN identifier as an annotation property to each entity in the transformation process from UML to OWL. However, as this was not done at the time of the transformation process in BEST, this will have to be included as future work.

Requirement ID	Requirement	Requirement Source	How the requirement is addressed
<b>F-REC-4</b>	The matching result should include correspondences between: <ul style="list-style-type: none"> <li>• Classes</li> <li>• Object Properties</li> <li>• Data Properties</li> <li>• Individuals (code list values)</li> </ul>	ACF-Req-16, ACF-Req-17, ACF-Req-18, AFH-Rule 17, AFH-Rule 62, SWIM-INFO-017	In this work we only provide mappings at class level, that is, Level 1 – AIRM Ready (see chapter <b>Feil! Fant ikke referansekinden.</b> ). Matching of properties for the purposes of producing an alignment between them would require different pre-processing strategies, although some of the same principles as applied for class matching (e.g. similarity between definitions) could be applied. The AIRM Compliance Validator utilises object- and data properties in the computation of semantic correspondence between classes.
<b>F-REC-5</b>	The AIRM Compliance Validator shall have functionality that lets the end-user import two ontologies.	Not documented, but required functionality for this type of application	The AIRM Compliance Validator shall provide functionality that enables the end-user to easily import two ontologies (either OWL or RDF) that will be matched in subsequent operations.
<b>F-REC-6</b>	The AIRM Compliance Validator shall have functionality to enable the end-user to configure his/her preferred matching parameters.	Not documented, but expected functionality for this type of application	For example, similarity thresholds, preferred set of matchers, format of matching result output (see F-REC-9).
<b>F-REC-7</b>	The AIRM Compliance Validator shall produce an alignment consisting of a set of semantic correspondences between the matched ontologies	Not documented, but required functionality for this type of application	Here, the alignment should follow alignment representation standards (the Alignment Format described in chapter 2.2.4) with possible extensions required from F-REC-9)

Requirement ID	Requirement	Requirement Source	How the requirement is addressed
<b>F-REC-8</b>	The AIRM Compliance Validator shall have functionality that enables the end-user to easily consolidate an assessment report of the matching result.	ACF-Req-10, ACF-Req-16, ACF-Req-17, ACF-Req-18, ACF-Req-19, AFH-Principle 12, AFH-Rule 119, AFH-Rule 120	From the alignment produced by the matching operation, the end-user should have tool support that visualises the alignment produced. Such visualisation could be provided by transforming the alignment to a report akin to an assessment report / mapping artefact in Excel.
<b>F-REC-9</b>	The AIRM Compliance Validator shall have functionality to measure the similarity between definitions provided as rdfs:comments to ontology entities.	ACF-Req-4, ACF-Req-6, ACF-Req-11, ACF-Req-13, ACF-Req-14, ACF-Req-15, AFH-Rule 116, AFH-Rule 60, AFH-Rule 108, AFH-Rule 59	In order to determine if two entities (one from the OuA and the other from AIRM) are semantically similar, their definitions must be compared and measured for similarity.

### 3.1.1 Overall process flow and requirements mapping

From the requirements described in chapter 3.1, Figure 2 suggest the following overall process flow for the AIRM Compliance Validator. The requirements described in the previous section are mapped to the functions responsible for addressing them.

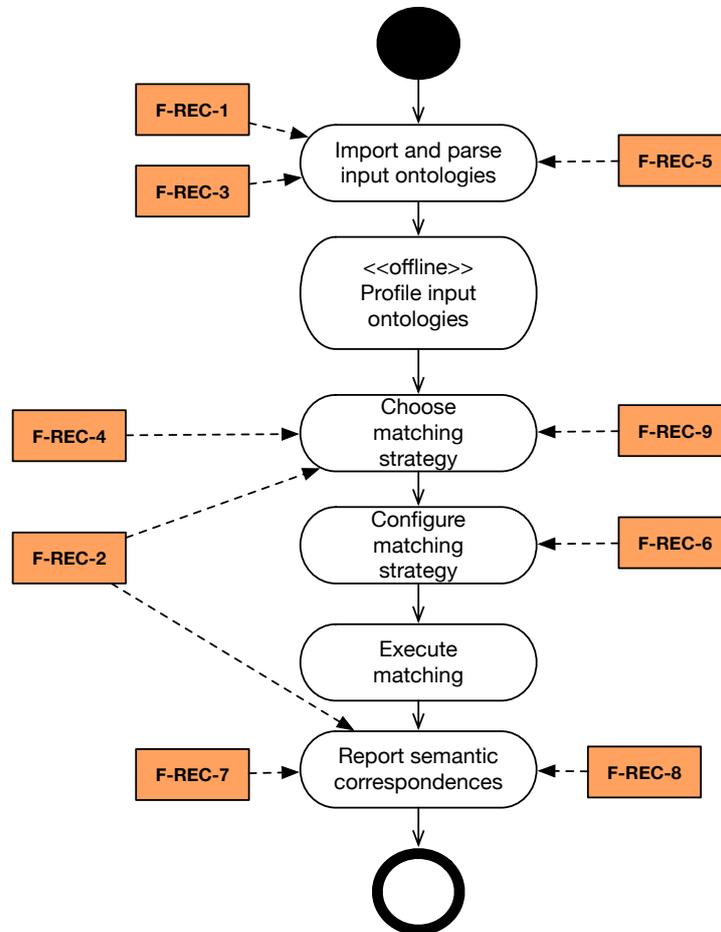


Figure 2. Process Flow AIRM Compliance Validator

**Import and parse input ontologies:** Starting from the top of the activity diagram, the AIRM Compliance Validator needs to import and parse the ontologies to be matched so that they can be easily accessed and processed by the matchers. Both the OWL API [14] and the Alignment API [15] offers OWL parsers and methods that enable easy processing of ontology constructs that can be used for the purposes of matching. According to *F-REC-1* definitions plays an important role in the identification of semantic correspondence, hence definitions<sup>2</sup> (*rdfs:comments*) associated with the ontology concepts must be maintained in this process. Furthermore, *F-REC-3* states that each concept should have a unique

<sup>2</sup> In the context of this deliverable we use 'definition' synonymously with the natural language definition associated with an ontology entity via the *rdfs:comment*.

identifier. This is covered by the fact that each OWL entity has a unique IRI. *F-REC-5* simply states that the AIRM Compliance Validator should have functionality to import two ontologies.

**Profile input ontologies:** In order to determine which matching strategies and specific matchers to use for a matching operation, some analysis of the ontologies to be matched should be performed. Typically, this includes analyses of string-based representation of concept names, and structural and lexical characteristics of the ontologies. This sub-process can be performed separately as an offline process or as an integrated process when running the application. In a more advanced setting, machine learning techniques could select and configure matchers depending on the results from such a profiling process. However, in our case the profiling of the ontologies to be matched was performed offline.

**Choose matching strategy:** Here, the user starts by choosing what type of semantic correspondences are of interest, in our case either equivalence correspondences, other types of correspondences or a combination. This choice influences what types of matcher(s) to execute and how they should be combined. Available matchers should express the type of correspondence identified according to *F-REC-2*. As a minimum, the set of matchers included should automatically identify equivalence correspondences and correspondences that restrict the definition of an AIRM concept, according to the SWIM Information Definition Specification (see chapter **Feil! Fant ikke referansekinden.**). Definitions express the semantic meaning of concepts in these ontologies (and other ontologies), hence it is important that the available matchers are able to process the concept definitions in their computation of semantic correspondence (*F-REC-9*). With regards to *F-REC-4*, we limit the matching to correspondences between classes in this work. This includes UML classes stereotyped as “features” and “objects” in AIXM and “CLDMObject”, “CLDMEntity” and “Codelist” in AIRM.

**Configure matching strategy:** Once relevant matcher(s) is/are selected, various parameters might have to be configured. This relates to *F-REC-6*. One such parameter is the confidence measure applied by the matcher(s). This depends on the techniques used by the matcher(s), but normally if this confidence is set too low there is a danger of having many false positive correspondences, if set too high, there is a danger of omitting true positive correspondences. Other configuration settings may include if and what type of background knowledge (i.e. external sources that can facilitate the matching process) is employed, different parameters that can speed up the matching process, etc. In this work we have limited the configuration settings to the confidence measure.

**Execute matching:** Once the appropriate matcher(s) is/are selected and configured, the actual matching is executed. A number of considerations have to be made, such as scalability characteristics, however, we consider scalability as out of scope since this would require substantial amount of resources to sufficiently analyse.

**Report semantic correspondences:** Once the matching is performed, the semantic correspondences should be presented with a clear indication of which concepts are involved, what type of semantic relation exists between the two concepts, and the confidence of the correspondence holding between the two concepts. The AIRM Compliance Validator returns a report of all semantic correspondences in the Alignment Format (see chapter 2.2.4). With this format the correspondences can be inspected using an XML editor, or easily transformed to another format (e.g. Excel) of an assessment report (*F-REC-7 and F-REC-8*).

### 3.2 Processes of the AIRM Compliance Validator

Figure 3 depicts the three main processes involved when running the AIRM Compliance Validator:

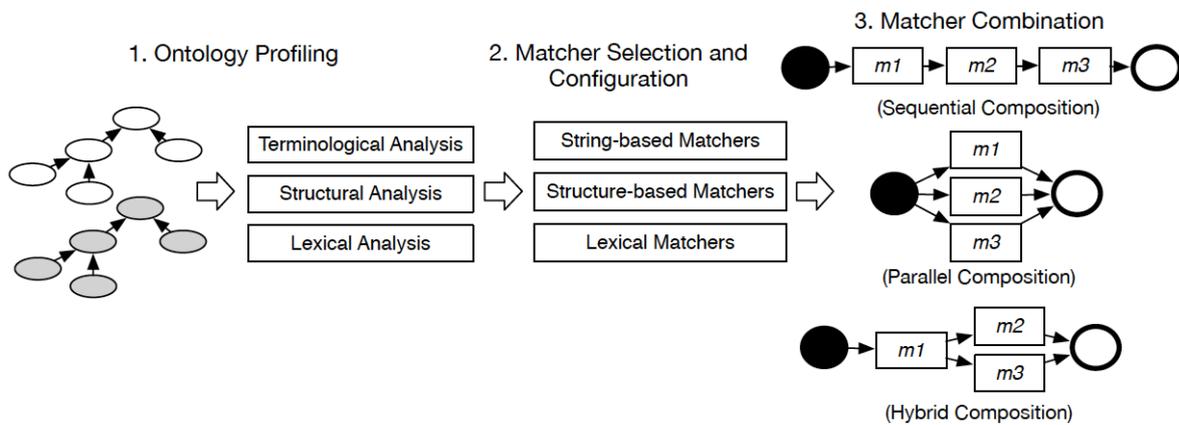


Figure 3. Framework for ontology matching

In the following we describe the three processes more in detail.

#### 3.2.1 Ontology Profiling

First, after the input ontologies have been pre-processed and parsed to an appropriate representation, a set of metrics that characterise the input ontologies are computed in the ontology profiling. These metrics evaluate the terminological, structural and lexical profile of the input ontologies and are computed as an average metric for both ontologies. The ontology profiling has the potential to support both the quality and the efficiency of the ontology matching process. Such a process can contribute to select the most optimal matchers and reduce processing run-time caused by excluding or giving less emphasis to matchers not capable of contributing to the task at hand. Based on these metrics, the set of optimal matchers are identified given the ontologies to be matched. For example, if the metric WordNet Coverage (the percentage of concepts from the ontologies that are also included in the WordNet lexicon [16]) is high, the WordNet-based matcher is included in the set of optimal matchers and weighted according to the metric score. On the other hand, if the WordNet Coverage is low, the WordNet Matcher is either omitted or given a lower weight than matchers that by the Ontology Profiling stage assumedly will perform better. The metrics included in the ontology profiling step are described in Table 3.

Table 3. Ontology Profiling Metrics

Analysis	Metric	Description
Terminological Analysis	Compound Ratio (CR)	Compound words are quite common in ontologies. A compound is a word consisting of one or more individual words, such as AerodromeProtectionArea. If the representation of compounds is high, this suggests that a matcher capable of exploiting such

Analysis	Metric	Description
		<p>linguistic structures should be employed. This also might suggest that the terminology of the ontology is quite "uniform", where existing concept names are appended (either through prefixing or suffixing), for example when creating sub-classes, instead of using a richer and more fine-grained terminology. In such a case, a string-based matcher could perform well. The Compound Ratio is computed by dividing the number of compound class names by the total number of classes:</p> $CR = \frac{ C^{Comp} }{ C }$ <p>where <math> C^{Comp} </math> represents the number of classes which is formed as a compound word, and <math> C </math> represents the total number of classes in the ontology.</p>
	Annotation Coverage (AC)	<p>The percentage of entities with annotation (a comment transformed from entity definition in UML), so basically the ontology entities that includes a comment divided by all entities. If this percentage is high, then a matcher specialising in finding similarity among annotation properties (comments) should be applied. This metric does not indicate whether such a matcher will be successful, but rather that if the score is low such a matcher won't contribute much. The Annotation Coverage is computed as:</p> $AC = \frac{ C^{Ann} }{ C }$ <p>where <math> C^{Ann} </math> represents the number of classes with a (natural language) definition, and <math> C </math> represents the total number of classes in the ontology.</p>
Structural Analysis	Inheritance Richness (IR)	<p>The Inheritance Richness measures the structural characteristics of the input ontologies as the average number of subclasses per class. Hence, if the Inheritance Richness is high, the concepts in the ontology have many sub-classes, something which could be exploited by a structural matcher. The Inheritance Richness is computed as and returns a real number, not a percentage:</p> $IR = \frac{\sum C_i \in C^{ H^C(C_i, C_i) }}{ C }$ <p>where the number of subclasses of class <math>C_i</math> is defined as <math> H^C(C_i, C_i) </math>, where <math>C_i</math> is a subclass of <math>C_i</math> [17].</p>

Analysis	Metric	Description
	Relationship Richness (RR)	<p>The Relationship Richness computes the percentage of relations that are different from subClassOf relations and can suggest to what extent properties can be exploited to infer entity mappings. If an ontology has a Relationship Richness close to zero, that would indicate that most of the relationships are is-a relations, and a structural matcher could be emphasized. On the other hand, if the Relationship Richness is high, this indicates that the ontology has a high percentage of object properties that could be exploited to infer either class equivalence or subsumption relations. The Relationship Richness is calculated as follows:</p> $RR = \frac{ P }{ SC  +  P }$ <p>Where <math> P </math> represents the number of object properties in the ontology, while <math> SC </math> represents the number of subclasses.</p>
Lexical Analysis	WordNet Synonym Coverage (WSC)	<p>One of the strengths of using WordNet in ontology matching is to identify (synonymic) relations between two concepts that other matchers cannot identify, typically through a shared synset among these concepts. So, if the degree of synonymy among the input ontologies is high, then it is likely that a matcher utilising WordNet synonyms could contribute positively in the matcher composition. This metric measures the extent to which a concept is represented by synonyms in WordNet. It is calculated by accumulating the number of concepts for which there exists a synonym and then divide this number by the total number of classes in each ontology. Whenever the concept name is a compound word, each compound part of the word is treated separately. That means that if a compound concept name (e.g. AerodromeProtectionArea) has a compound part (e.g. Protection) for which there is no set of synonyms in WordNet, it is omitted in the accumulation, and the score is reduced. The WordNet Synonym Coverage is computed as:</p> $WSC = \frac{ C^{WNSyn} }{ C }$ <p>Where <math> C^{WNSyn} </math> represents the accumulated number of classes having a synonym in WordNet and <math> C </math> represents the total number of classes in the ontology.</p>

### 3.2.2 Matcher Selection and Configuration

In the following the different matchers included in the study are described. We have implemented<sup>3</sup> the matchers using the ontology- and ontology matching infrastructures of the OWL API [14] and the Alignment API [13]. We categorise the matchers as string-based matchers, structure-based matchers or lexical matchers according to the framework depicted in Figure 3. Using a variety of matching techniques and algorithms enables exploitation and combination of different features of the ontologies as well as benefit from the synergetic strength of different matchers in order to compute alignments between the input ontologies.

#### Selecting matching algorithms based on ontology profiling

In order to help decide which matchers to implement, we analysed all ontologies in the datasets according to the metrics described in chapter 3.2.1. Table 4 shows the scores from the profiling of the ontologies in the datasets.

Table 4. Results from ontology profiling in datasets

Profiling Metric	D1	D2	D3	D4	D5	D6	D7	Avg.
Compound Ratio	0.94	0.91	0.93	0.93	0.76	0.94	0.92	0.9
Annotation Coverage	0.71	1.0	0.99	1.0	1.0	1.0	1.0	0.96
Inheritance Richness	1.12	1.47	1.48	1.46	1.47	1.48	1.46	1.42
Relationship Richness	0.59	0.56	0.56	0.57	0.57	0.57	0.57	0.57
WordNet Synonym Coverage	0.73	0.75	0.56	0.76	0.87	0.87	0.87	0.77

So, what do these profiling scores tell us? Well, the Number of compounds score tells us that most concept names in these ontologies are compound words (90 percent of all concept names in all ontologies involved are compounds), suggesting that there could be a hierarchical structure where a super concept (e.g. Wind) has children with concept names that append their parent (e.g. AerodromeSurfaceWind). This could be utilised by a subsumption matcher that identifies for example that AerodromeSurfaceWind is a specialisation (child concept of) of Wind. Furthermore, it suggests that it could be difficult to straightforwardly utilise lexical resources such as WordNet, since such resources often hold mostly general terms.

The Annotation Coverage shows that almost all concepts are well defined in the sense that they have a definition associated with them. This means that a matcher that analyses (similarity) between the concepts' definitions should be included in the experimentation. Of course, even if definitions are present, it doesn't mean that similarity can be inferred from them, but semantically similar concepts tend to have semantically similar definitions also, so such a matcher should be included in the evaluation.

<sup>3</sup> The ISub string matcher is a re-use from the Java API OntoSim (<http://ontosim.gforge.inria.fr/>), but the other matchers are developed as part of the BEST project.

The Inheritance Richness and the Relationship Richness scores in combination reveal that these ontologies have quite flat structures with few subclasses per class, but that the representation of relations (object properties) between the classes is relatively high. Based on this, matchers that exploit object properties as means for inferring similarity between classes should be included.

As earlier mentioned, the fact that most concept names are compounds makes the use of WordNet challenging. However, by splitting each concept names into individual compound tokens, e.g. [Aerodrome][Surface][Wind] as in the example used earlier, we then analysed to what extent each individual part had a representation of synonyms in WordNet, resulting in the WordNet Synonym Coverage. This metric represents an extension of the WordNet Coverage used for example in [18], [19].

From the above profiling and discussion, we developed the following set of matching algorithms:

*Table 5. Matching Algorithms in AIRM Compliance Validator*

Matcher	Target	Relation Type
ISub String	Entity - Name	Equivalence
Definitions Matcher	Entity - Definition	Equivalence
Range Matcher	Entity - Structure	Equivalence
Property Matcher	Entity - Structure	Equivalence
WordNet Synonym Matcher	Entity - Lexical Properties	Equivalence
Closest Parent Matcher	Entity - Structure	Subsumption
Compound Matcher	Entity - Name	Subsumption
Definitions Subsumption Matcher	Entity - Definition	Subsumption

The above matchers are described in detail in the following:

### Algorithms for equivalence matching

The *ISub String Matcher* is a string matching algorithm developed by Stoilos et al. [20]. The ISub algorithm applies three functions in order to find the similarity between two entity names  $e_1$  and  $e_2$  and considers both the commonality and difference between strings when computing a similarity score. The algorithm proceeds as follows:

$$ISubSim(e_1, e_2) = Comm(e_1, e_2) - Diff(e_1, e_2) + winkler(e_1, e_2)$$

The three functions are:

- The commonality function (*Comm*) is motivated by the substring metric where the biggest common substring between two strings is computed. This process is further extended by removing

the common substring and by searching again for the next biggest substring until no common substring can be found.

- The difference function (*Diff*) is based on the length of the unmatched strings resulted from the initial matching step (after all common substrings have been identified). The *Diff* function is given less importance than the commonality function (weight parameter 0.6 is a good value according to the authors).
- After the commonality and difference between two strings are computed the Winkler algorithm [21] is used for improving the results.

The **Definitions Equivalence Matcher** treats definitions associated with two entities as sets of individual words. Stopwords that carry little meaning, such as ‘the’, ‘a’, ‘is’, etc., are removed before the definitions are processed further. As with the other algorithms relying on set-theoretic similarity scores, this algorithm employs the Jaccard [22] set-theoretic similarity measure to compute a similarity score between the two definitions. This measure computes the intersection over union for sets *A* and *B* of words from two definitions as follows:

$$JaccardSim(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

If the two definitions *A* and *B* are identical, the Jaccard similarity is 1, if they are completely dissimilar (no identical words), the Jaccard similarity is 0. If the Jaccard similarity score of the two definitions is above a certain threshold the entities are considered equivalent, so one important factor in this algorithm is the configuration of the Jaccard threshold.

The **Property Matcher** measures the similarity of the properties associated with the entities to be matched. Both object properties and data properties where the entities to be matched are domains are collected into a single set for each entity and compared with Jaccard.

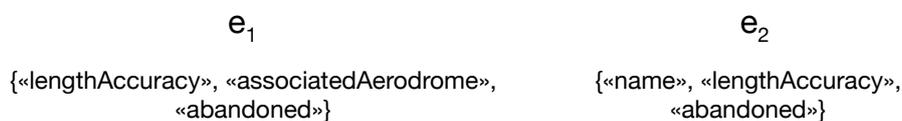


Figure 4. Sets of properties are compared in the Property Matcher

The **Range Matcher** measures the similarity of the sets of range classes of object properties where the entities  $e_1$  and  $e_2$  being matched represent the domain. If the Jaccard set similarity of the object properties’ range classes is above a certain threshold, the matcher considers that the two entities are equivalent.

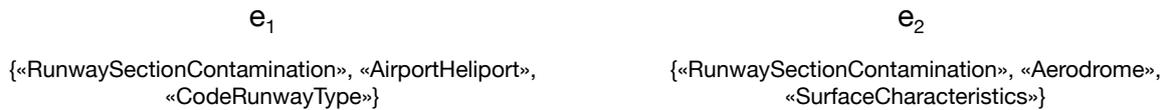


Figure 5. Sets of range classes are compared in the Range Matcher

The **WordNet Synonym Matcher (WNSyn)** computes a similarity score based on how many common WordNet synonyms the two concepts to be matched are associated with. If the concept name represents a compound, it is split into a set of compound parts, and synonyms associated with each part represent individual sets of words taking part in the similarity calculation. The synonyms associated with the respective concepts are represented as sets and a similarity score is computed using Jaccard.

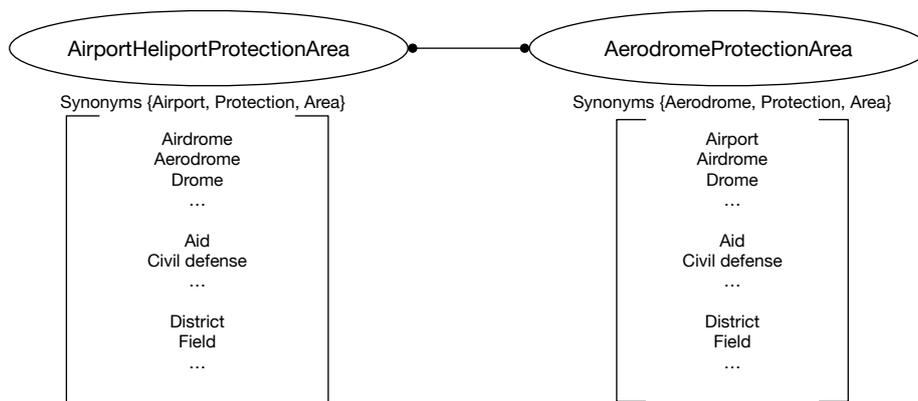


Figure 6. WordNet Synonym Matcher considers equal synonyms as indication of equivalence similarity. In this example no synonyms were found for 'Heliport'.

### Algorithms for identifying “other correspondences”

While the previously presented algorithms seek to identify equivalence relations, the algorithms in this section aim to identify other semantic relations. A challenge with the compliance process of AIRM is that the specialisation/generalisation is not exclusively subsumption relations, but also other semantic relations, for example, part-whole relations between concepts. We therefore generalise and consider all relations that are not equivalence as “other semantic relation”.

The **Closest Parent Matcher** determines that one entity  $e_1$  is a subclass of entity  $e_2$  if the superclass of  $e_1$  has a high similarity with  $e_2$ , as illustrated in Figure 7. This matcher relies on having a graph representation of the ontologies. We implement such a graph structure using a graph database called Neo4J<sup>4</sup>, which is open source.

<sup>4</sup> <https://neo4j.com/>

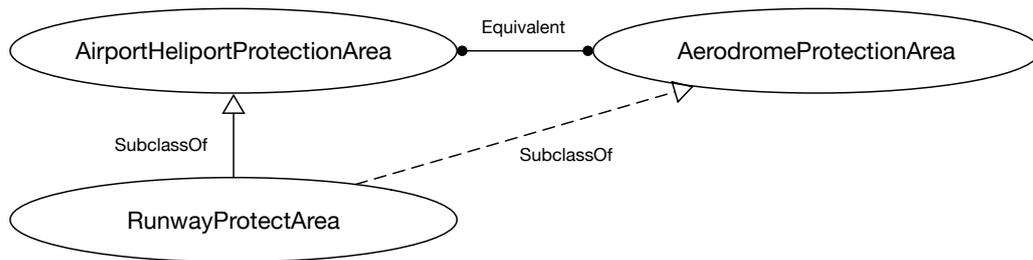


Figure 7. Closest Parent Matcher assumes subsumption based on superclass similarity

The **Compound Matcher** identifies subsumption relations between entities reusing principles from the compound strategy from Arnold and Rahm [23]. Compound means that several individual words are put together to form another word. Here, parts of compounds in entity names are identified and employed as an indicator of a subsumption relation. So, if one or more compound parts in one entity name  $e_1$  is represented as a subset of compounds in another entity name  $e_2$ , the Compound Matcher defines that  $e_1$  subsumes  $e_2$ . In Figure 8, “AerodromeHorizontalVisibility” is subsumed by “HorizontalVisibility”.

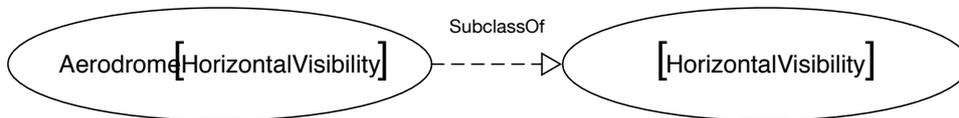


Figure 8. The Compound Matcher identifies subsumption relations based on identification of compounds

The **Definitions Subsumption Matcher** considers both commonality and number of words in the definitions in order to compute if two entities are in a subsumption relation. If the commonality of the definitions is above a certain threshold, we consider the size (number of words) of the definitions as a qualifier for subsumption, where an entity with a smaller definition subsumes an entity with a larger definition. The rationale for this is that the more specific and detailed the entity is, the more text is required to sufficiently describe it.

### 3.2.3 Matcher Combination

After the set of matchers have been selected (on the basis of the ontology profiling scores), the composition of the matchers is selected. Three different strategies for the matcher combination are evaluated in the experimental evaluation, and they are described in the following.

In the **Weighted Sequential Combination (WSC)** the initial alignment from the first matcher is refined by each following matcher in the sequence. Weight is added to correspondences that are identified by two consecutive matchers. If the correspondence is new, that is, identified only by the current matcher, or if the correspondence is only identified by the previous matcher(s) and not the current one, the correspondence is added to the refined alignment with equally reduced weight. As an example, consider that  $m_1$  has produced an alignment that is transferred to  $m_2$ , the next matcher in

the sequence. If the same correspondence (the same two entities and the same relation type) is identified as correct by both  $m_1$  and  $m_2$ , the confidence value associated with this correspondence is increased by a predefined weight. On the other hand, if a correspondence received by  $m_2$  from  $m_1$  is only identified as a correct correspondence by  $m_1$  and not by  $m_2$ , this correspondence is reduced by the same defined weight before the alignment is sent further to  $m_3$ . However, the correspondence is still kept in the alignment. The weighting scheme applied in this study is to add (or reduce) 12 percent to the confidence of the correspondence. Maximum confidence is 100 percent (1.0).

The **Simple Vote** configuration is a parallel combination strategy. Here, all matchers are run in parallel. The alignments they produce are initially treated equally important, but only those correspondences identified by a predefined ratio of matchers (for example using the majority vote, such as three out of five matchers) are eligible for the final alignment.

We have implemented the **AutoWeight++** algorithm [24] as a third combination strategy. As Simple Vote, this is also a parallel combination strategy, but a more sophisticated one, since it includes both matcher configuration and combination. The concept of *highest correspondences* is central in this approach. A correspondence between two entities  $e_1$  and  $e_2$  is considered a highest correspondence if it has a higher confidence value than any other correspondence that includes either  $e_1$  and  $e_2$ . The highest correspondences are used both for automatically configuring the matching algorithms' weight and for combining the individual alignments into an optimal final alignment.

There are two importance coefficients, one at the correspondence level, and the other at the matcher level. For every alignment produced by all matchers, an importance coefficient for each highest correspondence is computed. This importance coefficient considers how many matchers  $n$  have identified this particular correspondence. The importance of each particular highest correspondence is based on how many times this correspondence has been detected as a highest one across all correspondences from all matchers. If the highest correspondence is identified by all matchers (i.e. all alignments), it is omitted since it brings no useful information (i.e. is not discriminative enough).

The importance coefficient for a matcher is calculated by summing the importance coefficients of all highest correspondences produced by that matcher. The weight of a matcher is then computed as the ratio of the importance coefficient for that particular matcher and the sum of the importance coefficient of all matchers.

The next step is to aggregate all correspondences from all matchers into an intermediate common alignment. Now the confidence of each aggregated correspondence is calculated by multiplying their correspondence strength in each alignment (from each matcher) with the weight assigned to the matcher and then summing up those products.

When producing the final alignment (from the aggregated set of correspondences in the intermediate common alignment), Autoweight++ takes an iterative approach. It starts by taking the highest correspondences from the intermediate common alignment. Then in the following iterations, the correspondences that do not include entities taking part in the already established highest

correspondences are processed. The algorithm stops when there are no more correspondences above a given confidence threshold.

## 4 Experimental Evaluation

This chapter describes the experiments performed using the AIRM Compliance Validator prototype as well as evaluation results from the experiments. The evaluation is performed iteratively, improving the prototype and tuning its components in each iteration.

### 4.1 Experimental Setup

In the following we describe the datasets used for experimentation, how the evaluation is conducted, and some findings from the experiments.

#### 4.1.1 Datasets

We have selected 7 datasets for training and experimenting with the approach described in the previous chapter. The datasets include the AIRM ontology and ontology modules from AIRM, AIXM and IWXXM. Before starting the experiments, we processed the ontologies as follows:

- Removed package concepts. The transformation from UML to OWL models reported in deliverable D1.1 [9] maintained the UML packages in order to ensure easy navigation in the OWL ontologies. These package concepts represent only noise in the experiments and were therefore removed.
- Boolean properties in UML were transformed to classes in the resulting OWL ontologies. Since these are not represented as such in the mapping artefacts we base the reference alignments on, we removed them from the ontologies prior to the matching.

The datasets used in the experiments are described in Table 6. Note that the equivalence relations (EQ Relations) category of Reference Alignment include the Exact Copy, Syntactically Equal and Rewritten semantic correspondence degrees from the Excel mapping files (see chapter **Feil! Fant ikke referansekinden.**). The Other Relations category include correspondences of type specialisation and generalisation.

Table 6. A summary of the datasets used in experiments

Dataset #	Ontologies	# Classes	# Object Properties	# Data Properties	Reference Alignment	
					# EQ Relations	# Other Relations
1	AIXM-Airport Heliport	152	226	93	73	1
	AIRM-Aerodrome Infrastructure	195	345	69		
2	IWXXM-Common	8	1	0	0	9
	AIRM-Mono	915	1761	494		
3	IWXXM-METAR	46	46	36	11	7
	AIRM-Mono	915	1761	494		
4	AIXM-Shared	23	24	24	10	0
	AIRM-Mono	915	1761	494		
5	AIXM-Geometry	7	3	12	4	2

	AIRM-Mono	915	1761	494		
6	AIXM-Obstacle	15	27	7	3	2
	AIRM-Mono	915	1761	494		
7	AIXM-Organisation	10	15	5	5	0
	AIRM-Mono	915	1761	494		

### 4.1.2 Reference Alignments

In order to evaluate the performance of the AIRM Compliance Validator, we need to have a comparison base. During this work we have developed a set of reference alignments. The reference alignments represent the correct set of relations between entities in the datasets described in the previous chapter, and act as our comparison base for evaluating the quality of our techniques. The source material for the reference alignments are mapping files in Excel from the compliance assessment process of the exchange models AIXM and IWXXM. These mapping files contain manually developed relations based on expert judgement between the exchange models AIXM/IWXXM and AIRM and have been transformed to reference alignments in the Alignment Format (see chapter 2.2.4) using the Java library Apache POI<sup>5</sup>. Although the Excel mapping files contain relation between properties (associations and roles), we have limited the reference alignments in this experimentation to only contain relations between classes, that is, L1 – AIRM Ready compliance level (see chapter 2.1). For each dataset a filtered reference alignment is produced that only contain relations between concepts from the ontologies representing the particular dataset. For the datasets that include both equivalence and subsumption relations, separate reference alignments have been produced for these types of relations.

### 4.1.3 Evaluating the quality of the prototype

Typically, evaluation of ontology matching techniques is performed using precision and recall against so-called gold standard mappings or reference alignments [25]. Precision measures how many incorrect correspondences the system has managed to avoid and is computed as the ratio of correctly found correspondences (according to the reference alignment) over the total number of found correspondences. Recall measures how many correct correspondences the system manages to identify and is computed as the ratio of correctly found correspondences over the total number of expected correspondences (as expressed in the reference alignment). An evaluation measure that combines precision and recall is the F-measure. This is often used as an overall measure for representing the quality of an ontology matching technique or complete system. In the following we define these evaluation metrics more formally:

#### Precision

Given a reference alignment RA, the precision (P) of an alignment (A) computed by an ontology matching system is computed as:

<sup>5</sup> <https://poi.apache.org/>

$$P(A, RA) = \frac{|RA \cap A|}{|A|}$$

### Recall

Given a reference alignment RA, the recall (R) of an alignment (A) computed by an ontology matching system is computed as:

$$R(A, RA) = \frac{|RA \cap A|}{|RA|}$$

### F-measure

Given a reference alignment RA and a harmonisation number  $x$  between 0 and 1 (normally 0.5 to represent the harmonic mean between precision and recall), the F-measure (FM) of some alignment A computed by an ontology matching system is computed as:

$$FM_x(A, RA) = \frac{P(A, RA) * R(A, RA)}{(1 - x) * P(A, RA) + x * R(A, RA)}$$

#### 4.1.4 Experimental Procedure

For each of the datasets we run all individual matchers (see Table 5 for an overview) using 4 different confidence thresholds (0.5, 0.7, 0.9 and 0.95). This means that for example when confidence threshold 0.5 is applied, all correspondences that have a confidence measure below 0.5 are omitted. We then compare the alignments produced by each individual matcher against the reference alignments described in chapter 4.1.2. The principle parameter for quality is F-measure, since this balances precision and recall, but also precision and recall values are presented since an analysis of these scores can help determine further tuning of the matchers. For each dataset we describe equivalence and subsumption relations separately.

Once a set of individual alignment files are produced by the matchers, we then combine them as described in chapter 3.2.3 in order to see which combination performs best. Based on the F-measure scores of the individual alignments, we select the 3 matchers (i.e. alignments) that perform best on average. These 3 are then combined using the combination strategies described in chapter 3.2.3. The alignments resulting from the combination are measured against the same reference alignments as the individual matching.

For the evaluation of the equivalence relations we compare the results against two state of the art matching systems. The first one, AgreementMakerLight (AML) [26], is an open source ontology matching system which usually ranks as one of the top contenders of the Ontology Alignment Evaluation Initiative (OAEI) [27], an annual evaluation campaign for ontology matching systems. It is run in its default configuration settings mode, meaning that it includes a string matcher, a structural matcher, and a background knowledge matcher that utilises external sources (WordNet). The second baseline system is LogMap [28]. As AML, LogMap is a system that normally ranks as one of the top ontology matching systems in OAEI. We have generated alignments with confidence thresholds at 50, 70 and 90 percent for both baseline systems. AML and LogMap does not offer functionality for identifying other relations than equivalence and in practice there are no other available systems to compare such relations with.

All experiments are run on a MacBook Pro, 3.5 GHz Intel Core i7 processor with 16 GB memory.

## 4.2 Experimental Results and Findings

This chapter presents the results from the experiments per dataset. For each dataset we present charts illustrating the performance with respect to precision, recall and F-measure for the alignments produced by the included matchers. In addition, we present some observations from a manual investigation of the results for each dataset. We end this chapter with a concluding summary of all experiments. Table 7 explains the abbreviations used for the matchers in each chart describing the results from each dataset.

*Table 7. Abbreviations of matching algorithms used in charts*

Matching Algorithm	Abbreviation in chart
ISub String	ISUB
Definitions Matcher	DEF
Range Matcher	RANGE
Property Matcher	PROP
WordNet Synonym Matcher	WNSYN
Closest Parent Matcher	CP
Compound Matcher	COMP
Definitions Subsumption Matcher	DEFSUB

### 4.2.1 Dataset 1 AIXM AirportHeliport – AIRM AerodromeInfrastructure

**Equivalence.** As the chart in Figure 9 shows, the best performing equivalence matching algorithm with respect to F-measure is the Property Matcher configured with confidence threshold 0.5. This configuration achieves an F-measure of 63 percent. The ISub matcher, which only considers entity names obtains a maximum F-measure of 57 percent (with confidence threshold 0.95). The naming convention for elements seem to be quite similar for AIXM and AIRM, which is exploited by the ISub algorithm. The baseline matching system AML achieves a maximum of 62.4 percent F-measure in this dataset, while LogMap achieves an F-measure of 36.9 percent at confidence threshold 0.5.

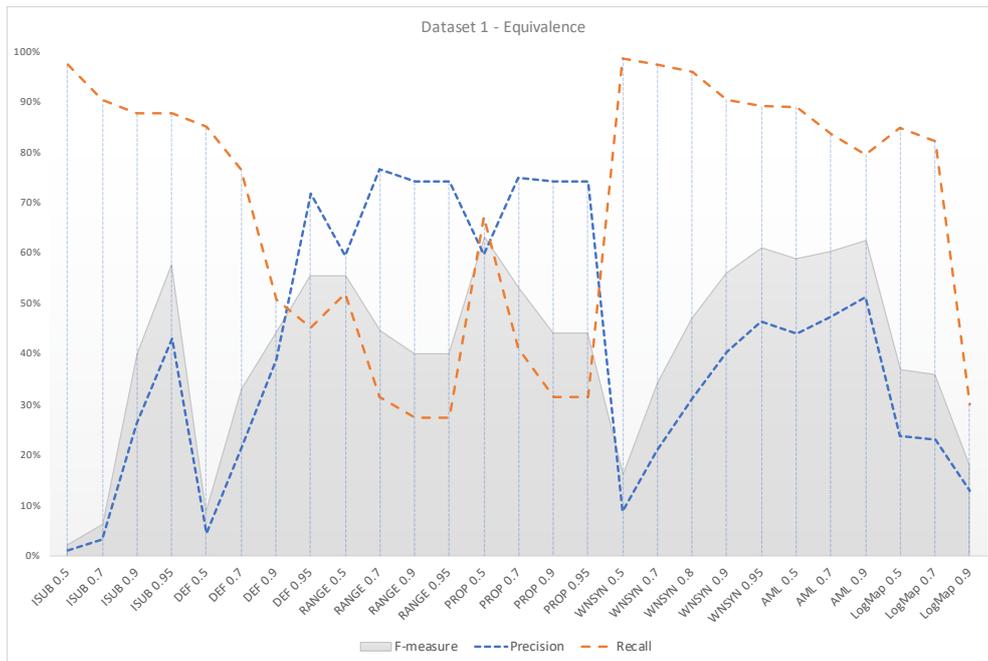


Figure 9. Dataset 1 Equivalence Matching Results (the percentage scores represent the F-measure)

When inspecting all alignments manually we see that the only two correspondences not found by any matcher were:

- *AirportHeliportCollocation – AerodromeCollocation*; and
- *AirportHeliportContamination – AerodromeContamination*.

This could be remedied by having a fixed rule stating that *AirportHeliport* is synonymous with *Aerodrome*. WordNet includes a synonymous relationship between *Airport* and *Aerodrome*, but not *AirportHeliport*.

For the ISub Matcher, the precision would be improved with a confidence of 1.0, but the recall would be worse. In other words, some true positive relations (with confidence lower than 1.0) would be omitted in the alignment. The Range Matcher and the Property Matcher identifies many of the same equivalents as ISub, but also a few true positive *AirportHeliport-Aerodrome* related correspondences that ISub is not able to catch.

**Other correspondences.** The results from the matching operation are shown in the chart in Figure 10. The only relation in the reference alignment is *RunwayElement-RunwayElement*, which intuitively suggests an equivalence relation. The reason why this seemingly equivalent relation is considered as a different semantic relation, is that the *definition* in AIXM is more specific than the AIRM one.

The Closest Parent Matcher with confidence thresholds 0.5 and 0.7 identified it with a confidence of 82 percent, but since these alignments also included a very large number of false positive relations,

the precision becomes very poor (6 percent and 9 percent respectively), resulting in very low F-measure scores.

The Definitions Subsumption matcher at threshold 0.5 identified it with a confidence of 52 percent, but as with the Closest Parent Matcher alignments, the precision and consequently the F-measure was very low due to a very large number of false positives.

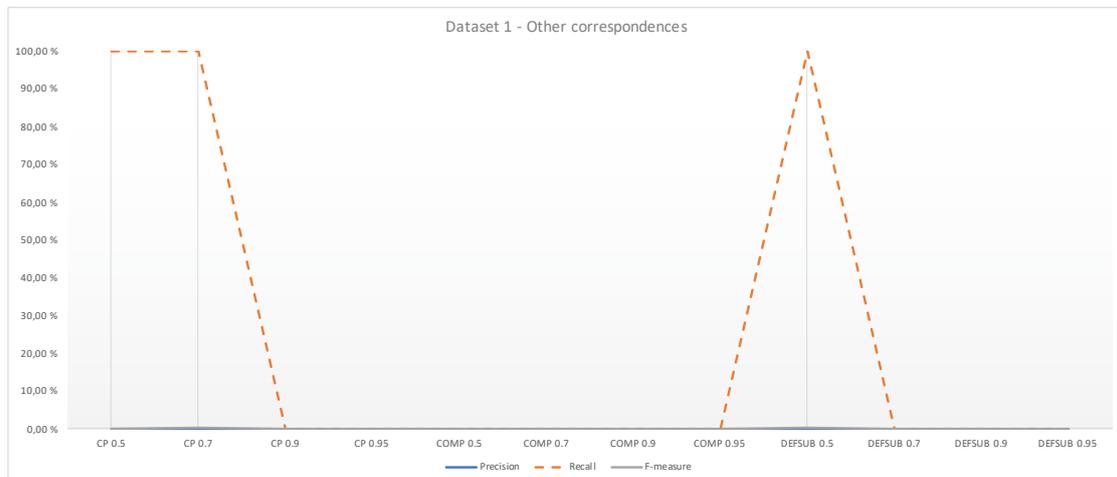


Figure 10. Dataset 1 Subsumption Matching Results

The next two datasets involve IWXXM ontology modules. These two datasets are challenging, based on the following observations:

- The definitions associated with the IWXXM concepts have little resemblance to the definitions associated with the AIRM concepts.
- Concept naming conventions differ significantly.
- There are several complex mappings, where a single entity in the IWXXM ontologies is mapped to several AIRM concepts.
- Asymmetric mappings, where for example a IWXXM concept representing a UML class is mapped to an AIRM concept originally representing a UML code list or message (IMMessage).

#### 4.2.2 Dataset 2 IWXXM Common – AIRM

In this dataset there are no equivalence relations, so here the focus is on other types of correspondences. There are 9 correspondences in the reference alignment. As shown in Figure 11, the best performing matcher is the Definitions Subsumption Matcher with equal scores at confidence thresholds 0.9 and 0.95. Both of them produced 1 true positive correspondence and no false positive ones, resulting in a 100 percent precision. However, since they missed the other 8 correspondences in the reference alignment, the recall was quite low.

The Compound Matcher computed also 1 true positive relation, but 1 false positive one. The Closest Parent Matcher computed 1 true positive relation, and 27 false positives. All the true positive relations identified by these 3 matchers are different relations, so in total 3 out of 9 relations in the reference alignment were identified.

The relations from the reference alignment not identified by any matcher were:

- *AerodromeForecastWeather* < *CodeSignificantWeatherQualifierType*
- *AerodromeForecastWeather* < *CodePrecipitationType*
- *AerodromeForecastWeather* < *CodeWeatherPhenomenonType*
- *AerodromeForecastWeather* < *CodeObscurationType*
- *AerodromeSurfaceWindTrendForecast* < *Wind*
- *AerodromeSurfaceWindTrendForecast* < *TREND*

where  $X < Y$  means that  $X$  is a specialisation of  $Y$ .

All these correspondences represent complex mappings, meaning that there is an 1..n relation between the IWXXM concept and the AIRM concepts. For the four first ones in the bullet list, the IWXXM concept *AerodromeForecastWeather* (which is a code list in the original UML model) has a relation to all four AIRM classes (which are also code lists in the original UML model) on the right-hand side. In the last two correspondences the IWXXM *AerodromeSurfaceWindTrendForecast* concept is mapped to the *Wind* class and to the *TREND* class (which in the original UML model is a separate message represented as an UML class). Neither the concept names, structure, or definitions can help infer equivalence relations for these complex mappings, so none of the matchers can make any contribution here.

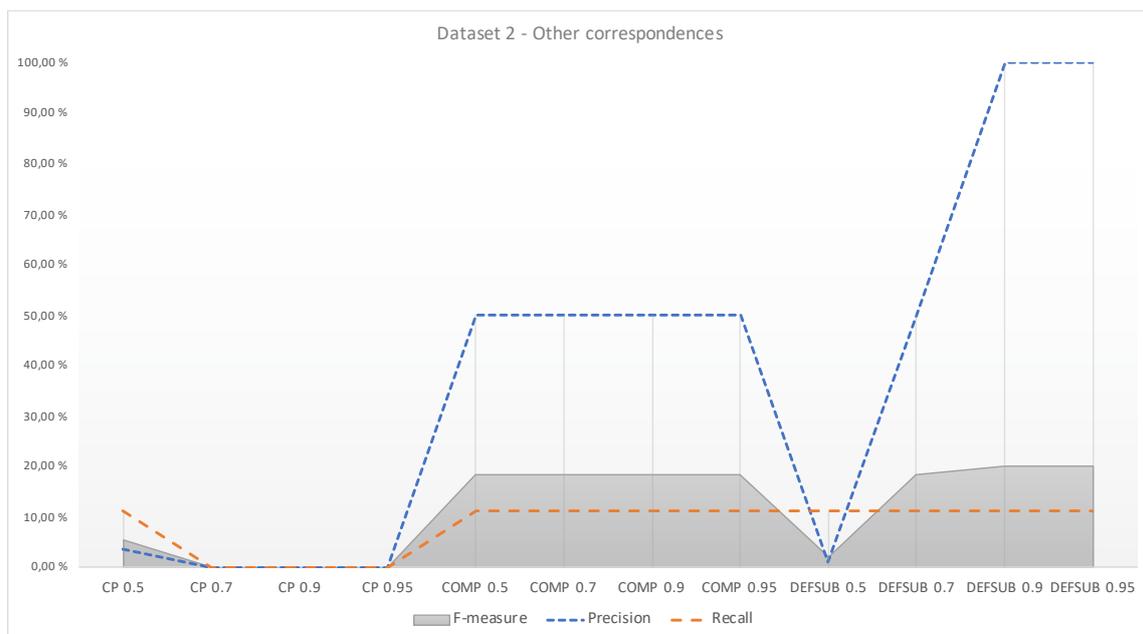


Figure 11. Dataset 2 Matching Results for Other Correspondences

### 4.2.3 Dataset 3 IWXXM METAR – AIRM

#### Equivalence

The reference alignment contains 11 equivalence relations. As Figure 12 shows, the best performing matcher is ISub with confidence 0.9. ISub identifies two true positive correspondences, but misses out on nine, resulting in an F-measure score of around 18 percent. The Definitions Matcher identifies 1 true positive correspondence and 1 false positive one, while the Property Matcher identifies 1 true positive (same as the Definitions Matcher and ISub) and no false positives.

As with dataset 2, there are several complex mappings in this dataset, and the matchers cannot benefit from similarity in concept name, structure or definitions.

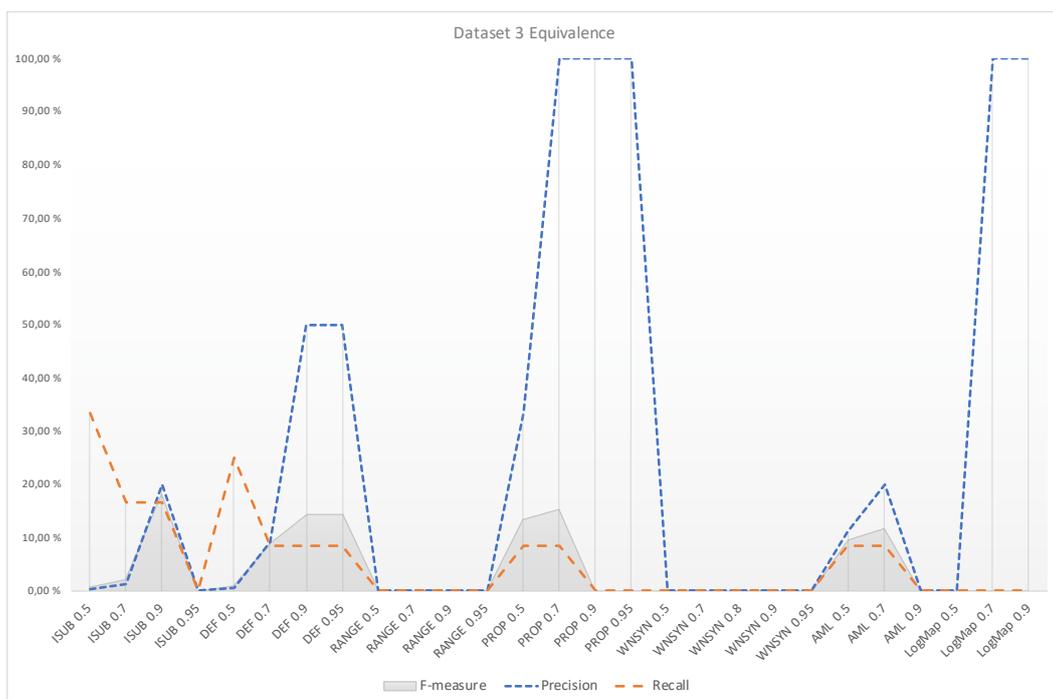


Figure 12. Dataset 3 Equivalence Matching Results

#### Other correspondences

The reference alignment contains 7 correspondences and the only matcher able to identify any true positives is the Compound Matcher (at all confidence thresholds), which identifies the following two:

- *AerodromeSurfaceWind - Wind*
- *AerodromeRunwayVisualRange – RunwayVisualRange*

Here, the use of *endocentric compounds* contributes to the identification of a subsumption relationship. Endocentric compounds consist of a compound *head*, which represent the base meaning of the compound, and one or more *modifiers* that serves to narrow the meaning of the compound as a whole [23]. So, in the first relation identified by the Compound Matcher, *Wind* is identified as the compound head of *AerodromeSurfaceWind*, thus concluding that *AerodromeSurfaceWind* is more restrictive than *Wind*, and *RunwayVisualRange* serves as the compound head of

*AerodromeRunwayVisualRange*, resulting in that *AerodromeRunwayVisualRange* is considered more restrictive than *RunwayVisualRange*.

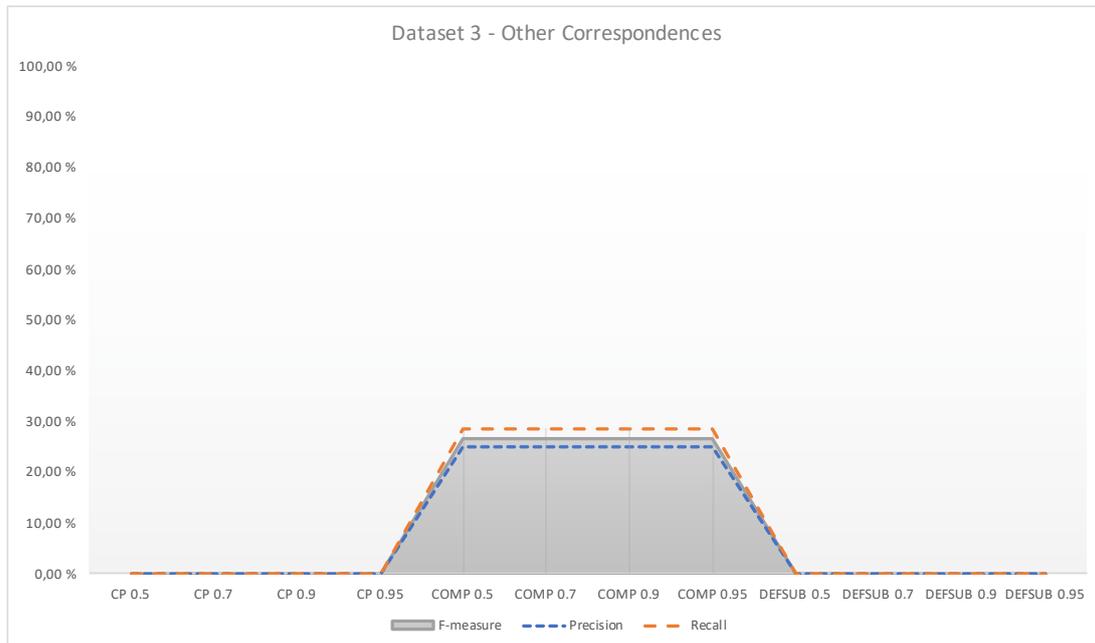


Figure 13. Dataset 3 Matching Results for Other Correspondences

#### 4.2.4 Dataset 4 AIXM Shared – AIRM

The reference alignment contains 10 correspondences. The chart in Figure 14 shows a summary of the matching results. As seen, the Property Matcher obtains the highest F-measure score of the individual matchers with 73.7 percent.

Table 8 presents the reference alignment and a summary of matchers that identified the different correspondences. As can be seen, in many of the correspondences, the concept names are identical, representing an easy task for the string-based matcher ISub.

All remaining correspondences except for one is identified by the other matchers.

Table 8. Reference alignment and identified equivalence correspondences for dataset 4

AIXM – Shared	AIRM	Identified by matcher
OnlineContact	OnlineContact	ISUB, DEF, RANGE, PROP, WNSYN
ContactInformation	ContactInformation	ISUB, DEF, RANGE, PROP, WNSYN
PropertiesWithSchedule	ObjectWithSchedule	RANGE
LightElement	LightElement	ISUB, DEF, WNSYN
PostalAddress	PostalAddress	ISUB, DEF, PROP, WNSYN

AIXM – Shared	AIRM	Identified by matcher
Meteorology	WeatherCondition	None
Timesheet	Timesheet	ISUB, DEF, RANGE, PROP
SpecialDate	SpecialDate	ISUB, PROP, WNSYN
LightElementStatus	LightStatus	RANGE, PROP
TelephoneContact	TelephoneContact	ISUB, DEF, PROP, WNSYN

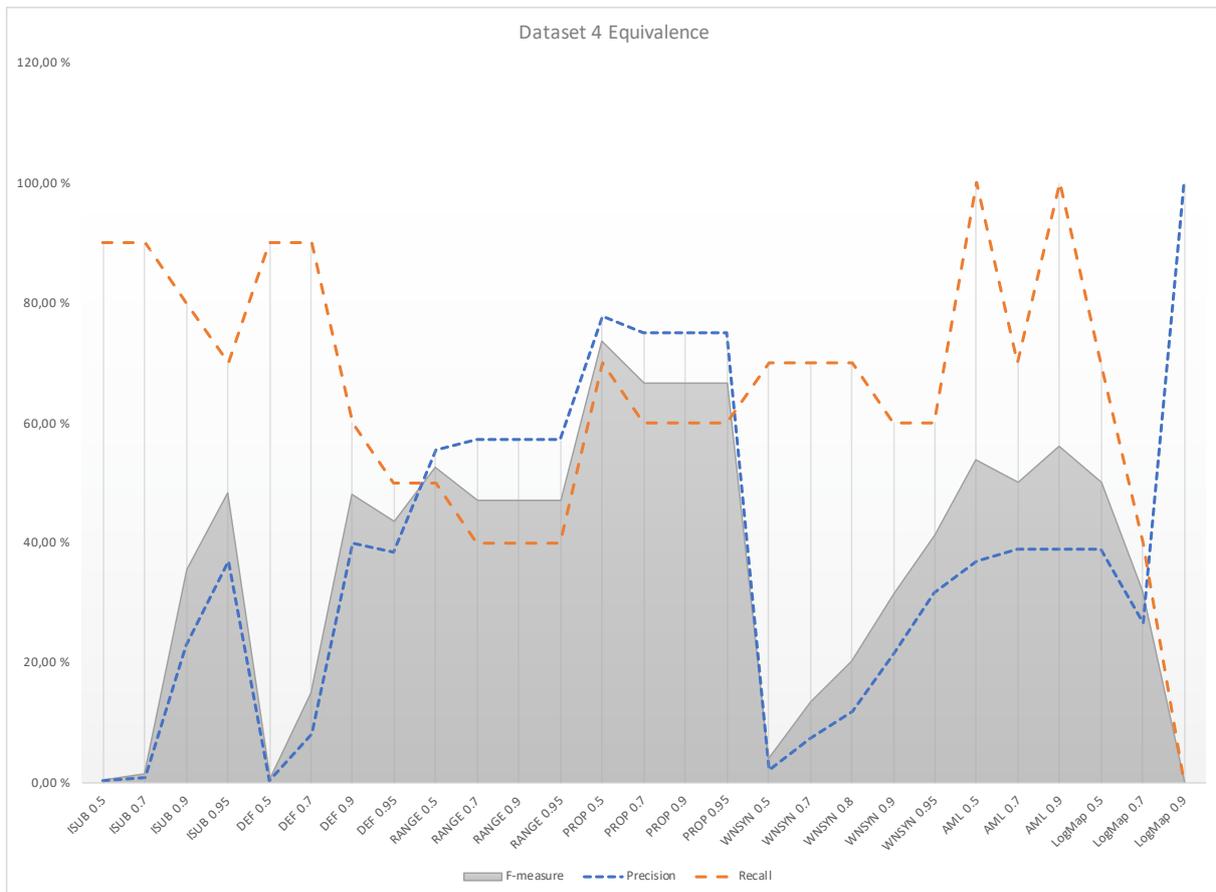


Figure 14. Dataset 4 Equivalence Matching Results

#### 4.2.5 Dataset 5 AIXM Geometry – AIRM

**Equivalence.** The reference alignment contains 4 equivalence correspondences. The only two matchers able to identify any true positive correspondences are the Definitions Matcher at confidence threshold 0.7 which identified 3 of them, and ISub at confidence 0.5 identifying 1. The best F-measure is 31.6 percent obtained by the Definitions Matcher at confidence 0.7. None of the baseline matching systems were able to identify any true positive relations in this dataset.

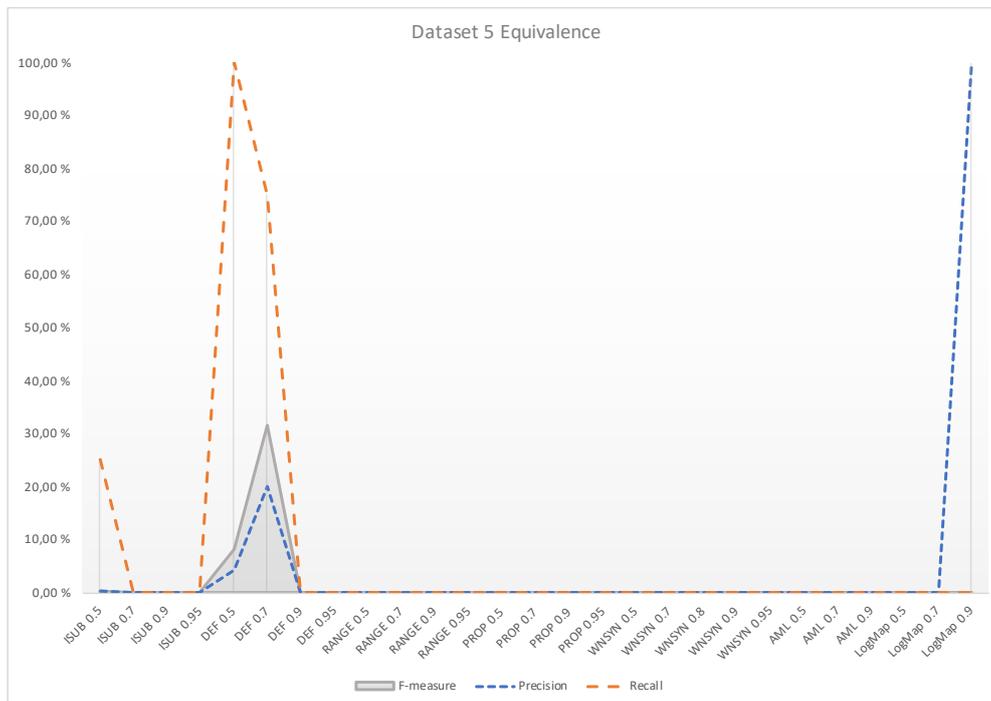


Figure 15. Dataset 5 Equivalence Matching Results

**Other correspondences.** The reference alignment contains two correspondences of which both were captured by the Definitions Subsumption Matcher. At confidence 0.95 this matcher identified these two correct correspondences, and 1 false positive correspondence, while at lower thresholds (0.9, 0.7, and 0.5) the number of false positives increased by lowering the threshold.

Figure 16 shows an example on how the Definitions Subsumption matcher identifies a more restrictive correspondence by a combination of definition size (i.e. number of words) and definition string similarity. Except for the added [An AIXM surface] extension, both definitions are identical. *Surface* is an AIXM concept while *TwoDimensionalSurfaceType* is an AIRM concept. See section 3.2.2 for a more elaborate explanation of the Definitions Subsumption matcher.

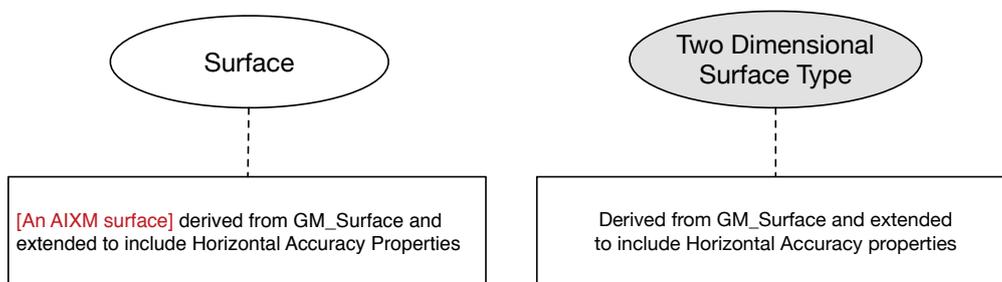


Figure 16. Example on how the Definitions Subsumption Matcher identifies more restrictive correspondence

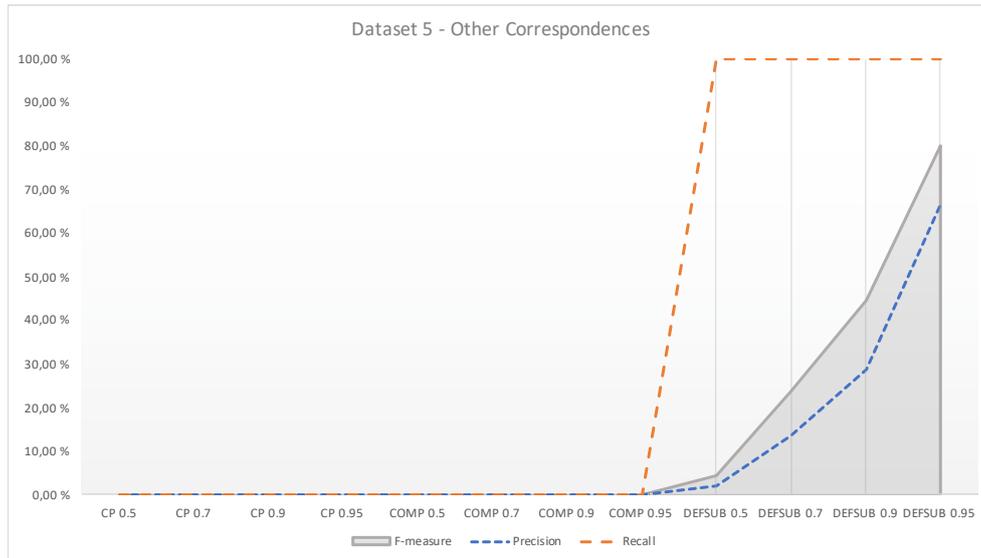


Figure 17. Dataset 5 Matching Results for Other Correspondences

#### 4.2.6 Dataset 6 AIXM Obstacle – AIRM

**Equivalence.** The reference alignment consists of 3 equivalence correspondences. The best F-measure score is obtained by the Property Matcher at confidence level 0.5, which identifies two true positive correspondences, and manages to disregard more false positives than the other matchers. Both ISub (at confidence threshold 0.95), the WordNet Synonym matcher (at confidence threshold 0.95), and the Definitions Matcher (at confidence threshold 0.5 and 0.7) identifies all three correspondences in the reference alignment, hence the high recall values, but also includes more false positives, which has a negative effect on the F-measure. The baseline systems identify all three correspondences, but include many false positives, hence the lower F-measure.

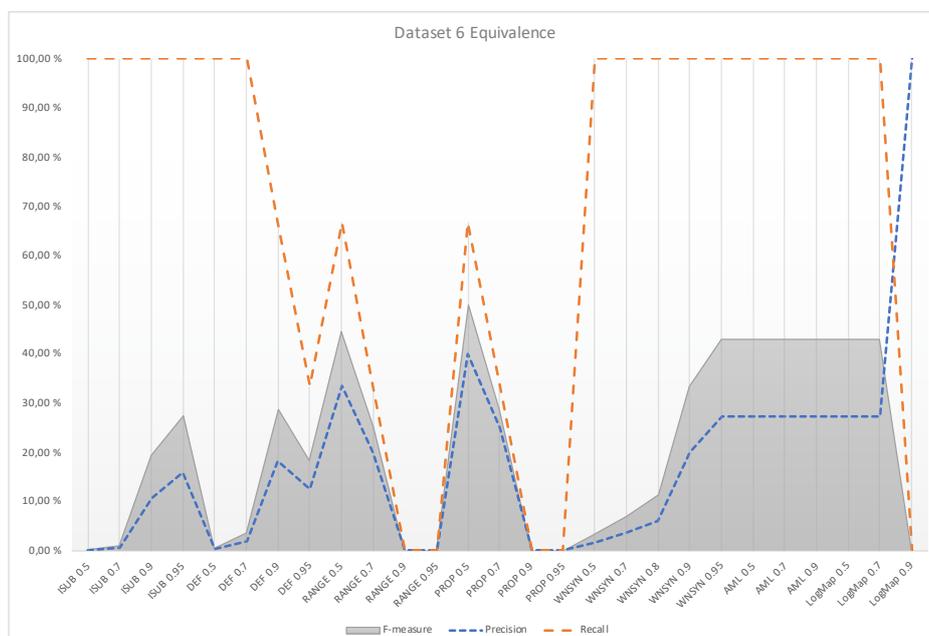


Figure 18. Dataset 6 Equivalence Matching Results

**Other correspondences.** The reference alignment includes two correspondences. As Figure 19 shows, the only matcher able to identify a true positive correspondence in this dataset is the Definitions Subsumption matcher at confidence thresholds 0.5 and 0.7. Both alignments identify one true positive correspondence, but contain a large number of false positives, resulting in a very low F-measure score of just over 1 percent for the best performing matcher (at confidence level 0.7).

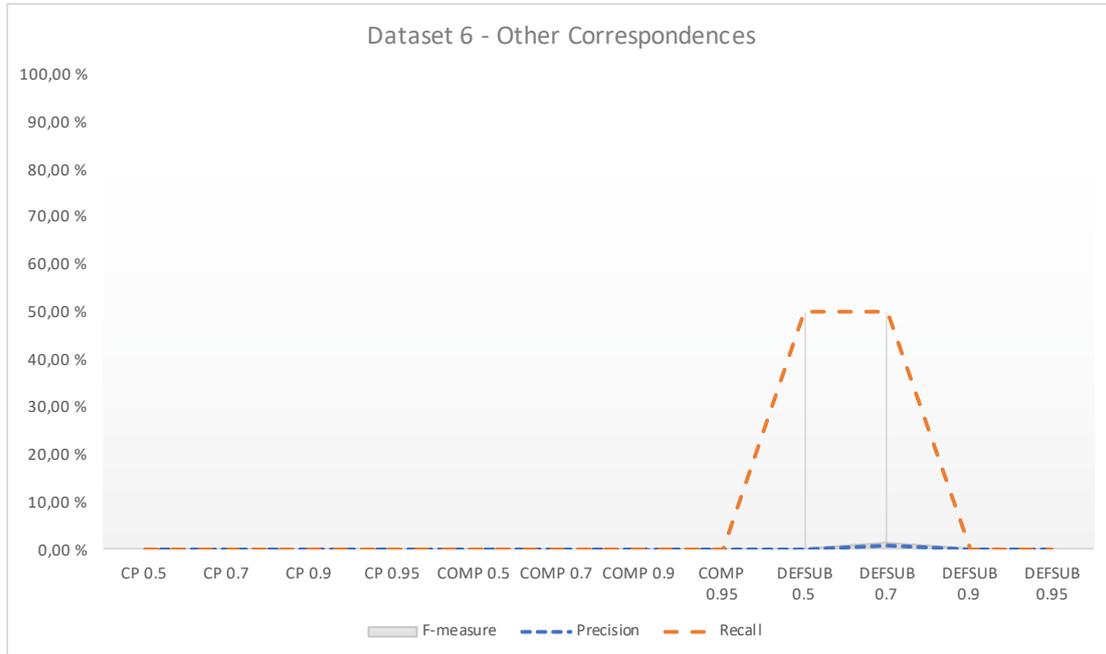


Figure 19. Dataset 6 Matching Results for Other Correspondences

#### 4.2.7 Dataset 7 AIXM Organisation – AIRM

The reference alignment for this dataset contains 5 equivalence correspondences. As Figure 20 illustrates, the best performing matchers are the ISub Matcher at confidence threshold 0.95 and the baseline system AML at confidence 0.5 which both achieves an F-measure of 57.1 percent.

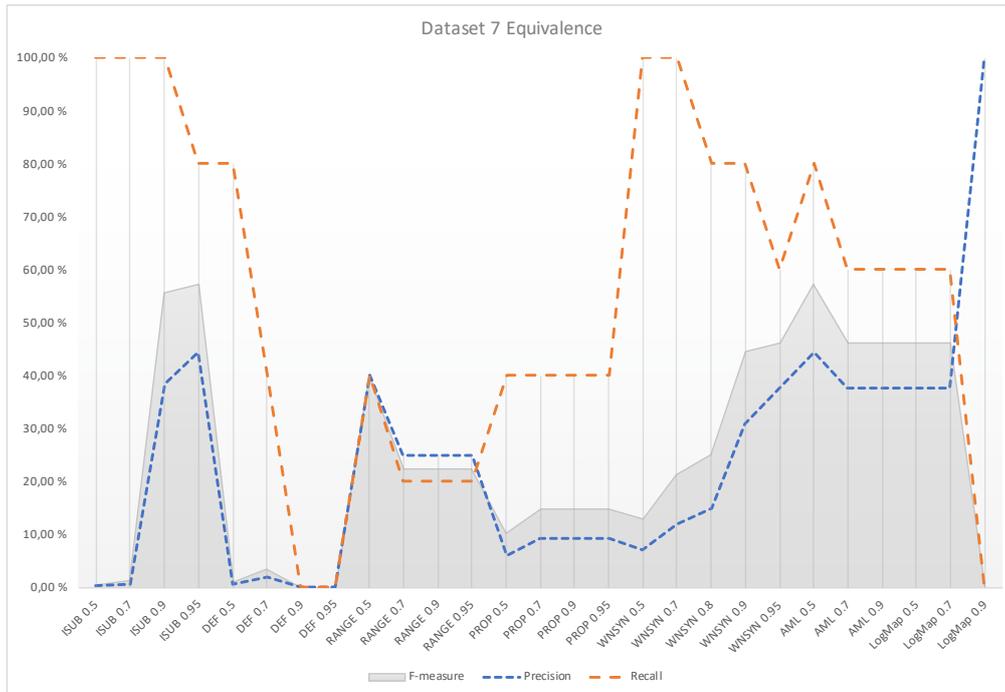


Figure 20. Dataset 7 Equivalence Matching Results

### 4.2.8 Average alignment quality – Equivalence

Figure 21 shows the best performing equivalence matchers in terms of F-measure averaged across all datasets. It shows that the baseline matcher AML achieves the highest F-measure score of 37.3 percent. The best individual matcher implemented in our work is the Property Matcher with a low confidence threshold of 0.5 which obtains an F-measure of 35 percent. At third position is the Range Matcher at confidence 0.5 with an overall F-measure score of 32.1 percent.

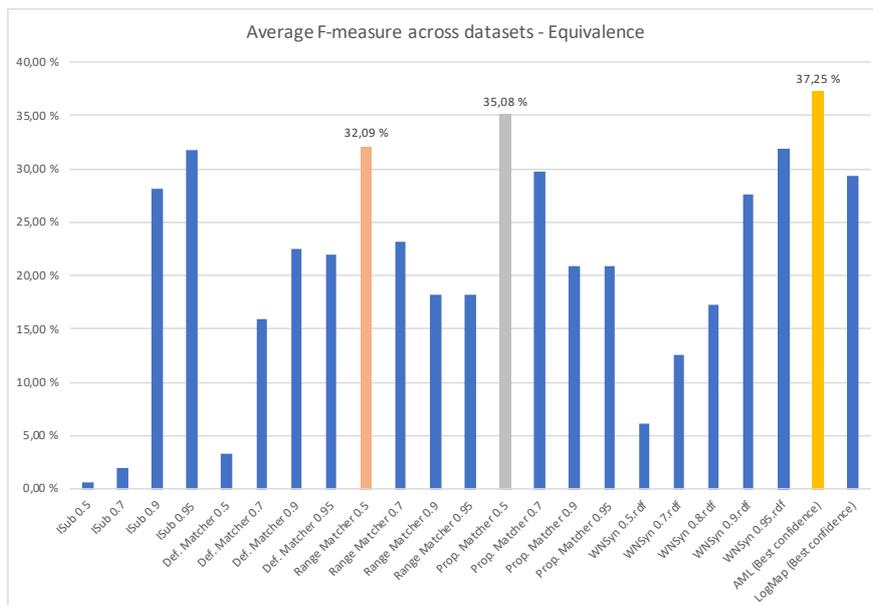


Figure 21. Alignment quality averaged across all datasets

#### 4.2.9 Average alignment quality – other relations

Figure 22 shows the best performing matchers when considering other relations than equivalence. The Definitions Subsumption Matcher obtains an overall F-measure of 20 percent, the same matcher but with confidence threshold 0.9 obtains 12.9 percent, while the Compound Matcher at confidence threshold 0.95 achieves an overall F-measure of 9 percent.

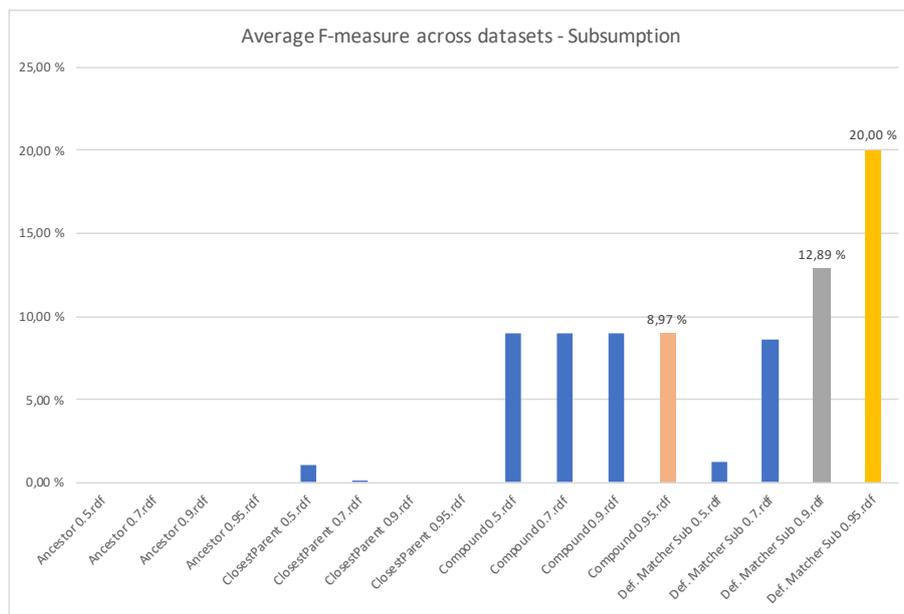


Figure 22. Average alignment quality "other relations" matchers

#### 4.2.10 Alignment combination results

**Equivalence.** We used the 3 best performing matchers presented in chapter 4.2.8 in our combination strategies (see chapter 3.2.3) as this resulted in higher F-measure scores than when combining all alignments. For the equivalence correspondences, the best combination strategy is Autoweight++, followed by SimpleVote. Both these produce higher quality (F-measure) alignments, than the baseline system AML which ranks third.

We used the same combination strategies for the other correspondences experiments also, using the 3 best individual matchers presented in chapter 4.2.9, but none of the combination strategies were able to produce better quality alignments than the individual matchers.

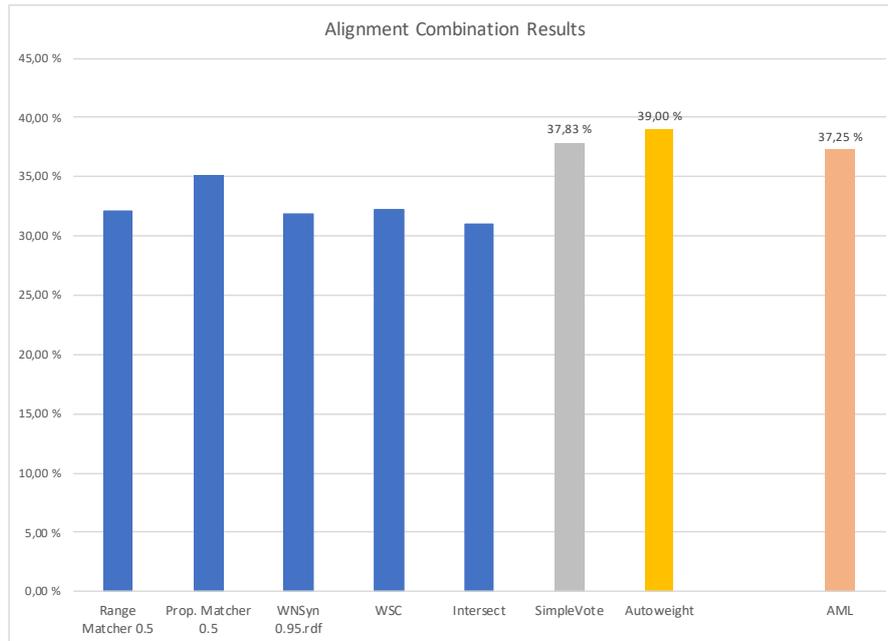


Figure 23. Alignment Combination Results - Equivalence

#### 4.2.11 Conclusions from the experimental evaluation

The general conclusions are that the identification of equivalence relations is far easier than identification of other semantic relations, such as subsumption, even if there is clearly some variability across the datasets. There are several contributing factors to this. First, the reference alignments (as a consequence of being directly transformed from the mapping files) contain subsumption relations that are not necessarily subsumption. As an example, in the AIXM-AIRM mapping file there is a less general (more restrictive) relationship between the AIXM RunwayElement entity and the AIRM RunwayElement entity. The reason why the first is more restrictive than the latter, is in our interpretation that RunwayElement is described in a generic way in AIRM, where it is defined as “A portion of a runway”, and there was thus a need for making the definition more accurate in AIXM. In AIXM RunwayElement is defined as follows: “Runway element may consist of one more polygons not defined as other portions of the runway class”. Secondly, and especially the case when trying to identify such relations between IWXXM and AIRM, it is very difficult to find usable patterns of specialisation in most of the mappings. Even if we in this work have developed matchers that can utilise both terminology patterns, structure patterns and lexical semantics patterns, most specialisation relations specified in the mapping files are not detected.

The quality of the equivalence matching is far better. When comparing against two of the top performing ontology matching systems, AgreementMakerLight and LogMap, the equivalence alignments produced by our matchers produce higher F-measure in all datasets but one. It is important to note though that the iterative experimentation/development have enabled us to tune the matchers towards the ATM context.

Another observation is that the matchers utilising properties as means for inferring class similarity (Property Matcher and Range Matcher) perform better when using low confidence thresholds, while the other matchers perform better at higher confidence thresholds. The explanation is that at lower

confidence, the other matchers produce too many false positives, reducing the precision. Finding a good compromise between precision (i.e. reducing false positives) and recall (i.e. retrieving as many true positives as possible) is the key to good performance of a matching system.

Furthermore, we see that combining the alignments improves the alignment quality in most datasets involving equivalence correspondences. Here, the combination strategies extract complementary true positive correspondences from each individual alignment, and also helps reduce the number of false positive correspondences. For other types of correspondences, combining the alignments hurts the quality, resulting in lower F-measure scores than for the best performing individual matchers.

# 5 User interface for the AIRM Compliance Validator

A command-line user interface has been developed for interacting with the AIRM Compliance Validator. In the following we present some screenshots to illustrate its functionality. The prototype source code is available from github at <https://github.com/sju-best-project/compliancevalidator>

The shown interactive user interface is provided by running *AIRMComplianceValidatorUI.java*

## 5.1 Import ontologies

The entire process starts by importing the two ontologies from which semantic correspondences will be identified. The parsers implemented in the AIRM Compliance Validator will only accept OWL ontologies, and the ontologies have to reside locally on disk, not online. Next, the user is asked to provide a path to a folder where the alignment holding all semantic correspondences will be stored.

```
Starting the AIRM Compliance Validator
Enter path to ontology 1: .files/ontologies/aixm/aixm_airportheairport.owl
Enter path to ontology 2: .files/ontologies/airm/aerodromeinfrastructure.owl
Enter path to folder where alignment file will be stored: ./files/alignments/
```

Figure 24. Import of ontologies to be matched

## 5.2 Match ontologies

Once the ontologies are imported and parsed, the matching of the two imported ontologies is performed with some initial configuration from the user. This includes selecting the desired type of semantic correspondence (equivalence or other semantic correspondence types), selecting matcher(s), and configuring the confidence measure.

### 5.2.1 Select matching strategy

Once the ontologies are imported, the user is asked to select whether the AIRM Compliance Validator should identify equivalence relations or other semantic relations. Afterwards, the user is presented with a list of available matchers and combination strategies (matcher configuration). The sub-menu shown presenting the available matchers depends on whether the user has selected equivalence relations or other semantic relations, see Figure 25Figure 26.

```
Select Equivalence (1) or Other Semantic Relations (2): 1
EQ1: ISub
EQ2: Definitions Matcher
EQ3: Property Matcher
EQ4: Range Matcher
EQ5: WordNet Synonym Matcher
COM1: Weighted Sequential Combination
COM2: Simple Vote Combination
COM3: Autoweight++ Combination
```

Figure 25. Select Matching Strategy - Equivalence

```
Select Equivalence (1) or Other Semantic Relations (2): 2
SUB1: Closest Parent Matcher
SUB2: Compound Matcher
SUB3: Definitions (Subsumption) Matcher
COM1: Weighted Sequential Combination
COM2: Simple Vote Combination
COM3: Autoweight++ Combination
```

Figure 26. Select Matching Strategy - Other Semantic Correspondences

Note that in order to run the Closest Parent Matcher an instance of the Neo4J database has to be installed and running. When Neo4J is running a database to hold the graph representation of the ontologies to be matched is created automatically.

## 5.2.2 Matcher configuration

If the user has selected a combination strategy from the sub-menus, he/she is asked to provide a path to the folder holding the alignments to be combined, see Figure 27.

```
EQ1: ISub
EQ2: Definitions Matcher
EQ3: Property Matcher
EQ4: Range Matcher
EQ5: WordNet Synonym Matcher
COM1: Weighted Sequential Combination
COM2: Simple Vote Combination
COM3: Autoweight++ Combination

Select matcher configuration: COM1

Enter path to folder holding alignments to be combined: ./files/testGUI

Weighted Sequential Combination completed!
```

Figure 27. Selection of folder holding alignments to be combined

If the user has selected an individual matcher, he/she is asked to configure which confidence threshold to be applied for the matcher, see Figure 28.

```
Select matcher configuration: SUB1
Select confidence threshold (decimal value between 0.0 and 1.0): 0.6
Matching ./files/ontologies/aixm/aixm_airportheairport.owl and ./files/ontologies/airm/aerodromeinfrastructure.owl
Closest Parent matcher completed!
```

Figure 28. Configuring the selected matcher

Once the configuration of confidence threshold is done, the matching is executed.

### 5.3 Report identified semantic correspondences

The identified semantic correspondences are presented in an RDF-XML file according to the Alignment Format (see chapter 2.2.4). Figure 29 shows the output from an equivalence matching operation using the XML editor OxygenXML<sup>6</sup>. Each equivalence correspondence is represented in a *map* element, and each map element contains one *cell* element. Within each cell element the two concepts forming the semantic correspondence is represented as *entity1* and *entity2*. The type of semantic correspondence between the two concepts is expressed in the *relation* element. For equivalence correspondences the relation is '=', while specialisation (restriction) which is shown in Figure 30 is specified as '&lt;' (less than). Generalisation would be specified as '&gt;' (or greater than).



Figure 29. Semantic Correspondences in Alignment Format - Equivalence

<sup>6</sup> <https://www.oxygenxml.com/>

```

    ▼ ● map
      ▼ ● Cell
        ● entity1 "http://project-best.eu/owl/iwxxm-mod/metar#AerodromeSurfaceWind"
        ● entity2 "http://www.project-best.eu/owl/airm-mono/airm.owl#Wind"
        ● relation &lt;
        ● measure "http://www.w3.org/2001/XMLSchema#float" 1.0
      ▼ ● map
        ▼ ● Cell
          ● entity1 "http://project-best.eu/owl/iwxxm-mod/metar#RunwayContamination"
          ● entity2 "http://www.project-best.eu/owl/airm-mono/airm.owl#Runway"
          ● relation &lt;
          ● measure "http://www.w3.org/2001/XMLSchema#float" 1.0
      ▼ ● map
        ▼ ● Cell
          ● entity1 "http://project-best.eu/owl/iwxxm-mod/metar#AerodromeRunwayVisualRange"
          ● entity2 "http://www.project-best.eu/owl/airm-mono/airm.owl#RunwayVisualRange"
          ● relation &lt;
          ● measure "http://www.w3.org/2001/XMLSchema#float" 1.0
  
```

Figure 30. Semantic Correspondences in Alignment Format - Other Correspondences

## 6 Conclusions and future work

---

### 6.1 Conclusions

This report has described the development, evaluation, and resulting functionality of the AIRM Compliance Validator, a proof-of-concept application for automatic identification of semantic correspondences between ATM ontologies. The AIRM Compliance Validator offers application support both during the model development when modellers investigate potentially re-usable elements in the reference model, and after completion of the model, when its compliance with the AIRM has to be verified and maintained for governance purposes.

The AIRM Compliance Validator includes 9 different matching algorithms that utilise different features of the ontologies to be matched in order to identify equivalence relations and other semantic relations. The algorithms have been developed based on terminological, structural and lexical analysis of the input ontologies which is performed automatically by an ontology profiling component.

From an experimental evaluation involving AIRM, AIXM and IWWXM in seven different datasets, we have learned that the AIRM Compliance Validator does a good job of identifying equivalence relations. Compared with two state-of-the-art ontology matching systems, AgreementMakerLight (AML) and LogMap, the AIRM Compliance Validator is able to identify more true positive relations and omit more false positive relations in most of the datasets. The best individual matching algorithm, named Property Matcher, exploits similarity in properties to infer class equivalence and obtains an F-measure of 35 percent over all datasets including equivalence relations.

The evaluation also shows that by combining individual alignments improve the quality. The best combination strategy, Autoweight++, obtains an average F-measure of 39 percent, hence an improvement of 4 percent compared to the best individual matcher.

The evaluation also shows that the automatic identification of other types of relations is more challenging. There are several reasons for this. First, it is very difficult to find commonalities in concept naming, natural language definitions, and the structural properties that could suggest “other” semantic relations between concepts. Second, the matchers are tuned towards specialisation relations, while the reference alignments (and the mapping files used as source for them) include a variety of different semantic relations (for example part-whole relations). Third, several of the relations are complex relations where for example one concept in the first ontology is mapped to several concepts in the second ontology. This is normally a challenging case for ontology matching which typically assumes a 1-1 correspondence between concepts.

In general, we see that the terminology used in the AIXM model is much closer to that of AIRM than the terminology used in the IWWXM model. In order to promote semantic interoperability different information models in ATM should try to align their terminology to the AIRM reference model as this would prevent interoperability barriers and foster re-use.

A basic command-line user interface has been developed that enables users to interact with the AIRM Compliance Validator. All source code of the AIRM Compliance Validator is published on github for others to use and extend.

## 6.2 Further work

This work has focused on the L1 – AIRM Ready compliance level. That means that semantic relations between concepts (classes) are identified, but not between properties. Further work should include implement matching algorithms that automatically identify property relations as well.

Ontology matching systems often use external resources to facilitate identification of semantic relations. In this work we have employed WordNet as an external resource, but other more domain-specific sources could possibly enhance the matching results. One such resource for the aviation domain is Skybrary<sup>7</sup>, a wiki that contains loads of domain knowledge related to aviation and ATM. Investigating methods on how a resource such as Skybrary could be utilised to support identification of semantic relations is an interesting further work item.

The “other relations” category includes other semantic relations than specialisation. Analysing the other types of semantic relations involved and finding techniques for their identification could lead to more precise matching results for this category. One example is part-whole (meronymy) relations. Investigating patterns in names, definitions and structure that could help reveal part-whole relations (and other possible relations) and distinguish them from equivalence and specialisation relations could lead to better and more accurate alignments.

Scalability is not considered in this work, but is an important quality to look at, especially when ontologies are as large as the AIRM ontology (counting over 3000 entities altogether). Some of the matchers required significant run-time, which probably could be substantially reduced by performing a thorough scalability analysis.

---

<sup>7</sup> [https://www.skybrary.aero/index.php/Main\\_Page](https://www.skybrary.aero/index.php/Main_Page)

## 7 References

---

- [1] S. Wilson, S. Keller, G. Marrazzo, and R. Suzic, “AIRM Compliance Framework,” 2015.
- [2] G. Marrazzo, “AIRM Compliance Handbook,” 2015.
- [3] S. Wilson, A. W. Tell, U. Larsson, R. Suzic, S. Keller, and G. Marrazzo, “AIRM Foundation: Rulebook (v4.1.0),” 2016.
- [4] S. Wilson, “EUROCONTROL Specification for SWIM Information Definition version 1.0,” Brussels, Belgium, 2017.
- [5] M. Ehrig, S. Staab, and Y. Sure, “Bootstrapping ontology alignment methods with APFEL,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3729 LNCS, pp. 186–200, 2005.
- [6] P. Hitzler, M. Krotzsch, B. Parsia, P. F. Patel-Schneider, and S. Rudolph, “OWL 2 Web Ontology Language Primer (Second Edition),” 2012. [Online]. Available: <https://www.w3.org/TR/owl2-primer/>. [Accessed: 12-May-2017].
- [7] S. B. Abbes, T. Meilender, and M. D’Aquin, “Characterizing modular ontologies,” in *Conference on Formal Ontologies in Information Systems-FOIS*, 2012.
- [8] S. Wilson, R. Suzic, and S. Van der Stricht, “The SESAR ATM information reference model within the new ATM system,” in *2014 Integrated Communications, Navigation and Surveillance Conference (ICNS) Conference Proceedings*, 2014.
- [9] A. Vennesland, B. Neumayr, C. Schuetz, and A. Savulov, “D1.1 Experimental ontology modules formalising concept definition of ATM data,” 2017.
- [10] A. Vennesland, E. Gringinger, and A. Kocsis, “D5.2 Ontology Modularisation Guidelines,” 2018.
- [11] G. Acampora, V. Loia, S. Salerno, and A. Vitiello, “A hybrid evolutionary approach for solving the ontology alignment problem,” *Int. J. Intell. Syst.*, vol. 27, no. 3, pp. 189–216, 2012.
- [12] J. Euzenat, “Algebras of ontology alignment relations,” in *Proceedings of the International Semantic Web Conference 2008*, 2008, pp. 387–402.
- [13] J. David, J. Euzenat, F. Scharffe, and C. T. Dos Santos, “The alignment API 4.0,” *Semant. Web*, vol. 2, no. 1, pp. 3–10, 2011.
- [14] M. Horridge and S. Bechhofer, “The OWL API: A Java API for OWL Ontologies,” *Semant. Web J.*, vol. 2, no. 1, pp. 11–21, 2011.
- [15] J. David, J. Euzenat, F. Scharffe, and C. T. Dos Santos, “The Alignment API 4.0,” *Semant. Web*, vol. 2, no. 1, pp. 3–10, 2010.
- [16] C. Fellbaum, *WordNet: An Electronical Lexical Database*. Cambridge: MIT Press, 1998.
- [17] I. F. Cruz, A. Fabiani, F. Caimi, C. Stroe, and M. Palmonari, “Automatic configuration selection using ontology matching task profiling,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7295 LNCS, pp. 179–194, 2012.
- [18] S. Tartir, I. Arpinar, M. Moore, a Sheth, and B. Aleman-Meza, “OntoQA: Metric-Based Ontology Quality Analysis,” *IEEE Work. Knowl. Acquis. from Distrib. Auton. Semant. Heterog. Data Knowl. Sources*, pp. 45–53, 2005.
- [19] I. F. Cruz, A. Fabiani, F. Caimi, C. Stroe, and M. Palmonari, “Automatic configuration selection using ontology matching task profiling,” 2012.
- [20] S. Stoilos, Giorgos and Stamou, Giorgos and Kollias, “A string metric for ontology alignment,” in

- Proceeding of the International Semantic Web Conference 2005*, 2005, pp. 624–637.
- [21] W. E. Winkler, “The state of record linkage and current research problems,” in *Statistical Research Division, US Census Bureau*, 1999.
  - [22] P. Jaccard, “Distribution de la flore alpine dans la Bassin de Dranses et dans quelques regions voisines,” *Bull. del la Société Vaudoisedes Sci. Nat.*, 1901.
  - [23] P. Arnold and E. Rahm, “Enriching ontology mappings with semantic relations,” *Data Knowl. Eng.*, vol. 93, pp. 1–18, 2014.
  - [24] M. Gulić, B. Vrdoljak, and M. Banek, “CroMatcher: An ontology matching system based on automated weighted aggregation and iterative final alignment,” *Web Semant. Sci. Serv. Agents World Wide Web*, no. 41, 2016.
  - [25] J. Euzenat, “Semantic Precision and Recall for Ontology Alignment Evaluation,” in *Proc. 20th International Joint Conference on Artificial Intelligence (IJCAI)*, 2007, pp. 348–353.
  - [26] D. Faria, C. Pesquita, E. Santos, M. Palmonari, I. F. Cruz, and F. M. Couto, “The AgreementMakerLight ontology matching system,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8185 LNCS, pp. 527–541, 2013.
  - [27] “Ontology Alignment Evaluation Initiative.”
  - [28] E. Ruiz-Jimenez and B. C. Grau, “LogMap : Logic-Based and Scalable Ontology Matching,” in *ISWC 2011*, 2011, pp. 273–288.
  - [29] S. Wilson, “AIRM Glossary,” 2016.

# Annex A: Requirements from SWIM Compliance Specifications

Table A-1. Requirements from AIRM Compliance Framework relevant for the development of the AIRM Compliance Validator

Requirement	Description of requirement	Relevance for AIRM Compliance Validator
<b>ACF-Req-4</b>	The definition of each source element shall be written out completely, or in the form of an unambiguous reference to an explicit definition in a referenced standard.	Relates to that the AIRM Compliance Validator can exploit entity definitions in the matching process between an OuA and AIRM.
<b>ACF-Req-6</b>	Whenever an Object under Assessment element uses a definition from an underlying standard, the reference should be provided.	Relates to how the AIRM Compliance Validator should manage similarity measurements between definitions of OuA and AIRM. Could indicate that the syntactic similarity between semantically similar entities are uniform.
<b>ACF-Req-10</b>	The (AIRM) Compliance Report should follow the template described in Appendix A of this document (an MS Office Word template)	Relates to how the output of the AIRM Compliance Validator should be.
<b>ACF-Req-11</b>	The AIRM elements providing definitions shall be identified uniquely using: <ul style="list-style-type: none"> <li>• Either the element’s UUID (or a model representation thereof, e.g. the UML element associated with the UUID), or</li> <li>• The element’s URN</li> </ul>	Relates to unique identification of ontology entities associated with the OuA and ontology entities associated with the AIRM ontologies (monolithic and/or ontology modules). In the ontologies and modules used in the BEST project we use unique URIs for all entities (e.g. ‘http://www.project-best.eu/owl/airm-mod/aerodromeinfrastructure.owl#Marking’)
<b>ACF-Req-13</b>	In order to claim AIRM Compliance Level 1, each Entity of the Object under Assessment shall have a definition. Each atomic information or data elements of the Object under Assessment shall be part of a definition-bearing entity.	Relates to that the AIRM Compliance Validator can exploit entity definitions in the matching process between an OuA and AIRM when “higher-level” entities (i.e. classes) are matched (see chapter 0.).
<b>ACF-Req-14</b>	In order to claim of AIRM Compliance Level 2, each entity and Property of the Object under Assessment shall have a definition.	Relates to that the AIRM Compliance Validator can exploit entity definitions in the matching process between an OuA and AIRM when also object properties are matched (see chapter 0.).
<b>ACF-Req-15</b>	In order to claim AIRM Compliance Level 3, each Entity, Property, Data Type of the Object under Assessment shall have a definition.	Relates to that the AIRM Compliance Validator can exploit entity definitions in the matching process between an OuA and AIRM when also data properties and their type specifications are matched (see chapter 0.).

Requirement	Description of requirement	Relevance for AIRM Compliance Validator
<b>ACF-Req-16</b>	In order to claim of AIRM Compliance Level 1, each Object under Assessment's Entity shall have documented semantic correspondence towards the stated version of the AIRM or map to an AIRM Compliance construct.	See also AFH-Rule 60 in section 2.1.3. Relates to how the definition associated with the entities of the OuA into the operational language of AIRM.
<b>ACF-Req-17</b>	In order to claim of AIRM Compliance Level 2, each Object under Assessment's Entity and Property shall have documented semantic correspondence towards the stated version of the AIRM or map to an AIRM Compliance construct.	See also AFH-Rule 60 in section 2.1.3. Relates to how the definition associated with the entities of the OuA into the operational and data language of AIRM.
<b>ACF-Req-18</b>	In order to claim of AIRM Compliance Level 3, each Object under Assessment's Entity, Property, Data Type and Constraints/Business Rule shall have documented semantic correspondence towards the stated version of the AIRM or map to an AIRM Compliance construct.	See also AFH-Rule 60 in section 2.1.3. The ontologies do not preserve the business rules (e.g. that constraints the length of a string value). AIRM Principle-17 in the AIRM Foundation Rulebook states that business rules can be restricted, but not extended. Need to see to what extent and how we deal with this.
<b>ACF-Req-19</b>	The compliance assessment shall result in a unique statement qualifying the Object under Assessment as a whole relative to the version of AIRM products mentioned in the compliance evidence. It shall take one of the values: <ul style="list-style-type: none"> <li>• "Ready/Compatible/Compliant" if the requirements for the level as stated above are fulfilled and there is no dependency on a Change Request not yet approved by the AIRM CCB</li> <li>• "Provisionally Ready/Compatible/Compliant" if the requirements have been met but there are dependencies on a Change Request not yet approved by the AIRM CCB</li> <li>• "Not Ready/Compatible/Compliant" if at least one requirement has not been met.</li> </ul>	Relevant for how the AIRM Compliance Validator should present the results from the ontology matching operation.

Rule or Principle	Description	Relevance for AIRM Compliance Validator
<b>AFH-Rule 116</b>	<p>A data or information construct is considered to be in semantic correspondence with the AIRM if one of the following conditions holds:</p> <ul style="list-style-type: none"> <li>• The definition of the construct is an exact copy of the definition of a specific AIRM element, or it is syntactically equal, or is rewritten or is specialised as described in AIRM_Rule 60.</li> <li>• It can be demonstrated that the definition of the construct can be decomposed into several elementary concepts, each corresponding to an AIRM element as per previous bullet. This decomposition must be comprehensive, i.e. cover all parts of the definition.</li> </ul>	<p>This is an overall rule stating what it requires for a definition of an OuA's element to be in semantic correspondence with AIRM.</p>
<b>AFH-Rule 60</b>	<p>The 'Definition:Adapted' AIRM::TaggedValue shall be completed in order to indicate the level of semantic correspondence with the source definition. The possible values are:</p> <ul style="list-style-type: none"> <li>• ExactCopy: Definition of source and target are exact copy of each other.</li> <li>• SyntacticallyEqual: Syntax corrections (grammar, spelling)</li> <li>• Rewritten: The definition has been rewritten for improved quality. The meaning is the same, i.e. the definition still describes exactly the same entity as the target definition.</li> <li>• Specialised: Source definition is a special case of the target definition.</li> <li>• Generalised: Source definition is a generalised case of the target definition.</li> </ul>	<p>Gives an indication of how the definition of an OuA's element definition relates to an equivalent AIRM element definition. The 'ExactCopy' level would be easy to identify using a string similarity technique. The other levels probably need some human involvement.</p>
<b>AFH-Rule 17</b>	<p>Any abbreviation or acronym for a model element's name shall be represented in an AIRM::TaggedValue 'Definition:Abbreviation'</p>	<p>As element names of the OuA are not required to be syntactically similar to a semantically equivalent AIRM element, possible acronyms or abbreviations must be checked.</p>

Rule or Principle	Description	Relevance for AIRM Compliance Validator
<b>AFH-Rule 62</b>	Any synonyms for a model element's name shall be represented as a comma separated list in an AIRM::TaggedValue 'Definition:Abbreviation'.	Investigating potential synonyms may help in identifying semantically equivalent classes, properties or individuals between the OuA and AIRM.
<b>AFH-Rule 108</b>	The AIRM Glossary shall contain a textual representation of terms and definitions used within the AIRM Information Model and the AIRM Consolidated Logical Data Model.	The AIRM Glossary [29] is a dictionary of definitions captured from the elements of the AIRM UML model. Can possibly be used as a look-up source for verifying that definitions used in an OuA comply with AIRM definitions.
<b>AFH-Rule 55</b>	The upper bound of the multiplicity specified in a derived model shall be lower or equal to the upper bound of the multiplicity and greater or equal to the lower bound of the multiplicity specified in the AIRM.	The AIRM Compliance Validator need to ensure that the cardinality used in the OuA and AIRM is in accordance with this rule.
<b>AFH-Rule 56</b>	The lower bound of the multiplicity specified in a derived model shall be greater or equal to the lower bound of the multiplicity and lower or equal to the upper bound of the multiplicity specified in the AIRM.	The AIRM Compliance Validator need to ensure that the cardinality used in the OuA and AIRM is in

Rule or Principle	Description	Relevance for AIRM Compliance Validator
		accordance with this rule.
<b>AFH-Rule 59</b>	A derived model shall not use an AIRM term with a conflicting definition.	If a derived model does not use the original AIRM definition, a reference to the AIRM definition needs to be provided. The AIRM Compliance Validator must compare definitions used by the OuA (possibly using the AIRM Glossary). If they do not match, it should check if a referenced definition is used.
<b>AFH-Principle 12</b>	Derivation of AIRM works by restriction. Therefore: <ul style="list-style-type: none"> <li>• Any additional model elements of an AIRM Derived Model, assumed to be within the scope of AIRM, should be mapped to the “AIRM_Change_Request” construct in the AIRM compliance report.</li> <li>• Any additional model elements of an AIRM Derived Model, assumed to be outside the scope of AIRM, should be mapped to the “AIRM_OutOfScope” construct in the AIRM compliance report.</li> </ul>	Need to investigate whether it is possible to categorise OuA elements that do not match AIRM elements according to this principle. Probably requires human involvement, but perhaps the AIRM Compliance Validator can assist the end-user in some way...

Rule or Principle	Description	Relevance for AIRM Compliance Validator
<b>AFH-Rule 119</b>	If the AIRM has missing model elements hindering the development of a mapping, an AIRM Change Request shall be raised. The object under assessment model element shall reference to the AIRM construct “AIRM_Change_Request”, and the number of the Change Request shall be recorded in the mapping.	Related to comment to Principle 12 above.
<b>AFH-Rule 120</b>	If a model element of object under assessment is found to have an error which prevents its correct mapping to the AIRM, it shall be mapped to a new construct named “OuA_Issue”. This construct shall be given a description explaining what the error or gap is, and how it is proposed to be handled.	Related to comment to Principle 12 above.
<b>AFH-Principle 16</b>	A derived model can further restrict a relationship. This means it is possible to move from what is a simple relationship in the AIRM to a composition relationship.	The AIRM Compliance Validator need a rule set to check allowed relationship restrictions. This means that we in the ontologies need to distinguish an object property transformed from a normal association in UML from an object property transformed from an aggregation or composition relationship in UML.

Rule or Principle	Description	Relevance for AIRM Compliance Validator
<b>AFH-Principle 19</b>	In a derived model, AIRM codelists: <ul style="list-style-type: none"> <li>• May remain as codelists; or</li> <li>• May be converted to enumerations (which can be seen as a restricted codelist); or</li> <li>• May be converted to a series of classes.</li> </ul>	Codelists are represented as classes both in the AIRM ontologies (monolithic and modules) and in the ontology modules created from the exchange models. Their values are represented as individuals. Do the values of the codelists need to be compliant as well?
<b>AFH-Principle 20</b>	A derived model may convert an attribute to a role name or vice versa. This means: <ul style="list-style-type: none"> <li>• A property modelled as an UML attribute in the AIRM may be converted into a property modelled as a role, with a complex “constructed” type.</li> <li>• A property modelled as a role name in the AIRM may be converted into an attribute (e.g. if multiplicity is restricted to 1..1).</li> </ul>	Attributes with complex data types are converted to object properties in the ontologies, and thus treated similarly as associations with role names, so this should not be a problem for the AIRM Compliance Validator.
<b>AFH-Principle 23</b>	To facilitate the AIRM compliance assessment process each AIRM model element has a globally unique name. This unique name is defined according to the Uniform Resource Name (URN) standard.	The ontologies transformed from the AIRM UML model do not include the URNs, but contains a unique URI/IRI. We could if needed include the full URN for an entity as an annotation?



Table A-3: Requirements from Specification for SWIM Information Definition relevant for the development of the AIRM Compliance Validator

Rule or Principle	Description	Relevance for AIRM Compliance Validator
<b>SWIM-INFO-01</b>	Exchanged information shall be documented in an information definition.	Relates to that the AIRM Compliance Validator can exploit entity definitions in the matching process between an Information Definition (OuA) and AIRM.
<b>SWIM-INFO-09</b>	If an information definition contains a concept with the same name as an AIRM concept or a synonym from the AIRM concept's list of synonyms, it shall preserve the meaning of the AIRM concept.	Suggests that even if the concept names are different, definitions will reveal semantic equivalence.
<b>SWIM-INFO-013</b>	An information definition shall document a semantic correspondence for each of its concepts.	This poses required functionality of the AIRM Compliance Validator.

Rule or Principle	Description	Relevance for AIRM Compliance Validator
<b>SWIM-INFO-014</b>	<p>A semantic correspondence shall be:</p> <ul style="list-style-type: none"> <li>• A mapping from a concept in the information definition to a concept or concepts in the AIRM; or</li> <li>• A declaration that the concept in the information definition is out-of-scope of the AIRM; or</li> <li>• A reference to a change request for the AIRM that intends to change the AIRM to cover the concept from the information definition; or</li> <li>• A declaration that no semantic correspondence has been established for the concept.</li> </ul>	<p>Even if the first point represents the target objective of the AIRM Compliance Validator, the AIRM Compliance Validator can also help contradict the other points.</p>
<b>SWIM-INFO-016</b>	<p>The mapping of an information concept shall contain a trace from the information concept in the information definition to the AIRM concept that has an equivalent or wider meaning.</p>	<p>A key requirement related to how semantic correspondences computed by the AIRM Compliance Validator should be presented.</p>
<b>SWIM-INFO-018</b>	<p>The mapping of a concept to an AIRM concept that has a wider meaning shall contain additional traces to AIRM concepts to fully describe the narrowing of the concept being mapped.</p>	<p>In addition to providing traces to equivalent concepts, the AIRM Compliance Validator should also have functionality to identify narrower (and wider) semantic meaning.</p>

The BEST consortium:

<p>SINTEF</p>	
<p>Frequentis AG</p>	
<p>Johannes Kepler Universität (JKU)  Linz</p>	
<p>SLOT Consulting</p>	
<p>EUROCONTROL</p>	