


How to Make Bespoke Experiments FAIR: Modular Dynamic Semantic Digital Twin and Open Source Information Infrastructure

Manuel Rexer  ¹

Nils Preuß  ¹

Sebastian Neumeier  ¹

Peter F. Pelz  ¹


1. Chair of Fluid Systems, Technische Universität Darmstadt, Darmstadt.



Date Submitted:

2024-05-28

Licenses:

This article is licensed under: 

Keywords:

FAIR, linked data, modular test environment, information model, experimental data, information infrastructure

Data availability:

Developed information models can be found here:

<https://git.rwth-aachen.de/fst-tuda/public/metadata>

Software availability:

The developed software and their sources are listed in table 3

Abstract.

In this study, we apply the FAIR principles to enhance data management within a modular test environment. By focusing on experimental data collected with various measuring equipment, we develop and implement tailored information models of physical objects used in the experiments. These models are based on the Resource Description Framework (RDF) and ontologies. Our objectives are to improve data searchability and usability, ensure data traceability, and facilitate comparisons across studies. The practical application of these models results in semantically enriched, detailed digital representations of physical objects, demonstrating significant advancements in data processing efficiency and metadata management reliability. By integrating persistent identifiers to link real-world and digital descriptions, along with standardized vocabularies, we address challenges related to data interoperability and reusability in scientific research.

This paper highlights the benefits of adopting FAIR principles and RDF for linked data proposing potential expansions for broader experimental applications., Our approach aims to accelerate innovation and enhance the scientific community's ability to manage complex datasets effectively.

1 Introduction

2 In scientific research, effective data management is key, especially when dealing with experi-
3 mental data. The increasing volume and complexity of data collected in experimental settings
4 demand rigorous methodologies to ensure that such data remains findable, accessible, interoper-
5 erable, and reusable (FAIR). These principles, established by Wilkinson et al. [1], are crucial
6 for enhancing the transparency, reproducibility, and utilisation of research data across various
7 scientific disciplines.

8 The primary aim of this research is twofold: to develop methodologies that make extensive
9 datasets not only searchable and uniformly usable but also traceable and comparable across

10 different studies. This is essential for building upon existing research without redundant experi-
11 ments, thereby accelerating scientific discovery and innovation. Our approach involves a detailed
12 examination of the test environment, which includes a wide array of measuring equipment and
13 units under test. The reliability of data processing and the precision in uncertainty quantification
14 heavily rely on our ability to thoroughly document and manage both raw data and its metadata.
15 The challenge in this context lies in effectively mapping all relevant information about the
16 experiment, including its components and physical objects, while linking this information to the
17 experimentally obtained data. To address this, it is essential to generate a digital representation
18 of the objects used and ensure its availability for the measurements.

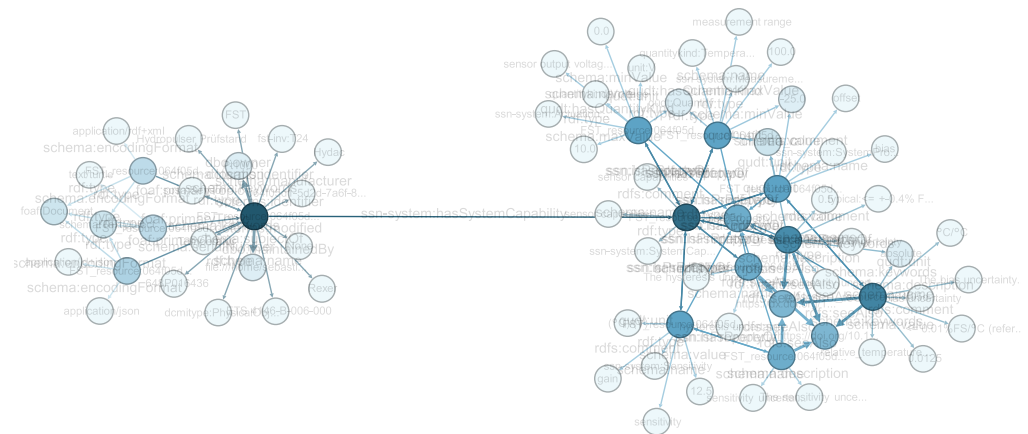


Figure 1: Graph of a digital data sheet of a sensor based on the developed information model. The colouring represents the connectivity (number of connected edges) of the different nodes where darker colours represent a higher, and lighter colours a lower connectivity. The data inside the graph is vaguely indicated by the faint text labels.

19 Using three key use cases from our modular test environment, we outline specific requirements
20 for effective data and metadata management. This paper presents an overview of current advance-
21 ments, technologies, and methods for making metadata FAIR. To meet these requirements, we
22 develop information models and implement a robust working environment to provide and easily
23 access the necessary information. Figure 1 shows an example for a populated information model,
24 that results in a semantically enhanced, detailed digital data set of a real-world object. Since the
25 data set closely describes the traits of the physical object it can be seen as a digital depiction or a
26 digital twin. The infrastructure for providing this information is based on open source resources.
27 We demonstrate practical benefits and improved efficiencies in data management through the
28 utilization of FAIR principles and our implementation.
29 Ultimately, this research exemplifies the broader applicability and significant advantages of
30 adopting FAIR principles within experimental research frameworks, potentially guiding future
31 utilization in similar settings.

32 2 Application Use Case and Requirements

33 In engineering researchers are involved with experimental test setups consisting of a large number
34 of sensors and actuators connected and interfaced with digital data acquisition hardware and
35 software. Depending on the research method and the research topic, these experiments can
36 either be highly individualised and thus designed to answer exactly one question, or they can be
37 universal test environments characterised by the fact that different questions and setups can be
38 answered in a short time.

39 This paper deals with the latter type. It focuses on two particular challenges related to (meta)data
40 management. Firstly, it is essential that the metadata can be captured as easily and quickly as
41 possible, since the test bench is frequently reconfigured. Secondly, it is particularly important to
42 correctly record the setup and the components used, as it is no longer possible to manually check
43 the setup and compare it with the measured data once the test bench has been reconfigured.

44 The following is a description of such a test environment, where various dynamic and quasi-static
45 tests are carried out on mechanical components that are mainly chassis components. From this,
46 requirements on the metadata management are derived.

47 2.1 Test Environment

48 The considered uni-axial servo hydraulic test rig¹ is a modular test rig, where several different
49 units under test with a high variety in sensor and application setups are investigated.

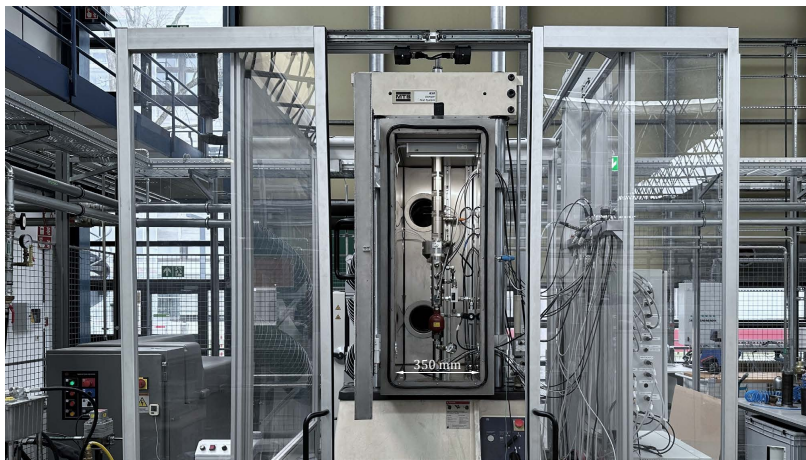


Figure 2: Uni-axial servo hydraulic test rig MTS 850 test damper at the chair of Fluid Systems at Technische Universität Darmstadt

50 Figure 2 shows the test rig providing dynamic testing in a temperature controlled environment in
51 a range of $T = -40\text{ }^{\circ}\text{C} \dots 100\text{ }^{\circ}\text{C}$. Dynamic forces of $F = \pm 50\text{ kN}$ at a cylinder stroke of up to
52 $\Delta z = 300\text{ mm}$ are possible [2]. This allows for a large number of static and dynamic experiments
53 to be performed. For example materials are tested for their strength, while dynamic transfer
54 functions of springs or dampers may also be determined. The measurement hardware used is
55 a dSPACE MicroLabBox with 32 analog in- and 16 outputs and all common digital interfaces.

1. IRI: <https://w3id.org/fst/resource/018beaa3-8fe6-7ab5-83f7-81468a8a8784>

56 The box is able to simulate in real time and is therefore suited to apply Hardware-in-the-Loop
 57 investigations [3]. All this illustrates that very heterogeneous test setups with a large number of
 58 different sensors can be examined on this modular test rig.

59 Figure 3 shows two very different test objects as examples. Both are chassis components for
 60 passenger cars with different complexity. The steel spring (left) is characterized simply with a
 61 deflection and a force sensor. The active air spring (right) [3], [4], [5], [6], on the other hand, has
 62 more degrees of freedom. The pressure and temperature in the spring must also be known, and
 63 additional displacement sensors are required to control the active system. This experiment also
 64 requires additional components, such as an external energy supply. In addition, the properties of
 65 the active air spring depend on the gas used, in this case dry air.

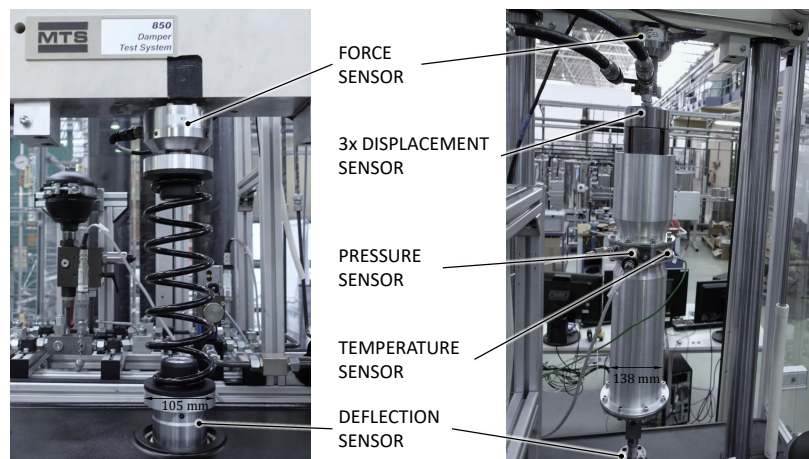


Figure 3: Two examples of different test objects whose dynamic properties are investigated. The left coil spring is measured with a deflection sensor and a force sensor to determine the transfer function. The air spring on the right is in addition also equipped with temperature, pressure and other displacement sensors.

66 There is a wide variety of sensor and component suppliers and most of the investigated hardware
 67 are new developments. Therefore, there are not always data sheets, let alone data sheets in a
 68 standardized or machine actionable form, available. This shows the need of a universal, efficient
 69 and easy metadata handling for this test environment.

70 From this modular setup test environment, three types of objects in use can be identified, which
 71 are described by similar information. For each of which information models are developed:

- 72 M1 Components (in-house developed as well as purchased)
- 73 M2 Substances (e.g. dry air which is used in air springs)
- 74 M3 Sensors (varying suppliers)

75 2.2 Research and User Objectives

76 Following we give the main objective for our way towards FAIR measurement data for the
 77 specific modular test rig as well as general requirements. The requirements are to be established
 78 on the basis of the following three specific but also generalizing examples or tasks which are
 79 typical during the described experiments. Furthermore, the aforementioned information models

80 are to be considered in conjunction with the specified requirements. Additionally, there are
81 requirements pertaining to experimental and measurement data.

82 Although the goal is to align as closely as possible with the FAIR criteria, including the sub-
83 criteria mentioned in [1], the approach taken here is to derive the requirements from the presented
84 practical examples. While there is overlap between the identified requirements and the FAIR
85 criteria, this overlap is not addressed further.

86 **1. Using basic sensor information during data acquisition** A typical task known by nearly
87 every experimenter is to add the sensor characteristics to the data acquisition environment. Each
88 sensor has an individual characteristic. In our case, these are exclusively linear characteristics,
89 which does not necessarily apply to all sensors. These characteristics can change over time,
90 which is why the sensors should be calibrated regularly. The sensor characteristics information
91 must be clearly assigned to the input channels of the data acquisition.

92 As already mentioned, there are plenty of sensor manufacturers all of whom provide sensor
93 information for their sensors, but there is no standard on how to provide this information.
94 Therefore, a universal sensor information model should meet the following requirements R_i .

95 R1 Information must be associated with a unique ID. (M1, M2, M3)

96 R2 The ID must be persistent. (M1, M2, M3)

97 R3 The information must be retrievable via ID (either known source or via protocol). (M1,
98 M2, M3)

99 R4 The information must be machine actionable². (M1, M2, M3)

100 R5 The sensor information model must include sensor characteristics (e.g. sensitivity and
101 bias). (M3)

102 R6 The information models should allow for changing information (and or redundant but
103 non-conflicting information). (M1, M2, M3)

104 **2. Tracing of data results back to component and substance information** Experimental data
105 is always subject to further processing and analysis, which can lead to new questions. It is
106 important to note that the experimenter and the scientist who conducts further analysis may not
107 be the same person. It is also crucial to identify the components and their properties used in the
108 experiment as their properties are dependent on the environmental conditions, particularly when
109 fluids are involved. Therefore, the ambient conditions should be recorded in the experiment.

110 The following requirements for the different information models to be developed are derived
111 from this problem.

112 R7 The component information model must allow representation of relevant quantities. (M1)

113 R8 Measurement results must be linked to measured data as well as component information,
114 which is used for model parameterization or as input in some other computation.

115 R9 Relevant physical properties should be able to depend on other variables. (M1, M2, M3)

2. Machine actionable means that the data can be processed automatically without human interaction.[7]

116 R10 Measurement results must specify which information to use when redundant data is
117 provided (e.g., multiple measurement ranges of a sensor).

118 **3. Using sensor information for uncertainty quantification** When evaluating experimental
119 data, it is essential to determine both systematic and stochastic uncertainties. This process
120 involves interpreting the uncertainty information provided by sensor manufacturers in data sheets
121 or calibration certificates and assigning it to the corresponding measured variables. However,
122 since sensor manufacturers do not adhere to standardized formats for reporting uncertainty
123 —such as those outlined by the Joint Committee for Guides in Metrology [8]— this interpretation
124 is laborious and time-consuming. Once completed, the extracted uncertainty information should
125 be available in the sensor information model to ensure accessibility and consistency.

126 R11 The sensor information model should include and differentiate typical uncertainty infor-
127 mation. (M3)

128 R12 The sensor information model should also specify the uncertainty information and the
129 source of them. (M3)

130 **In general** Hardware, substances, and sensors can be used at various test benches with different
131 data recording environments. This affects the reusability of the information and also consecutive
132 the choice of frameworks used to model the information. The usage at different test environments
133 also suggests that other aspects of a sensor or component may be significant, necessitating a
134 flexible information model.

135 Also this flexible standard information model must be flexible enough so that it can be reused
136 and adapted to describe different objects, where reoccurring descriptions for similar objects (e.g.
137 sensors) could then be made into an information model again. This ensures a recursive and
138 modular workflow. Also the whole process should be easy to use to get the users up and running
139 fast.

140 R13 The information models must be compatible to various measurement setups. (M1, M2,
141 M3)

142 R14 The information models must be hardware independent and, therefore, be deployable in
143 various experimental setups. (M1, M2, M3)

144 R15 The information models must be programming language independent. (M1, M2, M3)

145 R16 The information model[s] must be flexible and easily expandable.(M1, M2, M3)

146 R17 Version control of the information models is needed. (M1, M2, M3)

147 R18 Access control to the information models must be provided. (M1, M2, M3)

148 Experience has shown that test bench modifications require simple processes for connecting and
149 adding information. The more manual effort is required, the greater the likelihood of neglect or
150 bypassing the process by the experimenter. This can call the reliability of measurement metadata
151 into question and may even require repeating measurements.

152 R19 All information that is already available digitally should be collected automatically.

153 3 State of the Art and Relevant Standards

154 Numerous works and projects have focused on making research data FAIR, as seen in [9], [10],
155 [11].

156 The FAIR principles serve as guidelines. However, the specifics of how to achieve FAIR data are
157 still developing. Although future technologies may enhance these processes [12], current efforts
158 involve technologies and ideas from the semantic web, linked data and knowledge management
159 [13], [14], [15]. Since the early days of the Internet, technologies have been developed to
160 facilitate the interoperable availability of knowledge. Best practices and guidelines are provided
161 in resources like the FAIR CookBook [16].

162 **Persistent Identifiers (PID)** A crucial element towards achieving FAIR data is the use of unique
163 identifiers for specific objects [12]. A well-known example is the International Standard Book
164 Number (ISBN), which identifies books. However, it is not a web link and thus information
165 about the entity cannot be directly retrieved automatically. Consequently, the Digital Object
166 Identifier (DOI) has become established for identifying books and other digital objects, such as
167 published software code. It is important to note that the same object, such as a book, can have
168 multiple identifiers, e.g., an ISBN and a DOI.

169 **Semantic Web and Linked Data** The web contains an overwhelming amount of data, prepared
170 for human consumption, not standardized for machines. Humans can derive information from
171 context, a capability machines currently lack without assistance. The Semantic Web aims to
172 provide this assistance by making information available in a format that also machines can
173 process [17]. Promoted by the World Wide Web Consortium (W3C) [18], this initiative offers a
174 range of technologies and standards to facilitate this provision.

175 A core concept is the semantic presentation of data, contextualizing it through directed graphs
176 where objects (nodes) relate via directed edges. The standard framework for this is the Resource
177 Description Framework (RDF), also developed by W3C. For RDF-described information to
178 be interoperable, standardized vocabularies for nodes and edges are necessary, facilitated by
179 ontologies that store terminological knowledge.

180 The ultimate vision is a vast graph connecting knowledge across disciplines via standardized and
181 formalized terms, forming Web 3.0 [19]. Hitzler et al. [17] provide a comprehensive overview
182 of the Semantic Web. Relevant aspects for FAIR experimental data are summarized below.

183 **Resource Description Framework (RDF)** RDF represents relationships as subject-predicate-
184 object triples, a standard developed since the 1990s and continually refined [20], [21], [22].
185 Multiple triples build a bigger directed graph. Various serializations of the graphs exist, such as
186 Turtle or JSON-LD.

187 In RDF, Internationalised Resource Identifiers (IRIs) [23] are used.³ These unique IRIs denote
188 nodes and edges, serving as unique identifiers of information.[17]

3. Another option is the Uniform Resource Identifier (URI) [24], from which the IRI originated. The IRI expands the permissible character set. IRIs are used in this paper.

189 Objects can be connected or assigned properties, and data values in RDF are represented as
 190 literals, which are character strings that can also have assigned data types. However, objects can
 191 also be labeled as instances of a class.

192 **Ontologies** As ontologies may not be familiar to every scientist, especially in engineering
 193 disciplines where experiments are common, four basic questions are answered below.

194 *What is an ontology?*

195 The term "ontology" originates from philosophy and was characterized by Aristotle [25]. Since
 196 the late 20th century, the term has been adopted in computer science, where it refers to "a formal,
 197 explicit specification of a shared conceptualization" [26].

198 Staab et al. [25] provide a detailed overview of what exactly is meant by this term. Noy et al. [27]
 199 offer a more practical definition: "An ontology defines a common vocabulary for researchers
 200 who need to share information in a domain. It includes machine-interpretable definitions of basic
 201 concepts in the domain and relations among them" [27].

202 One of the well-known ontologies is the Friend of a Friend (FOAF) ontology⁴, which was
 203 developed to describe relationships between people, i.e., social networks.

204 The main components of an ontology are:

- 205 • Classes and subclasses, which describe concepts via common properties. These are
 206 analogous to classes in object-oriented programming to implement the concept of general-
 207 ization in contrast to individualization [28]. An example class that describes documents is
 208 [foaf:Document](#).
- 209 • Attributes or properties that can be assigned to a class and, in turn, point to another class,
 210 e.g., [foaf:maker](#), or contain a value, e.g., [foaf:name](#).

211 This small example is shown as a graph in the following Figure 4. The relationship (predicate)
 between a document (subject) and a person (object) is created via the object property maker.

PREFIX foaf: <<http://xmlns.com/foaf/0.1/>>

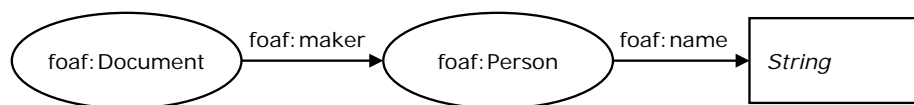


Figure 4: Simple example from the FOAF ontology, which represents the relationship between a document and a person with a name. Classes are oval and start with a capital letter by convention [17]. Properties that contain literal data are square.

212

213 Additional concepts such as owl:restriction allow the modeling of more complex concepts. The
 214 Web Ontology Language (OWL) [29], developed by the W3C, is often used to formulate complex
 215 ontologies.

216 *Why develop and use ontologies?*

4. <http://xmlns.com/foaf/0.1/>

217 For scientists in fields where ontologies are not the norm, the effort involved in creating and
218 using ontologies might seem substantial with little perceived benefit for the individual researcher.
219 However, data and information in the disciplines are often generated at significant expense in
220 terms of time and money. They should therefore also be prepared in such a way that they can
221 be reused. This is particularly evident in major initiatives for the FAIRification of data [30].
222 Considering the continuously increasing data-supported research, it is worthwhile to explicitly
223 create knowledge and prepare it in a way that it can be used by others (programs). Ontologies
224 offer this possibility and are already being successfully used in areas that rely on the research of
225 others, e.g., in life sciences [31].

226 *How to develop an ontology?*

227 Ontology engineering is a science in its own right. There are several methods for developing
228 ontologies, yet no single method has been established as the standard that must be strictly
229 followed. Femi Aminu et al. [32] provide an overview of ontology development methods and
230 categorize their advantages and disadvantages. Allemang and Hendler [28] provide practical
231 guidelines for developing ontologies and also give an overview of existing ones.

232 A common learning from all methods is: If possible, use existing, well-maintained vocabularies
233 [27], [33]. OBO Semantic Engineering Training also provides a guide on when not to develop
234 an ontology [34].

235 A second lesson is that development should be focused on a defined domain or application [27].
236 In the first draft, it is acceptable not to cover every possible application. Any given information
237 is better than none.

238 Thus, we develop information models for our application and utilize existing ontologies for this
239 purpose.

240 *What is the difference between an ontology and an information model?*

241 The distinction between an information model and an ontology is not completely straightforward.
242 In general, every ontology is an information model, but not every information model must be an
243 ontology [35]. An information model therefore does not have to contain all the information that
244 an ontology does. It is an application of the concept of an ontology to a specific problem.

245 Schulz et al. [36] provide an overview of the characteristics of both, shown in Table 1. However,
246 the authors themselves state that in reality, there is no sharp distinction in the use of the two
247 terms.

248 In our application, three information models are developed in RDF, which are based on existing
249 RDF vocabularies and ontologies.

250 Since only existing RDF vocabularies and ontologies are used, the extended modeling capabilities
251 of RDFS, or even more so that of OWL are not required and therefore RDFS or OWL are not
252 used.

Ontologies	Information Models
Contain classes that have really existing domain entities (particulars) as members	Classes have information entities as members
Represent real-world particulars in terms of their inherent properties	Represent artifacts that are built to collect or annotate information
Can exist independently of information models as long as only the existence of particular things is recorded	Are required to record beliefs or states of knowledge about real things or types of things (as represented by ontologies)
Context-independent	Context-dependent

Table 1: Comparison of ontologies and information models from [36]

253 4 Modeling Approach and Implementation

254 Three information models for 1. sensors, 2. components, and 3. substances were developed
 255 from the requirements of the work on the test bench and the known methods of knowledge
 256 representation. These information models were instantiated for the objects, used on the test
 257 environment presented, and made available online for use.

258 4.1 Information Model

259 In the following sections, we present the three developed information models, **M1**, **M2**, **M3**.
 260 While these models share fundamental properties, such as a common method for assigning IRIs
 261 and the use of the same ontologies, they differ in the information they contain and their structure.

262 **IRIs** Each object is assigned a unique identifier, for which we use a Universal Unique Identifier
 263 (UUID), version 7 [37]. This is a 128-bit code that can be generated automatically using a Python
 264 library⁵.

265 As persistent identifiers for the instances of our information model we use the *w3id.org* redirect
 266 service in combination with GitLab Webpages and GitLab repositories for each.

267 **Used Vocabulary** Various classes and properties are then linked to this object in a RDF graph.
 268 When linking, only known semantic vocabulary from established ontologies are used. The
 269 most important ontologies are summarised in the following table with their reference and their
 270 application domain.

271 **M1 Components** The model of a component is the most basic model and consists of three
 272 levels of information, metadata, further documentation and physical properties. Figure 5 presents
 273 a simplified version of the model, with nodes printed in bold, and edge descriptions in thin font.
 274 Any parent nodes are outlined with a box, and descriptive properties are arranged in tabular form
 275 below.

276 The metadata for the component is linked at the top level, defining it as a physical object with a
 277 name, serial number, and manufacturer, etc.. The UUID serves as the primary ID, but other IDs

5. <https://github.com/oittaa/uuid6-python>

abbreviation	URI prefix	application
rdf	http://www.w3.org/2000/01/rdf-schema#	RDF schema
dcTerms	http://purl.org/dc/terms/	general metadata terms
dcType	http://purl.org/dc/dcmitype/	general types
schema	http://schema.org/	general vocabulary
foaf	http://xmlns.com/foaf/0.1/	social networks
qudt	http://qudt.org/schema/qudt/	quantities and units
ssn	http://www.w3.org/ns/ssn/	sensor networks
ssn-system	http://www.w3.org/ns/ssn/systems/	systems for measurements
sosa	http://www.w3.org/ns/sosa/	sensors and actuators (based on ssn)

Table 2: Ontologies used with the abbreviation in the first column, the full link in the second column and the application area in the last column

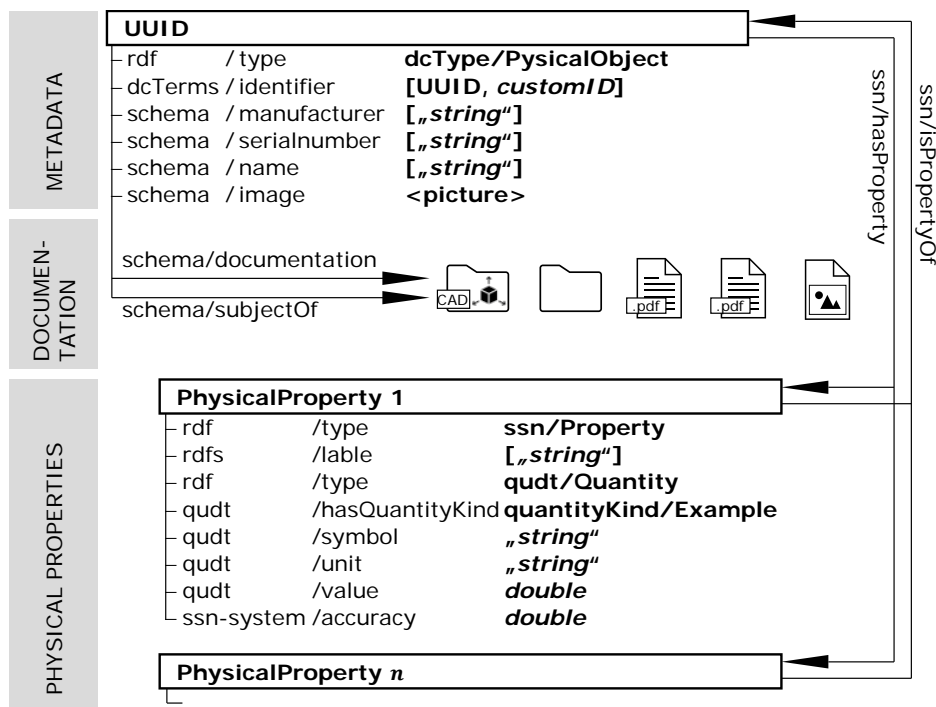


Figure 5: Simplified structure of the RDF graph of an information model of a component consisting of metadata, additional documentation and physical properties.

278 can also be assigned individually.

279 The second level provides additional information that cannot yet be processed by machines, such

280 as images, CAD data, reports, and data sheets.

281 The third level contains relevant physical properties. It is important to note that not all properties

282 of a component are specified. The user can specify the properties that are important for further

283 processing during or after an experiment. The RDF graph is expandable, allowing for additional

284 properties to be added at a later date if they are relevant for further investigations. The properties

285 displayed here consist of a fixed value, such as the volume of an air spring, cf. Section 2.1.

286 However, it is also possible to add characteristic fields, as shown in the next section on substances.

287 One example component developed at the chair of fluid systems is a gas spring which is exper-
288 imentally investigated in the test rig presented in section 2.1. The information model of this
289 component containing all its metadata and properties can be found at [https://w3id.org/fst](https://w3id.org/fst/290/resource/018bb4b1-db48-73b8-9d82-8a8ffb6ee225.ttl)
290 [/resource/018bb4b1-db48-73b8-9d82-8a8ffb6ee225.ttl](https://w3id.org/fst/resource/018bb4b1-db48-73b8-9d82-8a8ffb6ee225.ttl).

291 **M2 Substances** *Substances* in this context refers to <https://schema.org/ChemicalSubstance>,
292 namely "a portion of matter of constant composition, composed of molecular entities of
293 the same type or of different types". Only fluids, such as nitrogen or hydraulic oil, have been
294 described in the context of the particular test environment.

295 The information model of a substance, as shown in Figure 6, is based on that of the component.
296 The origin node **UUID** is described by metadata. Further documentation, such as safety data
297 sheets, can also be referenced, or physical properties analogous to those of the component can
298 be attached. However, these are not displayed in Figure 6 to provide a clearer overview.

299 However, the fluids used in the experiments whose information is shown here have physical
300 properties that depend on the ambient conditions. This is particularly evident in the properties
301 of gases, such as the density of nitrogen. The density ρ depends on the temperature T and the
302 pressure p . At pressures $p < 10$ bar the state can be described analytically with sufficient accuracy
303 using the ideal gas law $p = \rho RT$ and the specific gas constant of nitrogen $R_{N_2} = 297$ J/kgK [38].
304 At higher pressures the behaviour deviates from that of an ideal gas. Therefore, in standard
305 works such as the CRC Handbook of Chemistry and Physics [39], the VDI Wärmearbeitsatlas [40]
306 or the NIST Chemistry WebBook [41] density is given as a lookup table. The goal now is to
307 adequately represent this property in the information model.

308 A very direct approach would be to include all values or combinations of values in the graph.
309 However, this would lead to the graph becoming very large and confusing, and the actual coherent
310 information of the table is lost. It is much easier to store the values in a suitable data format and
311 refer to them in the information model, as well as describing the information stored in the file.
312 This means that the values can also be quickly read into a suitable data processing system and
313 used as a lookup table.

314 This is achieved in the model by using the open Hierarchical Data Format *.hdf5* [42]. The file
315 contains the lookup tables for the thermophysical property as well as the column and row vectors
316 that describe the table. The file is also described in the RDF graph, see Figure 6, where the
317 source of the data is given. The thermophysical property of the substance is linked to the dataset.

318 The complete information model of nitrogen as an example can be found at [https://w3id](https://w3id.org/fst/resource/018dba9b-f067-7d3e-8a4d-d60cebd70a8a.ttl)
319 [.org/fst/resource/018dba9b-f067-7d3e-8a4d-d60cebd70a8a.ttl](https://w3id.org/fst/resource/018dba9b-f067-7d3e-8a4d-d60cebd70a8a.ttl). The stored data⁶
320 originate from the NIST Chemistry WebBook [41].

321 **M3 Sensors** The last but not least information model represents sensors. Their Properties are
322 basically the same as those of the **Component** and **Substance**, which are directly linked to the

6. The data is stored in the file *nitrogen.h5* at <https://w3id.org/fst/resource/018dba9b-f067-7d3e-8a4d-d60cebd70a8a>

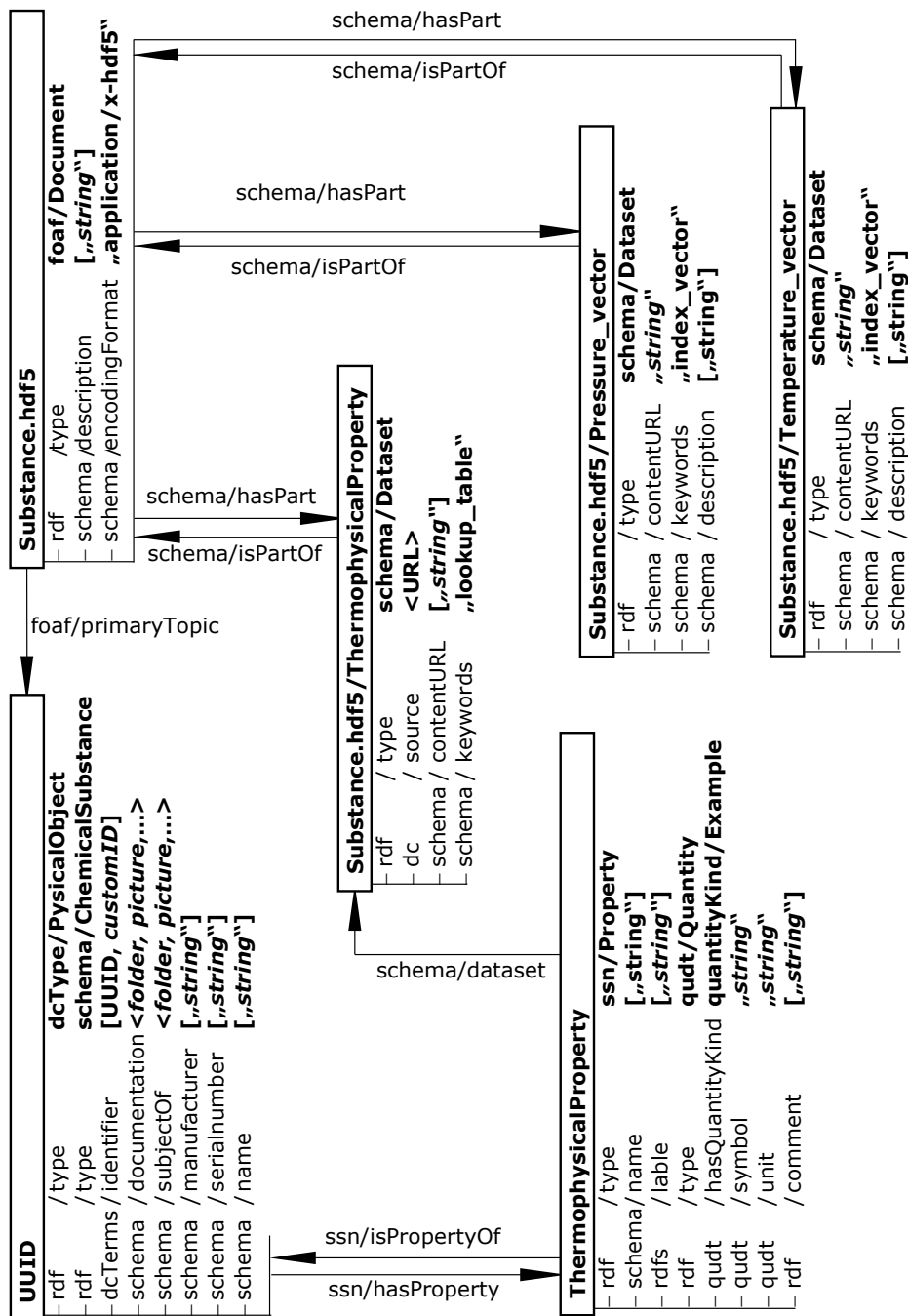


Figure 6: Simplified structure of the RDF graph of an information model of a substance. In addition to metadata of the substance, dependent thermophysical properties are also represented, the data of which is available as a lookup table.

323 origin node **UUID**. Figure 7 summarizes them in the **Properties** category. Sensors can also
 324 process signals, and these capabilities are grouped together under the **SensorCapability** node in
 325 Figure 7.

326 This capability is further specified in the second level. The test environment only uses sensors
 327 with a linear characteristic, so the characteristic properties of the characteristic are described

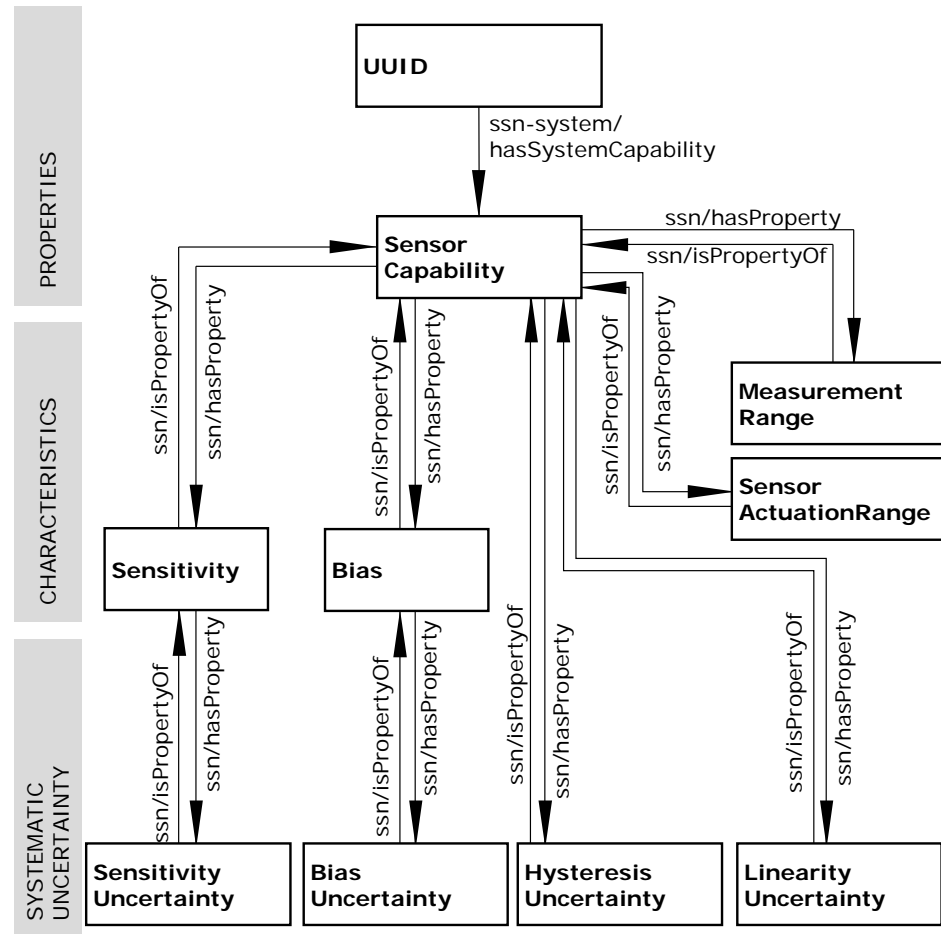


Figure 7: Overview of the main classes of the sensor information model.

328 by **Sensitivity** and **Bias**. In addition, the measuring range and the analogue output signal in the
 329 modelled case are specified under **SensorActuationRange**.

330 Finally, the uncertainty properties are described by four classes **SensitivityUncertainty**, **Bia-**
 331 **sUncertainty**, **LinearityUncertainty**, and **HysteresisUncertainty**. This distinction complies
 332 to the publications [43], [44] and originates from the book by Tränkler [45]. The first two
 333 uncertainty classes directly refer to the uncertainty of the sensitivity and bias parameters and are
 334 therefore linked to them. The last two uncertainties describe the entire characterisation and are
 335 therefore related to the sensor capability.

336 The complete model can be found in Figure 12 appendix 7. An example sensor can also be found
 337 under the following link <https://w3id.org/fst/resource/064f05d1-5d2d-7a6f-8000-a3da10f5a1a3>.
 338

339 4.2 Implementation and Usage

340 In the following, we will briefly show how the specific models are generated, stored and made
 341 available for further use.

342 **Instantiating the Models** The Python RDFlib package [46] is utilised to generate the information
343 models of specific entities, which can be serialised in various formats, including Turtle and
344 JSON-LD.

345 For unique components, the model can be created manually, while for a larger number of objects,
346 such as sensors, with the same description, they can be generated automatically from a table or a
347 database. An example code is available in the following repository [https://github.com/tes-](https://github.com/tes-t123-all/hydropulser-database-scripts)
348 [t123-all/hydropulser-database-scripts](https://github.com/tes-t123-all/hydropulser-database-scripts).

349 **Storing Information** Once the information is generated it has to be stored. As an online
350 repository service we use GitLab⁷ [47]. The generated information models and other descriptive
351 files are stored there in repositories. This has the following advantages:

- 352 i. It allows the upload of files, folders and subfolders of any format.
- 353 ii. It has an integrated version control with git [48] (R17). This allows the user to continuously
354 expand the information models (R16). In addition, a reference to the corresponding version
355 (commit hash) can be used to reference and access a corresponding status.
- 356 iii. It offers integrated access control (R18). Repositories can be made public or private at
357 any time. This can also be changed at a later date. Therefore, also data that is not to be
358 publicly accessible can be uploaded and used.
- 359 iv. The web interface is able to render Markdown files making it possible to display human-
360 readable information in a well formatted and easily digestible way.

361 One disadvantage is that no persistent identifiers are created. The URLs with which a repository,
362 a folder or a file is called up depend on the paths within the repositories and their storage location.
363 Therefore, a redirect service described below is required.

364 **Providing Information - Redirect Service** The long-term accessibility of information on the
365 web depends on several critical factors. Pointers to resources must be unique per resource, even
366 for multiple similar resources, which can be achieved using a unique ID (R1). Unique IDs
367 must remain unchanged once established for a resource to ensure persistence (R2). The content
368 associated with these IDs should also be easily retrievable using established web standards (R3)
369 Choosing a URI namespace that incorporates one unique UUID per item, like “[https://w3id.org/f-](https://w3id.org/fst/resource/UUID)
370 [st/resource/UUID](https://w3id.org/fst/resource/UUID)” and using that string as ID should be sufficient to achieve uniqueness on the
371 world wide web for every item (R1, R2). The “<https://>” part also indicates that these persistent
372 IDs can function as a direct mechanism to retrieve the information. For information hosted on
373 platforms like GitLab, like in this example, where URLs may change over time, a persistent entry
374 point is required. This entry point must allow for a flexible redirection to the current location of
375 the information and must be modifiable as needed to ensure continuous accessibility.

376 The w3id.org web service, provided and maintained by the W3C Permanent Identifier Community
377 Group, offers an open and permanent URL redirection service [49]. This initiative is also backed
378 by organizations with the joint goals to keep the longevity of the domain and also the webserver
379 functioning [49]. The deployed web server relies on .htaccess files to define redirection rules [49],

7. The instance is provided by RWTH Aachen University: <https://git.rwth-aachen.de/>.

380 which are typical for the Apache HTTP Server open-source project [50]. The service is managed
381 through a GitHub repository that is linked on their website, with the earliest pull requests dating
382 back to mid 2013 [49]. This indicates that the service should have been active for over a decade
383 and also continues to show significant activity.

384 Therefore the persistence of w3id.org URLs can be assumed, making the w3id.org web service a
385 reliable choice as the entry point for persistent IDs. Since both the web server and the entire
386 w3id.org repository are open source, the entire ecosystem could be reconstructed by a third party
387 if necessary.

388 The following sections provide a more detailed description of the complete redirection service.

389 The information is stored in a directory in a GitLab repository and is therefore accessible online.
390 The directory name is the UUID. By providing the information via an online platform, it is
391 independent of the specific test environment (R14) and can be used on multiple test benches
392 (R13).

393 The directory contains three representations of the information model RDF graph: a turtle-file (ttl),
394 a JSON-LD-file, and an XML-file. This redundancy allows users to choose the representation
395 that suits them best without the need for conversion. Additionally, a markdown-file is used to
396 store a human-readable representation of the information, which GitLab is able to render in
397 the browser. Any further documentation, such as data sheets or images, are stored in simple
398 directories, as previously described in the information models.

399 An example directory structure is shown on the right-hand side in Figure 8. The figure also
400 demonstrates how the information can be retrieved from any requesting program.

401 A http(s) GET request is used to access information. It consists of a prefix symbolized with
402 a square □, the UUID #, and the desired file extension, .* . The extension specifies which
403 representation of the information is required. If no type is specified, the request is forwarded to
404 the repository.

405 First the request is sent to the w3id.org web server defined by the prefix □ that functions as a
406 redirect service. There the URL is automatically reassembled using rules stored in an .htaccess-
407 file. For a given UUID # and extension .* the assembled URL redirects to an automatically
408 generated HTML-file stored in GitLab Pages, which in turn points and redirects to the requested
409 file in the repository through a html meta refresh tag.

410 This allows the file to be returned as a response, provided that a corresponding HTML-file exists
411 in GitLab Pages for the requested file in the repository.

412 This two stage redirect has three advantages:

- 413 i The W3ID service is maintained and hosted by a large community and is therefore likely
414 to be available for a very long time (persistence).
- 415 ii The first stage of the automated redirect at w3id does not need to be updated even if names
416 and paths in the GitLab repository change.
- 417 iii The redirect service at GitLab Pages can be created using an automatic program, therefore
418 maintaining the redirects requires very little effort overall (R19).

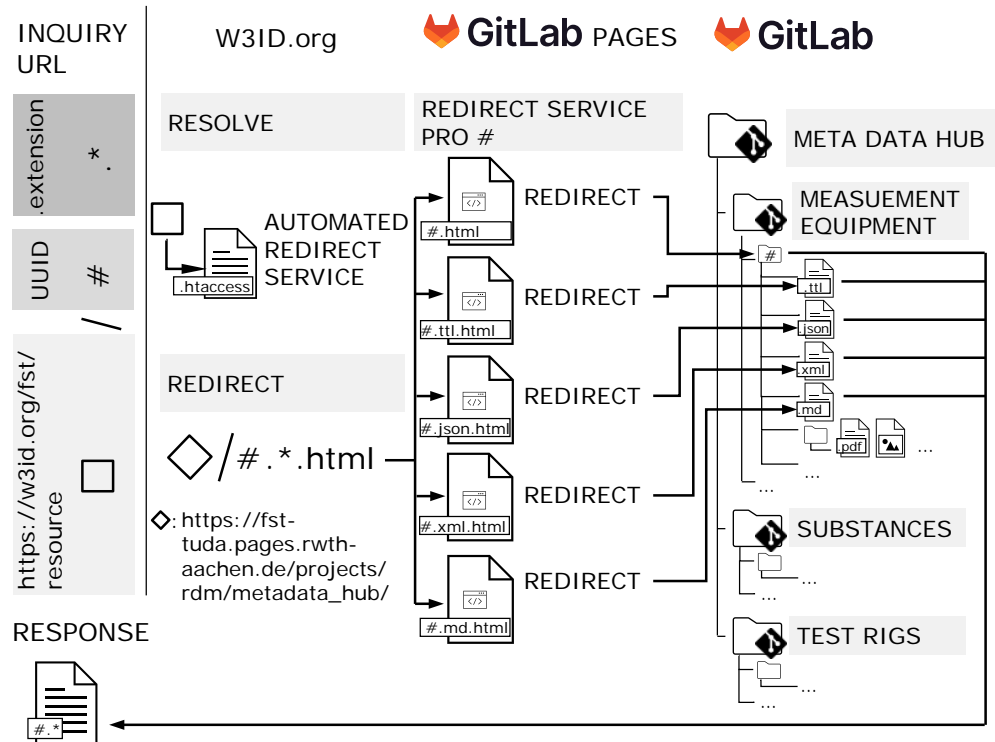


Figure 8: Two stage redirect service to get data from the META DATA HUB repository on GitLab using W3ID.org and GitLab pages.

419 **CI/CD Pipeline for Automated Update of the Second Redirect Stage** The functionality of the
 420 developed software⁸ that automatically generates the HTML-files is described in more detail
 421 in the following. Additionally, the software is embedded into a CI/CD Pipeline combined with
 422 GitLab Pages to further automate the process.

423 The primary objective of the second stage redirect is to establish a single, primary base URL
 424 that does not require frequent updates and resolves all UUIDs to their corresponding repository
 425 and directory. Both the repository and directory path may undergo changes. For instance, the
 426 repository location could shift within the GitLab instance, or the data set directory location could
 427 alter in relation to the repository. Both actions result in alterations to the URL, which would
 428 necessitate updates to the w3id service. Updating the URLs within the w3id.org service would be
 429 an unfeasible amount of work for the w3id service team. Therefore, it is necessary to implement
 430 a second stage in the redirect process, whereby the URLs are managed automatically by the user.

431 The CI/CD Pipeline of the META DATA HUB repository is depicted in Figure 9. Initially,
 432 the user updates or creates new data set directories within one of the sub-module repositories.
 433 Subsequently, the user must also update and commit the modified sub-modules within the META
 434 DATA HUB repository. Every commit uploaded to the main branch of the META DATA HUB
 435 repository initiates the CI/CD pipeline. The GitLab CI/CD pipeline⁹ configuration is stored as
 436 the `.gitlab-ci.yml`-file within the root directory of the META DATA HUB repository.

8. The software can be found at <https://github.com/test123-all/html-redirect-file-generator>

9. Further information on how to configure GitLab CI/CD pipelines can be found at <https://docs.gitlab.com/ci/pipelines/>.

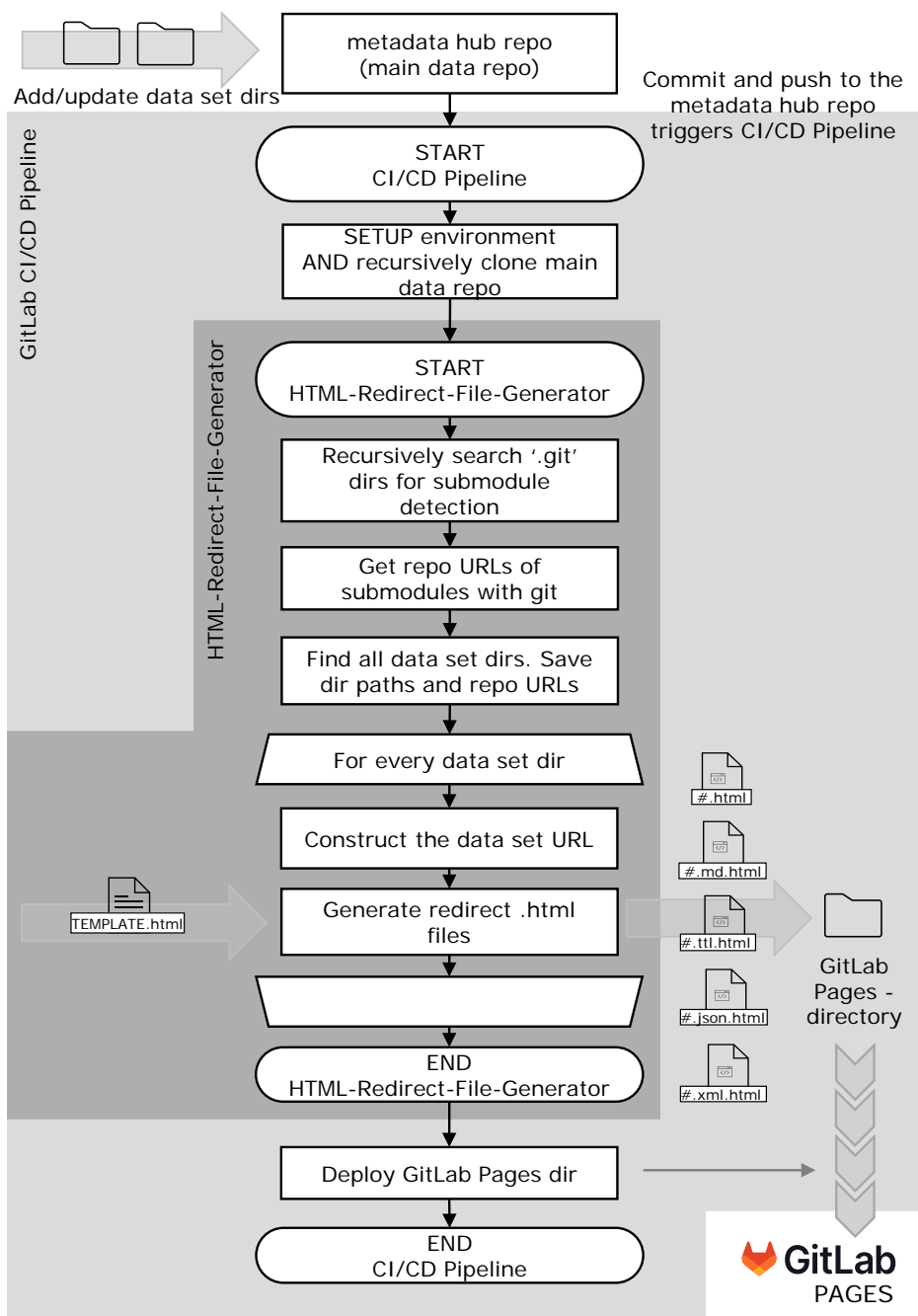


Figure 9: CI/CD Pipeline of the META DATA HUB repository on GitLab using the HTML-File-Redirect-Generator and GitLab Pages to generate and host the HTML-redirect-files of the second redirect stage.

- 437 The initial step undertaken by the pipeline is to establish the requisite environment. This involves
- 438 the download of requisite software and the recursive cloning of the META DATA HUB repository.
- 439 The recursive clone ensures the replication of the META DATA HUB repository and all its sub-
- 440 module repositories, which contain the data set directories.
- 441 The next stage is the initiation of the HTML-File-Redirect-Generator. The following input

442 arguments are required: the path of the cloned META DATA HUB repository and the path where
443 the HTML-redirect-files will be saved to. The HTML-redirect-files path is set to the GitLab
444 Pages directory. The GitLab Pages directory is a special directory path within the pipeline where
445 the HTML-files must be saved in order for them to be accessible and deployable by the GitLab
446 Pages web service.

447 The HTML File Redirect Generator employs a recursive search of the META DATA HUB data
448 directory and its subdirectories to identify directories with the extension ".git." Each cloned
449 repository, including its submodules, contains a ".git" directory at the root level, which enables
450 the distinction of these directories and the retrieval of their respective directory paths in relation
451 to the META DATA HUB data directory.

452 In the subsequent step, the URLs of the distinct repositories can be retrieved by executing a git
453 command at the location of the different root paths of the submodules. The URLs are also stored
454 for later retrieval. Subsequently, all data set directories for the distinct submodules are identified
455 by a location convention within the different submodule directories. The data set directory paths
456 are also saved to a list for later retrieval.

457 For each data set directory, a URL must be constructed that includes the repository URL of the
458 data set directory in question, as well as the directory path of the data set relative to the repository
459 directory. Subsequently, for each data set directory URL and a HTML redirect template, the
460 main redirect URL for the dataset and the different file type redirects (.ttl, .json, .xml, .md) are
461 constructed, parsed within the template and saved as their own file. The HTML files are saved
462 to the GitLab Pages directory. This results in five redirect files as shown in Figure 9 on the
463 right-hand side.

464 Once the HTML redirect files have been created, the HTML File Redirect Generator terminates
465 and the process is handed back to the CI/CD pipeline. In the subsequent step, the pipeline
466 deploys the GitLab Pages directory to the GitLab Pages web service, which is also shown on the
467 right-hand side of the Figure 9 at the bottom.

468 Subsequently, the pipeline reaches its final state and successfully terminates, ultimately providing
469 the automatically generated HTML files for the second stage of the redirect, as illustrated in
470 Figure 8.

471 5 Application

472 Now that the implementation is known and the data is accessible, the question remains as
473 to how the data can be integrated into the environment that is being used. A measurement
474 recording program was created using the MATLAB software [51] and the Simulink simulation
475 environment [52] due to proprietary restrictions of the test environment. In order to be able to
476 use the RDF data in MATLAB it is necessary to develop a MATLAB package that downloads
477 the RDF information, extracts it from a turtle file and converts it into a MATLAB *struct* data
478 structure.¹⁰ If the information is not publicly accessible, a personal access token is used to obtain
479 access authorisation. With the help of the IRI, the information can be used both for recording

10. The software is available at:<https://github.com/test123-all/fst-rdf-utilities>

480 and analysing measurement data. The information is therefore traceable to a single source,
 481 without the need to keep values inside different scripts updated. It is also possible to extend the
 482 recorded data afterwards through loaded information, that were not explicitly recorded during
 483 the experiment, to generate new knowledge or easily check for anomalies that were not obvious
 484 before. Ultimately this leads to processes and workflows that do not need the rerecording of
 485 time- and cost-ineffective measurements that do not provide much added value.

486 21 components, 6 substances and 162 sensors have already been created.¹¹ Experiments were
 487 conducted in the transfer project T12 of the CRC805 based on the created metadata. No reference
 488 can be made to publications that use this data as the experimental data is still being analysed.
 489 For validation purposes, the three example tasks and their workflow are presented below.

490 **1. Using basic sensor information during data acquisition** The sensor information is used in
 491 a Simulink model that specifies the measurement data acquisition on the software side using
 492 blocks provided by dSpace.

493 The IRIs are represented by a QR code to provide easy accessibility and are respectively perma-
 494 nently assigned to one unique entity of a physical sensor. This is combined with other human
 495 readable information on a label and attached to the sensor, cf. Figure 10. With the help of QR
 496 code readers, the IRI can easily be inserted anywhere in a computer program.

497 In Simulink, the IRI is used in a custom block to automatically retrieve curve information and
 498 metadata about the sensor from the repository. This is shown on the right hand side of Figure 10.

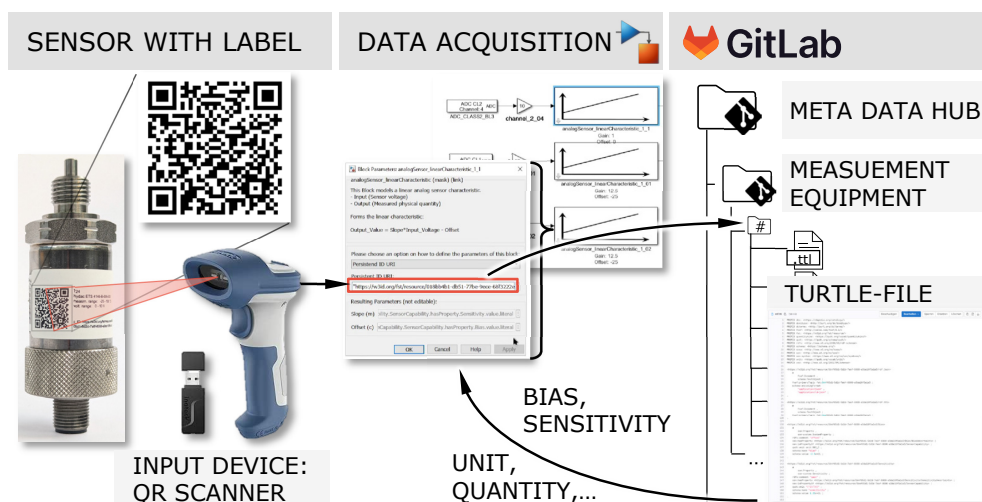


Figure 10: Interaction between IRI on the label of the sensor, the data acquisition software and the sensor information in the GitLab repository.

499 Overall, this approach meets all requirements R1-R6. Additionally, it offers the benefits of saving
 500 time when setting up new measurement environments and ensures that all relevant data is stored.

501 **2. Tracing provenance of results to component and substance information** To demonstrate
 502 how the information can be traced, let us examine the structure of a measurement in Figure 11.

11. Available at <https://git.rwth-aachen.de/fst-tuda/public/metadata>, although some of the data is not publicly accessible.

503 The data represents a measurement of hydraulic accumulators [53] and is stored as a MATLAB
 504 struct, which is a hierarchical data structure.

505 Each sensor provides a time series as measurement data. These series are highlighted in yellow.
 506 When examining the time series, it is important to note that in addition to the actual values, there is
 507 also metadata stored that pertains to the measurement itself, such as the unit of measurement and
 508 physical quantity. It is worth noting that additional information is not stored in each measurement
 509 file, but rather the link to the sensor is given under *sensor_data*. This allows for additional
 510 information to be obtained afterwards.

511 Two types of measurement metadata are also stored and highlighted in blue in Figure 11. Firstly,
 512 there are model parameters that can be set and read out, such as the excitation frequency. On the
 513 other hand, there is metadata which contains more general information, including details about
 514 the experimenter, software setup, and hardware setup.

515 The setup information is initially presented as a list to ensure all components used are recorded
 516 accurately. The information provided always includes the IRI to enable retrieval of all relevant
 517 information at any time. Objects can also be specified in a nested form. In this example, the
 518 accumulator is filled with nitrogen, as shown in Figure 11 at the bottom right. Additionally, a
 519 type is specified for all components, with the label **TestObject** used to identify the analysed
 520 component. It is important to note that the used label is not part of any standardized vocabulary.
 521 However, *sosa:FeatureOfInterest* could be a first suitable option.

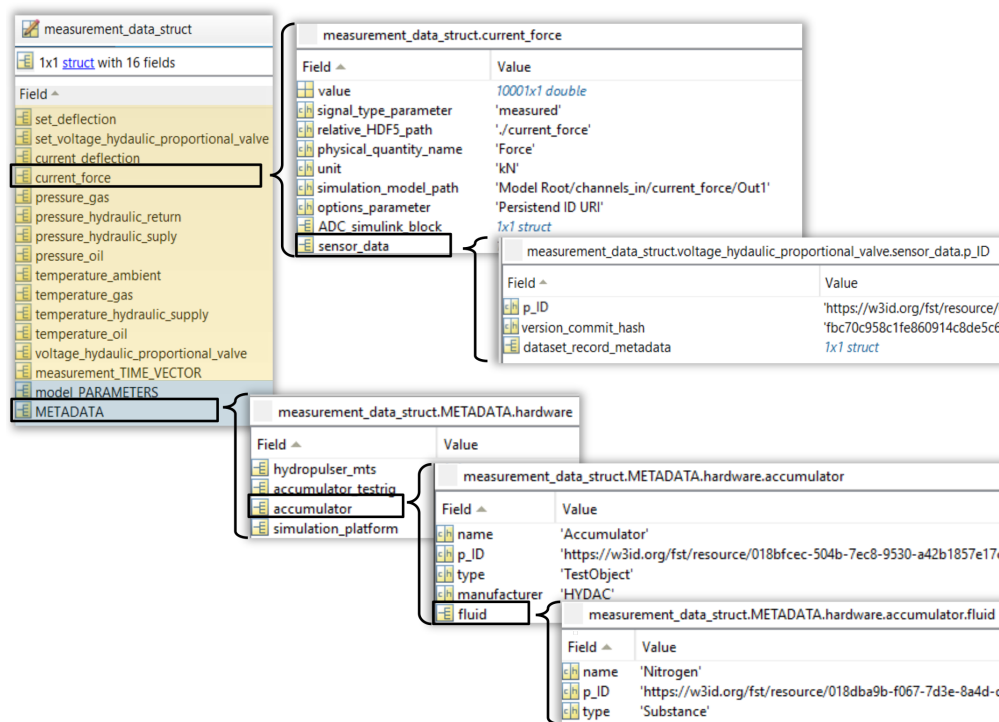


Figure 11: Excerpt from the data structure of a measurement in MATLAB. The time series are highlighted in yellow and metadata of the measurement are marked in blue.

522 As no data analysis has been published yet, it is not possible to demonstrate how the results can

523 be traced back to the components used. R8 and R10 are therefore only partly fulfilled. There are
524 also plans to automatically create a graph of the experiment itself to enable searches for specific
525 measurements.

526 **3. Using Sensor information for uncertainty quantification** Section 4.1 details how uncertainty
527 information is integrated into the sensor model. This model supports storing multiple types of
528 uncertainty data directly connected to the sensor characteristics as well as overall uncertainty
529 information provided by the sensor manufacturer. We implemented this approach for various
530 sensor types, including among others pressure, force, and displacement sensors, across different
531 manufacturers. The provided uncertainty data can be easily and directly applied for quantifying
532 measurement uncertainty.

533 Additionally, it serves as a foundation for advanced uncertainty propagation methods, such as
534 those implemented in a MATLAB framework [44], [54], enabling the transformation of sensor
535 systematic uncertainty from the time domain to the frequency domain [43].

536 The IRI provides access to both general R11 and specific R12 uncertainty information.

537 **In general** The description of the information using RDF graphs ensures that it is independent
538 of the programming language R15 and hardware R14 used. In order for the models to be used
539 on any test environment, it is only necessary to program the appropriate interfaces to retrieve the
540 information from the repositories and insert it in the measurement program (R13).

541 **6 Conclusion**

542 This research has highlighted the importance of FAIR principles in managing experimental data,
543 demonstrating significant improvements in the accessibility, interoperability, and reusability of
544 data through tailored information models and linked data technologies. By integrating persistent
545 identifiers and standardized vocabularies within a dynamic test environment, we have streamlined
546 data acquisition and analysis processes, enhancing both efficiency and reliability.

547 The application of these models within our test environment has not only reduced manual effort
548 but has also increased the adaptability and scalability of our data management systems. This
549 approach promises substantial benefits for future experimental research.

550 Moving forward, the focus will be on broadening the application of these models to include
551 a wider range of experimental setups and to improve the usability and efficiency of the tools
552 to build ontologies and information-models. These efforts will continually result in further
553 supporting of the scientific community in achieving more systematic and effective research data
554 management.

555 7 Appendix

556 **Support** If you are interested in using the proposed framework, please do not hesitate to contact
 557 the authors for further support under info@fst.tu-darmstadt.de. The required software code is
 558 already referenced in the text and summarised in the following table 3. The software code that is
 559 not explicitly mentioned in the text, but which is relevant for this paper, is summarised in table 4

Name	Description	URL
hydropulser-database-scripts	This repository contains the python scripts to create the RDF dataset files (.ttl, .json, .xml, .md) of the different information models. Some of the scripts have a template character, for example the one for the sensors, other ones are purely hard-coded. This software repository is mentioned in 4.2.	https://github.com/test123-all/hydropulser-database-scripts
HTML-Redirect-File-Generator	Software to generate the HTML-Redirect-Files for the second redirect stage of the persistent ID URI redirect service explained in more detail in 4.2.	https://github.com/test123-all/html-redirect-file-generator
FST RDF utilities	Software that is able to load graphs given by a main node into python dictionaries and matlab structs to be able to load and use a subset of RDF data more easily and efficiently without the need to break long established and intuitive data usage habits in Python and Matlab. The program needs to start the mapping of the graph into a hierarchical data structure at one main node and will traverse and load all sub nodes, their sub-nodes and so on, that are connected and directed away from them until there are no nodes left or got already used in the graph. This software gets mentioned in section 5.	https://github.com/test123-all/fst-rdf-utilities

Table 3: Software referenced in this paper

Name	Description	URL
FST Label Creator	Software (Python package without CLI or GUI yet) with which custom labels on PDF label sites (only DIN A4 for now) with chosen preconfigured label templates and digital spreadsheets (like from Microsoft Excel, Apache OpenOffice Calc or LibreOffice Calc) could be generated. The generated PDF label site[s] can be printed later on store-bought label sheets using a laser printer. The previously chosen preconfigured label template inside the software should match the label template of the label sheet. The software is in an initial and very raw state and therefore might be subject to significant changes in the future.	https://github.com/test123-all/fst-label-creator
NIST Scripts	The scripts used to download the data from the NIST Chemistry WebBook - Thermophysical Properties of Fluid Systems - website and to generate the HDF5 files from the downloaded data and the corresponding RDF files. The scripts have not been published to avoid potential conflicts with German or American law (since NIST is a U.S. government institute and agency). Additionally, NIST may have an interest in ensuring that we do not share scripts that download their data. For this reason, we have only used the scripts internally to improve our data workflow and provide feedback.	/ not available /

Table 4: Additional software used in this paper

560 **Information Model of a Sensor** The complete information model is shown in the following
 561 figure. As there are a relatively large number of nodes, they are grouped together. An RDF graph
 562 of an example sensor is available at [https://w3id.org/fst/resource/064f05d1-5d2](https://w3id.org/fst/resource/064f05d1-5d2d-7a6f-8000-a3da10f5a1a3.ttl)
 563 [d-7a6f-8000-a3da10f5a1a3.ttl](https://w3id.org/fst/resource/064f05d1-5d2d-7a6f-8000-a3da10f5a1a3.ttl). This can be displayed, for example, with an online RDF
 564 visualiser <https://issemantic.net/rdf-visualizer>.

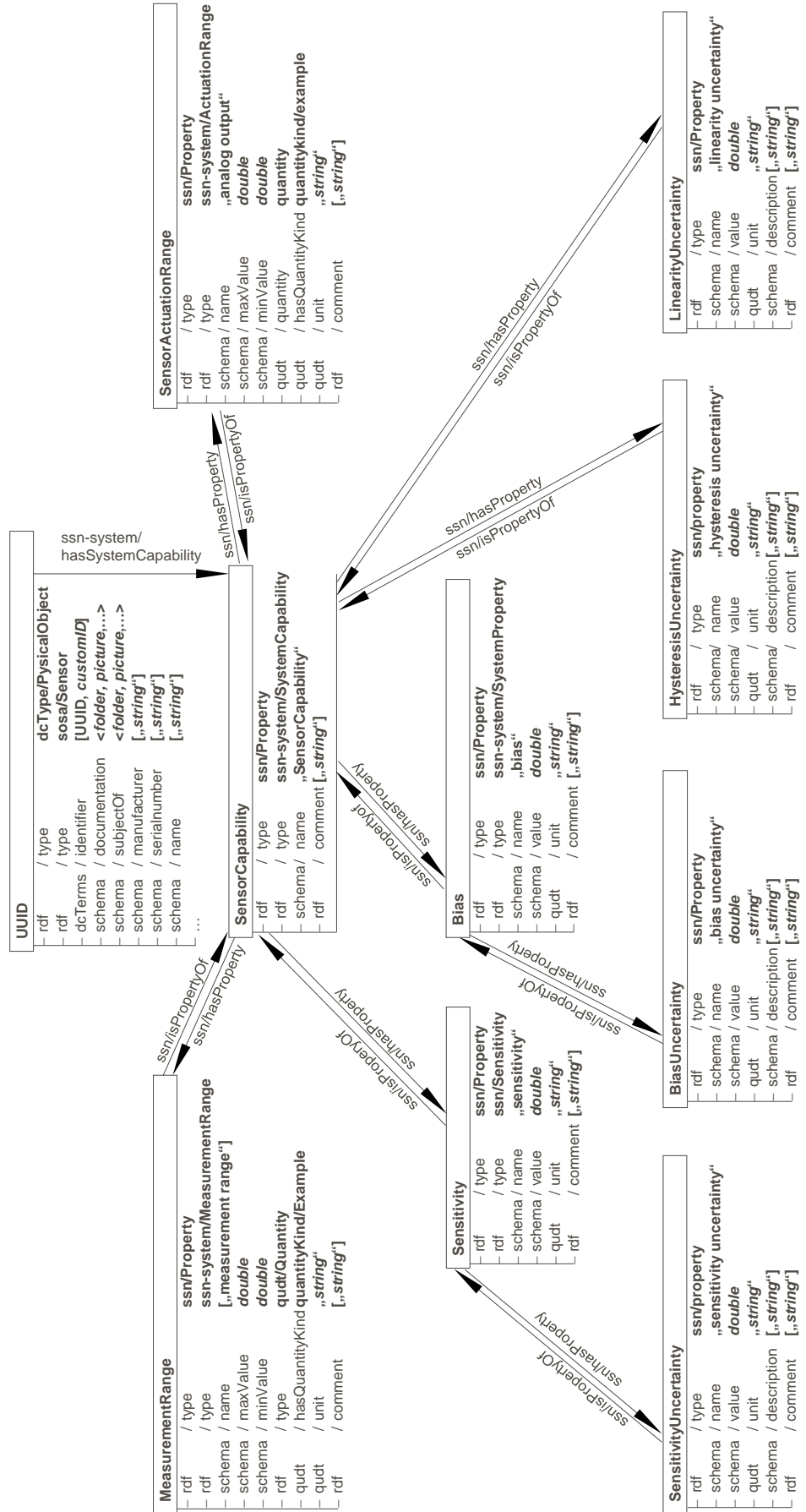


Figure 12: Complete sensor information model

565 8 Acknowledgements

566 Special thanks to the funding projects:

567 Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project number
568 57157498 – SFB805,

569 Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) –Project number
570 432233186 - AIMS,

571 Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) –Project number
572 442146713 - NFDI4Ing.

573

574 9 Roles and contributions

575 **Manuel Rexer:** Investigation, Conceptualization, Project administration, Visualization, Writing
576 – original draft, Writing – review & editing

577 **Nils Preuß:** Conceptualization, Software, Methodology, Writing – original draft

578 **Sebastian Neumeier:** Software, Data curation, Methodology, Visualization, Writing – original
579 draft, Writing – review & editing

580 **Peter F. Pelz:** Supervision, Funding acquisition

581 References

- 582 [1] M. D. Wilkinson et al., “The fair guiding principles for scientific data management and
583 stewardship,” *Scientific data*, vol. 3, p. 160 018, 2016. DOI: [10.1038/sdata.2016.18](https://doi.org/10.1038/sdata.2016.18).
- 584 [2] M. Puff and P. F. Pelz, *Entwicklung einer Prüfpezifikation zur Charakterisierung von*
585 *Luftfedern* (FAT-Schriftenreihe). Berlin: Verband der Automobilindustrie (VDA), 2009.
- 586 [3] E. Lenz, P. Hedrich, and P. F. Pelz, “Aktive luftfederung – modellierung, regelung und
587 hardware-in-the-loop-experimente,” *Forschung in Ingenieurwesen*, pp. 1–15, 2018, ISSN:
588 0015-7899. DOI: [10.1007/s10010-018-0272-2](https://doi.org/10.1007/s10010-018-0272-2).
- 589 [4] P. F. Pelz, P. Groche, M. E. Pfetsch, and M. Schaeffner, Eds., *Mastering Uncertainty in*
590 *Mechanical Engineering* (Springer Tracts in Mechanical Engineering), 1st ed. 2021. Cham:
591 Springer International Publishing and Imprint Springer, 2021, ISBN: 978-3-030-78353-2.
592 DOI: [10.1007/978-3-030-78354-9](https://doi.org/10.1007/978-3-030-78354-9).
- 593 [5] P. Hedrich, E. Lenz, and P. F. Pelz, “Minimizing of kinetosis during autonomous driving,”
594 *ATZ Woldwide*, vol. 120, no. 7-8, pp. 68–75, 2018.
- 595 [6] P. Hedrich, “Konzeptvalidierung einer aktiven luftfederung im kontext autonomer fahrzeuge,”
596 Dissertation, Technische Universität Darmstadt, Darmstadt, 2018.
- 597 [7] S. Islam, “Fair digital objects, persistent identifiers and machine actionability,” *FAIR*
598 *Connect*, vol. 1, pp. 29–34, 2023, 1, ISSN: 2949-799X. DOI: [10.3233/FC-230001](https://doi.org/10.3233/FC-230001).
599 [Online]. Available: <https://doi.org/10.3233/FC-230001>.

- 600 [8] Joint Committee for Guides in Metrology, *Evaluation of Measurement Data—Guide to*
601 *the Expression of Uncertainty in Measurement*. 2008. Accessed: Oct. 15, 2021. [Online].
602 Available: <https://www.bipm.org/en/publications/guides>.
- 603 [9] European Commission, Directorate-General for Research, and Innovation, *Turning FAIR*
604 *into reality – Final report and action plan from the European Commission expert group*
605 *on FAIR data // Turning FAIR into reality : final report and action plan from the European*
606 *Commission expert group on FAIR data*. Luxemburg and Hannover: Publications Office,
607 Amt für Veröffentlichungen, and Technische Informationsbibliothek, 2018, ISBN: 978-
608 92-79-96546-3. DOI: [10.2777/1524](https://doi.org/10.2777/1524).
- 609 [10] D. Hornung, F. Spreckelsen, and T. Weiß, “Agile research data management with open
610 source: Linkahead: Ing.grid volume 1 issue 1 2023,” 2024. DOI: [10.48694/INGGRID.3](https://doi.org/10.48694/INGGRID.3866)
611 [866](https://doi.org/10.48694/INGGRID.3866).
- 612 [11] S. Ferenz and A. Nieße, “Towards improved findability of energy research software
613 by introducing a metadata-based registry: Ing.grid volume 1 issue 2 2023,” 2023. DOI:
614 [10.48694/INGGRID.3837](https://doi.org/10.48694/INGGRID.3837).
- 615 [12] B. Mons, C. Neylon, J. Velterop, M. Dumontier, L. O. B. Da Silva Santos, and M. D.
616 Wilkinson, “Cloudy, increasingly fair; revisiting the fair data guiding principles for the
617 european open science cloud,” *Information Services & Use*, vol. 37, no. 1, pp. 49–56,
618 2017, ISSN: 01675265. DOI: [10.3233/ISU-170824](https://doi.org/10.3233/ISU-170824).
- 619 [13] M. D. Wilkinson et al., “Interoperability and fairness through a novel combination of web
620 technologies,” *PeerJ Computer Science*, vol. 3, e110, 2017. DOI: [10.7717/peerj-cs.1](https://doi.org/10.7717/peerj-cs.110)
621 [10](https://doi.org/10.7717/peerj-cs.110).
- 622 [14] A. Mazimwe, I. Hammouda, and A. Gidudu, “Implementation of fair principles for ontolo-
623 gies in the disaster domain: A systematic literature review,” *ISPRS International Journal*
624 *of Geo-Information*, vol. 10, no. 5, p. 324, 2021. DOI: [10.3390/ijgi10050324](https://doi.org/10.3390/ijgi10050324).
- 625 [15] N. Jeliaskova et al., “Towards fair nanosafety data,” *Nature nanotechnology*, vol. 16, no. 6,
626 pp. 644–654, 2021. DOI: [10.1038/s41565-021-00911-6](https://doi.org/10.1038/s41565-021-00911-6).
- 627 [16] P. Rocca-Serra et al., “The fair cookbook - the essential resource for and by fair doers,”
628 *Scientific data*, vol. 10, no. 1, p. 292, 2023. DOI: [10.1038/s41597-023-02166-3](https://doi.org/10.1038/s41597-023-02166-3).
- 629 [17] P. Hitzler, M. Krötzsch, S. Rudolph, and Y. Sure, *Semantic Web: Grundlagen* (eXa-
630 men.press). Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, ISBN: 9783540339946.
631 [Online]. Available: [http://nbn-resolving.org/urn:nbn:de:bsz:31-epflicht-](http://nbn-resolving.org/urn:nbn:de:bsz:31-epflicht-1582600)
632 [1582600](http://nbn-resolving.org/urn:nbn:de:bsz:31-epflicht-1582600).
- 633 [18] W3C, *About us*, 12.04.2024. Accessed: Apr. 12, 2024. [Online]. Available: [https://www](https://www.w3.org/about/)
634 [.w3.org/about/](https://www.w3.org/about/).
- 635 [19] T. Berners-Lee, *Giant global graph*, 2007. Accessed: Apr. 12, 2024. [Online]. Available:
636 [https://web.archive.org/web/20160713021037/http://dig.csail.mit.edu](https://web.archive.org/web/20160713021037/http://dig.csail.mit.edu/breadcrumbs/node/215)
637 [/breadcrumbs/node/215](https://web.archive.org/web/20160713021037/http://dig.csail.mit.edu/breadcrumbs/node/215).
- 638 [20] D. Brickley and R. V. Guha, *Resource description framework (rdf) schema specification*,
639 W3C Recommendation, Ed., 1999. Accessed: Apr. 8, 2024. [Online]. Available: [https:](https://www.w3.org/TR/PR-rdf-schema/)
640 [/www.w3.org/TR/PR-rdf-schema/](https://www.w3.org/TR/PR-rdf-schema/).

- 641 [21] D. Brickley and R. V. Guha, *Rdf schema 1.1*, W3C Recommendation, Ed., 2014. [Online].
642 Available: <https://www.w3.org/TR/rdf-schema/>.
- 643 [22] RDF Working Group, Ed., *Rdf 1.1 concepts and abstract syntax*, 2014. [Online]. Available:
644 <https://www.w3.org/TR/rdf11-concepts/>.
- 645 [23] M. J. Dürst and M. Suignard, *Internationalized resource identifiers (iris)*, 2005. DOI:
646 [10.17487/RFC3987](https://doi.org/10.17487/RFC3987). [Online]. Available: [https://www.rfc-editor.org/info/rfc](https://www.rfc-editor.org/info/rfc3987)
647 [3987](https://www.rfc-editor.org/info/rfc3987).
- 648 [24] T. Berners-Lee, R. T. Fielding, and L. M. Masinter, *Uniform resource identifier (uri):*
649 *Generic syntax*, 2005. DOI: [10.17487/RFC3986](https://doi.org/10.17487/RFC3986). [Online]. Available: [https://www.r](https://www.rfc-editor.org/info/rfc3986)
650 [fc-editor.org/info/rfc3986](https://www.rfc-editor.org/info/rfc3986).
- 651 [25] S. Staab and R. Studer, Eds., *Handbook on ontologies* (International handbooks on
652 information systems), Second Edition. Berlin and Heidelberg: Springer, 2009, ISBN:
653 9783662499955. [Online]. Available: [http://www.loc.gov/catdir/enhancements](http://www.loc.gov/catdir/enhancements/fy1312/2008943971-d.html)
654 [/fy1312/2008943971-d.html](http://www.loc.gov/catdir/enhancements/fy1312/2008943971-d.html).
- 655 [26] R. Studer, V. Benjamins, and D. Fensel, “Knowledge engineering: Principles and methods,”
656 *Data & Knowledge Engineering*, vol. 25, no. 1-2, pp. 161–197, 1998, ISSN: 0169023X.
657 DOI: [10.1016/S0169-023X\(97\)00056-6](https://doi.org/10.1016/S0169-023X(97)00056-6).
- 658 [27] N. F. Noy and D. I. McGuinness, *Ontology development 101: A guide to creating your*
659 *first ontology*, 2001.
- 660 [28] D. Allemang and J. A. Hendler, *Semantic Web for the working ontologist: Effective model-*
661 *ing in RDFS and OWL*, 2nd ed. Waltham, MA: Morgan Kaufmann Publishers/Elsevier,
662 2011, ISBN: 9780123859655. DOI: [10.1016/C2010-0-68657-3](https://doi.org/10.1016/C2010-0-68657-3).
- 663 [29] W3C, Ed., *Owl 2 web ontology language document overview (second edition)*, 2017.
664 Accessed: Apr. 14, 2024. [Online]. Available: [https://www.w3.org/TR/owl2-overvi](https://www.w3.org/TR/owl2-overview/)
665 [ew/](https://www.w3.org/TR/owl2-overview/).
- 666 [30] S. Stall et al., “Make scientific data fair,” *Nature*, vol. 570, no. 7759, pp. 27–29, 2019.
667 DOI: [10.1038/d41586-019-01720-7](https://doi.org/10.1038/d41586-019-01720-7).
- 668 [31] OBO Foundry, Ed., *Obo foundry*, 2024. Accessed: Apr. 14, 2024. [Online]. Available:
669 <https://obofoundry.org/>.
- 670 [32] E. Femi Aminu, I. O. Oyefolahan, M. Bashir Abdullahi, and M. T. Salaudeen, “A re-
671 view on ontology development methodologies for developing ontological knowledge
672 representation systems for various domains,” *International Journal of Information Engi-*
673 *neering and Electronic Business*, vol. 12, no. 2, pp. 28–39, 2020, ISSN: 20749023. DOI:
674 [10.5815/ijieeb.2020.02.05](https://doi.org/10.5815/ijieeb.2020.02.05).
- 675 [33] Z. Aloulou, K. Belhajjame, D. Grigori, and R. Acker, “A domain-independent ontology for
676 capturing scientific experiments,” in *Information Search, Integration, and Personalization*,
677 ser. Communications in Computer and Information Science, D. Kotzinos, D. Laurent,
678 N. Spyrtos, Y. Tanaka, and R.-i. Taniguchi, Eds., vol. 1040, Cham: Springer International
679 Publishing, 2019, pp. 53–68, ISBN: 978-3-030-30283-2. DOI: [10.1007/978-3-030-30](https://doi.org/10.1007/978-3-030-30284-9_4)
680 [284-9{\textunderscore}4](https://doi.org/10.1007/978-3-030-30284-9_4).

- 681 [34] N. Matentzoglou and S. Toro, *Creating an ontology from scratch - obo semantic engineering*
682 *training*, 2024. Accessed: Apr. 12, 2024. [Online]. Available: <https://oboacademy.github.io/obook/howto/create-ontology-from-scratch/>.
683
- 684 [35] PedroD, *What is the difference between an information model and an ontology?* 2015.
685 Accessed: Apr. 12, 2024. [Online]. Available: <https://stackoverflow.com/questions/30562062/what-is-the-difference-between-an-information-model-and-an-ontology>.
686
687
- 688 [36] S. Schulz, D. Schober, C. Daniel, and M.-C. Jaulent, "Bridging the semantics gap between
689 terminologies, ontologies, and information models," in *Proceedings of the 13th World*
690 *Congress on Medical Informatics*, ser. Studies in health technology and informatics, C.
691 Safran, Ed., Amsterdam: IOS Press, 2010, pp. 1000–1004, ISBN: 9781607505877. DOI:
692 [10.3233/978-1-60750-588-4-1000](https://doi.org/10.3233/978-1-60750-588-4-1000).
- 693 [37] K. R. Davis, B. Peabody, and P. Leach, *Universally unique identifiers (uuid): Internet-*
694 *draft*, 2023. [Online]. Available: <https://datatracker.ietf.org/doc/draft-ietf-uuidrev-rfc4122bis/14/>.
695
- 696 [38] P. Stephan, K. Schaber, K. Stephan, and F. Mayinger, *Thermodynamik: Grundlagen und*
697 *technische Anwendungen Band 1: Einstoffsysteme* (Springer-Lehrbuch), 19., ergänzte
698 Aufl. 2013. Berlin, Heidelberg and s.l.: Springer Berlin Heidelberg, 2013, ISBN: 978-3-
699 642-30098-1. DOI: [10.1007/978-3-642-30098-1](https://doi.org/10.1007/978-3-642-30098-1).
- 700 [39] W. M. Haynes and D. R. Lide, Eds., *CRC handbook of chemistry and physics: A ready-*
701 *reference book of chemical and physical data*, 91. ed., 2010 - 2011. Boca Raton, Fla.:
702 CRC Press, 2010, ISBN: 9781439820773.
- 703 [40] VDI-Gesellschaft Verfahrenstechnik und Chemieingenieurwesen, Ed., *VDI-Wärmeatlas:*
704 *Mit 320 Tabellen* (VDI-Buch), 11., bearb. und erw. Aufl. Berlin and Heidelberg: Springer
705 Vieweg, 2013, ISBN: 978-3-642-19982-0.
- 706 [41] P. Linstrom, *Nist chemistry webbook, nist standard reference database 69*, 1997. DOI:
707 [10.18434/T4D303](https://doi.org/10.18434/T4D303).
- 708 [42] The HDF Group, *Hierarchical data format, version 5*, 2023. [Online]. Available: <https://github.com/HDFGroup/hdf5>.
709
- 710 [43] M. Rexer, P. F. Pelz, and M. M. G. Kuhr, "Propagation of systematic sensor uncertainty
711 into the frequency domain," *ASCE-ASME Journal of Risk and Uncertainty in Engineering*
712 *Systems, Part B: Mechanical Engineering*, pp. 1–41, 2025, ISSN: 2332-9017. DOI: [10.1115/1.4067828](https://doi.org/10.1115/1.4067828).
713
- 714 [44] M. Rexer, P. F. Pelz, and M. M. G. Kuhr, "Propagation of Systematic Sensor Errors
715 into the Frequency Domain: A MATLAB Software Framework," in *Model Validation*
716 *and Uncertainty Quantification, Vol. 3*, ser. Conference Proceedings of the Society for
717 Experimental Mechanics Series, R. Platz, G. Flynn, K. Neal, and S. Ouellette, Eds.,
718 Cham: Springer Nature Switzerland, 2025, pp. 15–21, ISBN: 978-3-031-68892-8. DOI:
719 [10.1007/978-3-031-68893-5_3](https://doi.org/10.1007/978-3-031-68893-5_3).
- 720 [45] H.-R. Tränkler and G. Fischerauer, *Das Ingenieurwissen: Messtechnik*. Berlin, Heidelberg:
721 Springer Berlin Heidelberg, 2014, ISBN: 978-3-662-44029-2. DOI: [10.1007/978-3-662-44030-8](https://doi.org/10.1007/978-3-662-44030-8).
722

- 723 [46] D. Krech et al., *RdfLib*, 2023. DOI: [10.5281/zenodo.6845245](https://doi.org/10.5281/zenodo.6845245). [Online]. Available:
724 <https://github.com/RDFLib/rdfLib>.
- 725 [47] GitLab Inc., *Gitlab*, 2024. [Online]. Available: <https://gitlab.com>.
- 726 [48] J. Hamano, S. Pearce, and L. Torvalds, *Git*, 2024. [Online]. Available: <https://git-scm.com/>.
- 728 [49] W3C Permanent Identifier Community Group, *W3id.org - permanent identifiers for the*
729 *web*, 12.02.2024. [Online]. Available: <https://w3id.org/>.
- 730 [50] Apache HTTP Server Project, *Apache http server "httpd"*, 2024. [Online]. Available:
731 <https://httpd.apache.org/>.
- 732 [51] The MathWorks Inc., *Matlab*, Natick, Massachusetts, United States, 2019. [Online].
733 Available: <https://www.mathworks.com>.
- 734 [52] The MathWorks Inc., *Simulink*, Natick, Massachusetts, United States, 2019. [Online].
735 Available: <https://www.mathworks.com>.
- 736 [53] M. Rexer, P. Kloft, F. Bauer, J. Hartig, and P. F. Pelz, "Foam accumulators: Packaging and
737 weight reduction for mobile applications," in *12th International Fluid Power Conference*
738 *(12. IFK)*, Dresden: Technische Universität Dresden, 2020, pp. 181–188. DOI: [10.25368](https://doi.org/10.25368/2020.26)
739 [/2020.26](https://doi.org/10.25368/2020.26).
- 740 [54] M. Rexer, S. Neumeier, and M. M. G. Kuhr, *Propagation of systematic sensor errors into*
741 *the frequency domain - a matlab software framework*, 2024. DOI: [10.5281/ZENODO.10](https://doi.org/10.5281/ZENODO.10532137)
742 [532137](https://doi.org/10.5281/ZENODO.10532137).