ANNEX 81

EBC
Energy in Building and Communities Programme

International Energy Agency

Survey of metadata schemas for data-driven smart buildings

Energy in Buildings and Communities Technology Collaboration Programme

May 2022

ANNEX 81

EBC
Energy in Building and Communities Programme

International Energy Agency

Survey of metadata schemas for data-driven smart buildings

Energy in Buildings and Communities Technology Collaboration Programme

May 2022

Authors

Gabe Fierro, Colorado School of Mines, Colorado, USA (gtfierro@mines.edu)
Pieter Pauwels, Eindhoven University of Technology, Netherlands (p.pauwels@tue.nl)

Contributing Authors

Aslak Johansen, University of Southern Denmark, Denmark (asjo@mmmi.sdu.dk)
Tianzhen Hong, Berkeley Lab - LBNL, California, USA (thong@lbl.gov)
Arne Hansen, Buildings Evolved, Australia (arne@buildingsevolved.com)
Dimitrios Rovas, University College London, UK (d.rovas@ucl.ac.uk)
Stephen White, CSIRO, Australia (stephen.d.white@csiro.au)
Chun Ping Gao, Building and Construction Authority, Singapore (gao_chun_ping@bca.gov.sg)
TK Wang, VBIS, Australia (tkwang@vbis.com.au)
Maggie Sullivan, Switch Automation, Colorado, USA (esullivan@switchautomation.com)

Published by CSIRO, PO Box 330, Newcastle, NSW 2300, Australia

ISBN 978-1-4863-1896-4

Additional copies of this report may be obtained from: EBC Executive Committee Support Services Unit (ESSU), C/o AECOM Ltd, The Colmore Building, Colmore Circus Queensway, Birmingham B4 6AT, United Kingdom
www.iea-ebc.org
essu@iea-ebc.org

Preface

The International Energy Agency

The International Energy Agency (IEA) was established in 1974 within the framework of the Organisation for Economic Co-operation and Development (OECD) to implement an international energy programme. A basic aim of the IEA is to foster international co-operation among the 30 IEA participating countries and to increase energy security through energy research, development and demonstration in the fields of technologies for energy efficiency and renewable energy sources.

The IEA Energy in Buildings and Communities Programme

The IEA co-ordinates international energy research and development (R&D) activities through a comprehensive portfolio of Technology Collaboration Programmes (TCPs). The mission of the IEA Energy in Buildings and Communities (IEA EBC) TCP is to support the acceleration of the transformation of the built environment towards more energy efficient and sustainable buildings and communities, by the development and dissemination of knowledge, technologies and processes and other solutions through international collaborative research and open innovation. (Until 2013, the IEA EBC Programme was known as the IEA Energy Conservation in Buildings and Community Systems Programme, ECBCS.)
The high priority research themes in the EBC Strategic Plan 2019-2024 are based on research drivers, national programmes within the EBC participating countries, the Future Buildings Forum (FBF) Think Tank Workshop held in Singapore in October 2017 and a Strategy Planning Workshop held at the EBC Executive Committee Meeting in November 2017. The research themes represent a collective input of the Executive Committee members and Operating Agents to exploit technological and other opportunities to save energy in the buildings sector, and to remove technical obstacles to market penetration of new energy technologies, systems and processes. Future EBC collaborative research and innovation work should have its focus on these themes.
At the Strategy Planning Workshop in 2017, some 40 research themes were developed. From those 40 themes, 10 themes of special high priority have been extracted, taking into consideration a score that was given to each theme at the workshop. The 10 high priority themes can be separated in two types namely 'Objectives' and 'Means'. These two groups are distinguished for a better understanding of the different themes.

Objectives - The strategic objectives of the EBC TCP are as follows:

- reinforcing the technical and economic basis for refurbishment of existing buildings, including financing, engagement of stakeholders and promotion of co-benefits;
- improvement of planning, construction and management processes to reduce the performance gap between design stage assessments and real-world operation;
- the creation of 'low tech', robust and affordable technologies;
- the further development of energy efficient cooling in hot and humid, or dry climates, avoiding mechanical cooling if possible;
- the creation of holistic solution sets for district level systems taking into account energy grids, overall performance, business models, engagement of stakeholders, and transport energy system implications.

Means - The strategic objectives of the EBC TCP will be achieved by the means listed below:

- the creation of tools for supporting design and construction through to operations and maintenance, including building energy standards and life cycle analysis (LCA);
- benefitting from 'living labs' to provide experience of and overcome barriers to adoption of energy efficiency measures;
- improving smart control of building services technical installations, including occupant and operator interfaces;
- addressing data issues in buildings, including non-intrusive and secure data collection;
- the development of building information modelling (BIM) as a game changer, from design and construction through to operations and maintenance.

The themes in both groups can be the subject for new Annexes, but what distinguishes them is that the 'objectives' themes are final goals or solutions (or part of) for an energy efficient built environment, while the 'means' themes are instruments or enablers to reach such a goal. These themes are explained in more detail in the EBC Strategic Plan 2019-2024.

The Executive Committee

Overall control of the IEA EBC Programme is maintained by an Executive Committee, which not only monitors existing projects, but also identifies new strategic areas in which collaborative efforts may be beneficial. As the Programme is based on a contract with the IEA, the projects are legally established as Annexes to the IEA EBC Implementing Agreement. At the present time, the following projects have been initiated by the IEA EBC Executive Committee, with completed projects identified by (*) and joint projects with the IEA Solar Heating and Cooling Technology Collaboration Programme by (☼):

Annex 1: Load Energy Determination of Buildings (*)
Annex 2: Ekistics and Advanced Community Energy Systems (*)
Annex 3: Energy Conservation in Residential Buildings (*)
Annex 4: Glasgow Commercial Building Monitoring (*)
Annex 5: Air Infiltration and Ventilation Centre
Annex 6: Energy Systems and Design of Communities (*)
Annex 7: Local Government Energy Planning (*)
Annex 8: Inhabitants Behaviour with Regard to Ventilation (*)
Annex 9: Minimum Ventilation Rates (*)
Annex 10: Building HVAC System Simulation (*)
Annex 11: Energy Auditing (*)
Annex 12: Windows and Fenestration (*)
Annex 13: Energy Management in Hospitals (*)
Annex 14: Condensation and Energy (*)
Annex 15: Energy Efficiency in Schools (*)
Annex 16: BEMS 1- User Interfaces and System Integration (*)
Annex 17: BEMS 2- Evaluation and Emulation Techniques (*)
Annex 18: Demand Controlled Ventilation Systems (*)
Annex 19: Low Slope Roof Systems (*)
Annex 20: Air Flow Patterns within Buildings (*)
Annex 21: Thermal Modelling (*)
Annex 22: Energy Efficient Communities (*)
Annex 23: Multi Zone Air Flow Modelling (COMIS) (*)
Annex 24: Heat, Air and Moisture Transfer in Envelopes (*)
Annex 25: Real time HVAC Simulation (*)
Annex 26: Energy Efficient Ventilation of Large Enclosures (*)
Annex 27: Evaluation and Demonstration of Domestic Ventilation Systems (*)
Annex 28: Low Energy Cooling Systems (*)
Annex 29: ☼ Daylight in Buildings (*)
Annex 30: Bringing Simulation to Application (*)
Annex 31: Energy-Related Environmental Impact of Buildings (*)
Annex 32: Integral Building Envelope Performance Assessment (*)
Annex 33: Advanced Local Energy Planning (*)
Annex 34: Computer-Aided Evaluation of HVAC System Performance (*)
Annex 35: Design of Energy Efficient Hybrid Ventilation (HYBVENT) (*)
Annex 36: Retrofitting of Educational Buildings (*)
Annex 37: Low Exergy Systems for Heating and Cooling of Buildings (LowEx) (*)
Annex 38: ☼ Solar Sustainable Housing (*)
Annex 39: High Performance Insulation Systems (*)
Annex 40: Building Commissioning to Improve Energy Performance (*)
Annex 41: Whole Building Heat, Air and Moisture Response (MOIST-ENG) (*)
Annex 42: The Simulation of Building-Integrated Fuel Cell and Other Cogeneration Systems (FC+COGEN-SIM) (*)
Annex 43: ☼ Testing and Validation of Building Energy Simulation Tools (*)
Annex 44: Integrating Environmentally Responsive Elements in Buildings (*)
Annex 45: Energy Efficient Electric Lighting for Buildings (*)
Annex 46: Holistic Assessment Tool-kit on Energy Efficient Retrofit Measures for Government Buildings (EnERGo) (*)
Annex 47: Cost-Effective Commissioning for Existing and Low Energy Buildings (*)
Annex 48: Heat Pumping and Reversible Air Conditioning (*)
Annex 49: Low Exergy Systems for High Performance Buildings and Communities (*)
Annex 50: Prefabricated Systems for Low Energy Renovation of Residential Buildings (*)
Annex 51: Energy Efficient Communities (*)
Annex 52: ☼ Towards Net Zero Energy Solar Buildings (*)
Annex 53: Total Energy Use in Buildings: Analysis and Evaluation Methods (*)
Annex 54: Integration of Micro-Generation and Related Energy Technologies in Buildings (*)
Annex 55: Reliability of Energy Efficient Building Retrofitting - Probability Assessment of Performance and Cost (RAP-RETRO) (*)
Annex 56: Cost Effective Energy and CO2 Emissions Optimization in Building Renovation (*)
Annex 57: Evaluation of Embodied Energy and CO2 Equivalent Emissions for Building Construction (*)
Annex 58: Reliable Building Energy Performance Characterisation Based on Full Scale Dynamic Measurements (*)
Annex 59: High Temperature Cooling and Low Temperature Heating in Buildings (*)

Annex 60: New Generation Computational Tools for Building and Community Energy Systems (*)
Annex 61: Business and Technical Concepts for Deep Energy Retrofit of Public Buildings (*)
Annex 62: Ventilative Cooling (*)
Annex 63: Implementation of Energy Strategies in Communities (*)
Annex 64: LowEx Communities - Optimised Performance of Energy Supply Systems with Exergy Principles (*)
Annex 65: Long-Term Performance of Super-Insulating Materials in Building Components and Systems (*)
Annex 66: Definition and Simulation of Occupant Behavior in Buildings (*)
Annex 67: Energy Flexible Buildings (*)
Annex 68: Indoor Air Quality Design and Control in Low Energy Residential Buildings (*)
Annex 69: Strategy and Practice of Adaptive Thermal Comfort in Low Energy Buildings
Annex 70: Energy Epidemiology: Analysis of Real Building Energy Use at Scale
Annex 71: Building Energy Performance Assessment Based on In-situ Measurements
Annex 72: Assessing Life Cycle Related Environmental Impacts Caused by Buildings
Annex 73: Towards Net Zero Energy Resilient Public Communities
Annex 74: Competition and Living Lab Platform
Annex 75: Cost-effective Building Renovation at District Level Combining Energy Efficiency and Renewables
Annex 76: ☼ Deep Renovation of Historic Buildings Towards Lowest Possible Energy Demand and CO_2 Emissions
Annex 77: ☼ Integrated Solutions for Daylight and Electric Lighting
Annex 78: Supplementing Ventilation with Gas-phase Air Cleaning, Implementation and Energy Implications
Annex 79: Occupant-Centric Building Design and Operation
Annex 80: Resilient Cooling
Annex 81: Data-Driven Smart Buildings
Annex 82: Energy Flexible Buildings Towards Resilient Low Carbon Energy Systems
Annex 83: Positive Energy Districts
Annex 84: Demand Management of Buildings in Thermal Networks
Annex 85: Indirect Evaporative Cooling
Annex 86: Energy Efficient Indoor Air Quality Management in Residential Buildings

Working Group - Energy Efficiency in Educational Buildings (*)
Working Group - Indicators of Energy Efficiency in Cold Climate Buildings (*)
Working Group - Annex 36 Extension: The Energy Concept Adviser (*)
Working Group - HVAC Energy Calculation Methodologies for Non-residential Buildings (*)
Working Group - Cities and Communities
Working Group - Building Energy Codes

Summary

The IEA Annex 81[1] on Data-Driven Smart Buildings "*imagines a future world empowered by access to discoverable, reliable, ubiquitous real-time data from buildings, such that digital solutions can rapidly scale and where energy efficiency knowledge can be widely encapsulated and disseminated within highly accessible software 'Applications'.*"

Metadata schemas are a key tool for achieving this vision. A metadata schema provides a labeling/cataloging structure that represents the associations between data being collected and the objects that the data is sourced from and/or is in some way related to. In this way, a metadata schema gives an agreed standard approach for attaching context to data. It supports implementation of the FAIR (Findable, Accessible, Interoperable, Reusable) data principles.

Activity A3 in IEA Annex 81 'Data-Driven Smart Buildings' aims to provide the knowledge, standards, protocols, and procedures for low-cost, high-quality data capture, sharing, and utilisation in buildings, particularly focusing on 'Data Information Management'.

In this work, we present a survey of available metadata schemas for data-driven buildings. The focus is on metadata schemas for the operations and management stage of the building lifecycle. It excludes schemas that are used solely or predominantly for the design and construction phase of the building.

Buildings produce endless streams of sensor, meter, actuator, and other cyber-physical data. The task is to easily capture, organise and use this data for energy efficiency and other smart building applications. Ideally, the installation, configuration and use of software applications (that use available data) can be automated, to avoid the need for costly expert assistance.

Unfortunately, a business-as-usual approach treats each building as a stand-alone entity, replete with bespoke engineering solutions and a mix of standardised and bespoke labelling practices, to describe objects and states within the building. Consequently, application programming interfaces interacting with this data must be manually implemented with bespoke code, because the data naming and organization differs from building to building, and even from building system to building system.

This approach is impractical and creates significant inefficiencies that hinder the adoption of digital technologies in buildings, and impedes scaling up of energy productivity solutions. Annotating this data, so that it can be re-used as effectively and meaningfully as possible, regardless of the building typology, location or fabric, is a task best achieved by a common metadata schema (essentially a standard) that can be applied across the built environment. Metadata seeks to create effective abstractions of the inherent complexities in each building that allow the creation of site-agnostic (i.e. *portable*) applications. This will further the proliferation of value-adding applications and services.

Metadata for smart buildings serves two primary audiences: (1) those in charge of operational management at a building level, who need to find and access data for reporting and contractor management and (2) software application developers creating innovative streamlined software solutions at a portfolio or sector-wide level.

This survey is targeted at engineers and technical managers amongst these audiences, with the aim of helping decision makers to select suitable metadata schema(s) for their needs.

[1] https://annex81.iea-ebc.org/

While the survey is inherently highly technical in nature, it hopes to provide insight and clarity with respect to the overall structure, benefits, trade-offs, and arguments behind each of the major metadata schemas available for data-driven smart buildings. This includes context for how those metadata schemas are applied in practice.

Fundamentally, the choice to adopt semantic metadata is one that helps enable interoperability with existing and future data-driven solutions for smart buildings. Decisions made in relation to the adoption of semantic metadata and the selection of a metadata schema can also have a lasting impact on (i) IT system and data management solutions, (ii) available OT systems and solutions, and (iii) access to third party services and competition.

This report aims to provide essential background for the decision-making process by identifying (i) the prevailing approaches to building metadata and (ii) the key stakeholders and characteristics of dominant metadata schemas. This includes, amongst other things, qualifying metadata solutions by their desire for flexibility vs application focus and desire for breadth vs detail. No one schema is appropriate for all use-cases, and there can be benefit in utilising more than one schema to cover the needs of a building or portfolio of buildings.

Increasingly, the schemas detailed in this report are aligning to provide *harmonized* digital representations of buildings that support diverse perspectives and use cases. We hope this trend towards complementary rather than competing designs continues. In particular, we see RDF-based metadata schemas as demonstrating the highest degrees of interoperability and reusability compared to existing proprietary models.

Table of contents

Abbreviations

Abbreviations	Meaning
AEC	Architecture, Engineering and Construction
AHU	Air Handling Unit
AMS	Asset Management System
API	Application Programming Interface
ASHRAE	American Society of Heating, Refrigerating and Air-Conditioning Engineers
BACnet	Building Automation and Control networks
BAS	Building Automation System
BDNS	Building Device Naming Standards
BEO	Building Element Ontology
BIM	Building Information Modelling
BMS	Building Management System
BOT	Building Topology Ontology
BPO	Building Product Ontology
CDL	Control Description Language
CLA	Contributor Licence Agreement
COBie	Construction Operations Building Information Exchange
DB	Database
DBMS	Database Management System
DNAS	Drivers Needs Actions Systems
DOT	Damage Topology Ontology
EMIS	Energy Management Information System
EMS	Energy Management System
ETSI	European Telecommunications Standards Institute
FDD	Fault Detection and Diagnostics
FMIS	Facility Management Information System
FOG	File Ontology for Geometry formats
gbXML	GreenBuilding XML
GUID	Globally Unique IDentifier
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
HVAC	Heating, Ventilation, and Air-Conditioning
ID	Identifier
IEA	International Energy Agency
IETF	Internet Engineering Task Force
IFC	Industry Foundation Classes
I/O	Input/Output
IoT	Internet of Things

IP	Intellectual Property
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
JSON-LD	JavaScript Object Notation – Linked Data
LBD	Linked Building Data
MEP	Mechanical, Electrical, Plumbing
MQTT	Message Queue Telemetry Transpor
NBIMS	National BIM Standard
O&M	Operations and Maintenance
obXML	Occupant Behavior XML
OGC	Open Geospatial Consortium
oneM2M	One Machine to Machine
OPC-UA	Open Platform Communications - Unified Architecture
OWL	Web Ontology Language
OWL 2 RL	Web Ontology Language 2 Rules Logic
PH	Project Haystack
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
REC	Real Estate Core
R2RML	RDB to RDF Mapping Language
RML	RDF Mapping Language
SAREF	Smart Applications REFerence
SHACL	Shapes Constraint Language
SOSA	Sensor, Observation, Sample, and Actuator
SPARQL	SPARQL Protocol and RDF Query Language
SPF	STEP Physical File
SSN	Semantic Sensor Network
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UUID	Universal Unique Identifier
VAV	Variable air volume
VBIS	Virtual Buildings Information System
W3C	World Wide Web Consortium
XML	eXtensible Markup Language
ZINC	Zinc Is Not CSV

Definitions

BIM: Building Information Modelling provides a 3D digital representation of the building structure and the plant and equipment contained within. 4D, 5D, 6D and xD BIM extend basic design and construction to scheduling, cost estimation, facility management and performance evaluations, respectively. As of 2021, 6D and xD BIM use is limited, with most active BIM users typically found in the architectural, engineering and construction (AEC) professions.

BMS: A Building Management System is a combination of software, hardware and communications infrastructure, designed to support the building operation including the HVAC systems and subsystems such as fans, pumps and chillers. This is also referred to as a Building Automation System (BAS).

Cyber-physical System: a system with both physical and digital (cyber) components. The digital component contains intelligence that governs and/or informs the operation of the physical elements, often providing services such as network connectivity, data logging, and/or optimized operation.

EMS: An Energy Management System (also called an EMIS — Energy Management Information System) is a BMS system focused on monitoring, control and orchestration of large energy consuming devices within the building. When the focus is on whole-building energy management, the term Building Energy Management System (BEMS) is also used.

Information Model: a digital model that represents a collection of information, for example about a building (e.g. Building Information Model). A model follows a well defined schema.

Internet of Things: the collective name for the set of digital networked objects – *things* – that are embedded within sensors, actuators, and other products to enable the exchange of data and other information over the internet

Instance: The individual parts that together form a model, and that follow a particular schema (e.g. metadata schema).

Metadata: refers to 'data describing the context, content, and structure of records and their management through time'.

Metadata schema: defines the overall structure of metadata. It provides a labelling, tagging, or coding system used for describing and/or annotating data sets and data streams. The schema describes the structure of metadata, what values it contains, the relationships, and what concepts and constructs are part of the metadata.

Model: While there exist many interpretations and definitions for a "model", we try to use the term here to refer to the particular digital representation created for a specific system or building or installation. "Model" thus refers to all relevant data (instance data) for that particular system, building or installation — not a schema or high-level data model in this survey.

Telemetry: in situ measurements or other data at remote points and their automatic transmission to receiving equipment (telecommunication) for the purposes of monitoring, storage and further processing.

1. Introduction

1.1 What is a data-driven smart building?

The *Internet of Things* and modern trends towards digitization have dramatically increased the quantity of available data on modern *cyber-physical systems*, including buildings - mirroring the trend towards digitalization of business processes in the broader economy. This has begun to enable new types of data-driven processes in buildings, automating and replacing many of the traditionally manual tasks carried out by domain practitioners. These tasks - including *fault detection and diagnosis, optimised control strategies* and sequences of operations, performance measurement, benchmarking, and energy auditing - enable buildings to be more efficient and resilient in their operation and more comfortable and reactive to their occupants.

However, there are fundamental barriers to realising this vision of data-driven building processes (Fig. 1). Although there is a wealth of building *telemetry* available, the data-driven processes themselves are characterised by a need for detailed information about the internal structure and composition of building subsystems. Buildings are heterogeneous, complex systems consisting of diverse arrays of equipment; hence, proper use of data requires that data be *contextualised* in terms of how it relates to the operation of the building. Contextualising data is the role of a *metadata schema* – an organizational structure that governs how data can be described and related to the structure, composition, and topology of a building. We focus our survey on metadata schemas that enable data management, rather than existing protocols and standards that handle the exchange of the data itself.

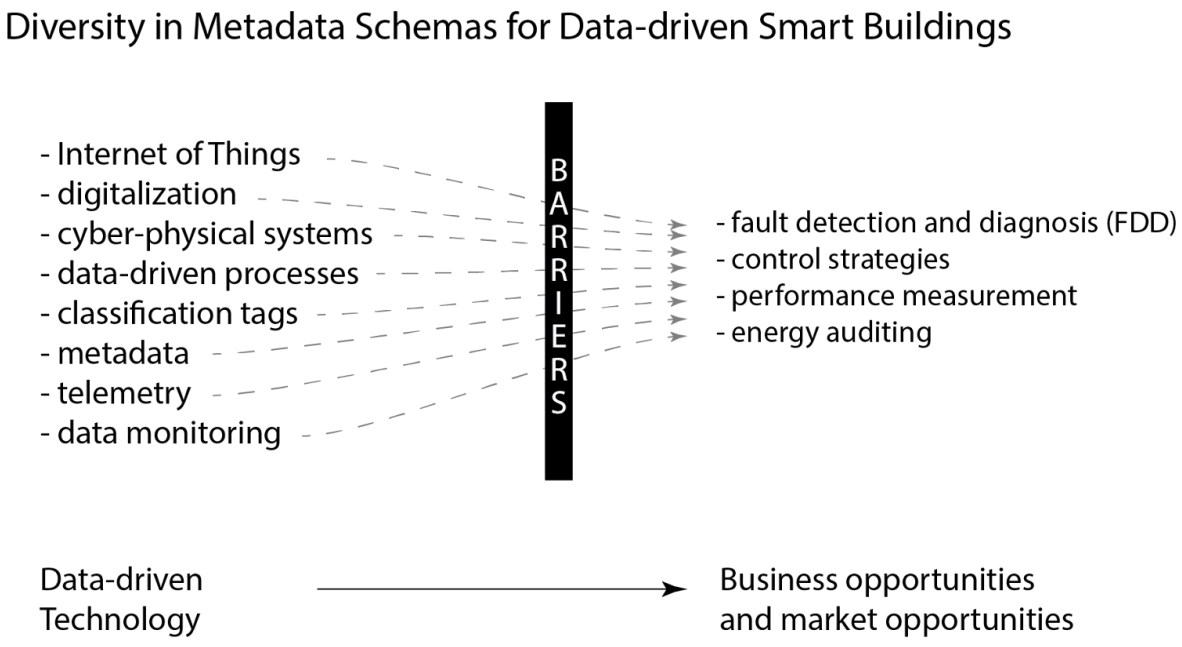


Figure 1: Diversity in metadata schemas and unclarity about their technical potentials make it difficult to enable clear business opportunities from available data and technologies in smart buildings.

This survey presents metadata schemas that enable or contribute to the realisation of data-driven buildings. **A *data-driven building* is one whose processes are automated and driven by the use of historical and/or live building telemetry and which receives digital commands through such a data-driven process.** These processes not only include data analytics and monitoring, but also decision processes that

affect the functionality of the building. In essence, a data-driven smart building is one that uses data to provide feedback on building operation to inform automated or human-driven decision-making processes.

Keeping in line with the scope of Annex 81, the survey focuses on metadata schemas enabling data-driven processes for the *operations and management stage* of the building lifecycle. This includes the asset management phase, in which predictive maintenance of building equipment and components is required. Our survey excludes digital representations of the building which are used solely or predominantly for the *design and construction of the building* (AEC industry),but we recognize that these AEC representations may aid in the population and maintenance of metadata for smart building operation.

1.2 Key criteria for metadata schemas for data-driven smart buildings

Considering all the above, key criteria that we will rely on in this survey are:

1. **Support for storage of data rather than semantics.** Our definition of *data* in *data-driven processes* incorporates empirical, measured, sampled data, time-series, and derived streams.
2. **Focus on cyber-physical data for building operation rather than occupant-related data.** To constrain the scope of the survey, we do not consider metadata schemas which center occupant or tenant data as these cannot directly inform building operation.
3. **Focus on Energy Management data.** While other data is available and considered to a lesser extent, such as access control data, fire safety measures, security systems, etc., the primary focus of this document is on energy data (BMS, EMS) in line with the scope of the IEA.
4. **Focus on the operational phase**, which excludes simulation models and data that would normally be used in the design and engineering phase (e.g., IFC, gbXML, BIM)
5. **Focus on the management of data**, and less on the clear intervention inside buildings through actuators, control logic, and control systems. Although protocols and algorithms are important in a smart building (Wetter et al., 2018, 2022[2]), e.g. BACnet or Control Description Language (CDL)[3], they are considered out of scope for the survey that focuses on metadata for real-time and historical data in buildings.

1.3 Reading Guide

This document has **four main sections** beyond the introduction, as shown in Fig. 2. In Section 2, this document gives an overview of available metadata schemas, considering the given criteria. This section only focuses on the core metadata schemas considered in scope, and not on the more remote schemas that may be of reference and are relevant (e.g. BIM schemas, simulation models, control model schemas). These tangentially relevant metadata schemas are listed under related works. In Section 3, a qualitative technical comparison is made between the core metadata schemas, following six comparison criteria defined as part of this section: model structure, completeness, and formal rigour, alignments, position in reference system architecture, required tooling, and creation and maintenance of models. Section 4 reviews additional schemas and models (related works), while Section 5 concludes this survey with a summary of findings and recommendations.

[2] Wetter, Michael, Paul Ehrlich, Antoine Gautier, Milica Grahovac, Philip Haves, Jianjun Hu, Anand Prakash, Dave Robin, Kun Zhang. "OpenBuildingControl: Digitizing the control delivery from building energy modeling to specification, implementation and formal verification" *Energy* 238, Part A, no 121501: 2022. https://www.sciencedirect.com/science/article/pii/S0360544221017497
Wetter, Michael, Milica Grahovac, Jianjun Hu. "Control Description Language", Proceedings of The American Modelica Conference 2018. pp. 17-26. http://dx.doi.org/10.3384/ecp1815417.
[3] https://obc.lbl.gov/specification/cdl.html

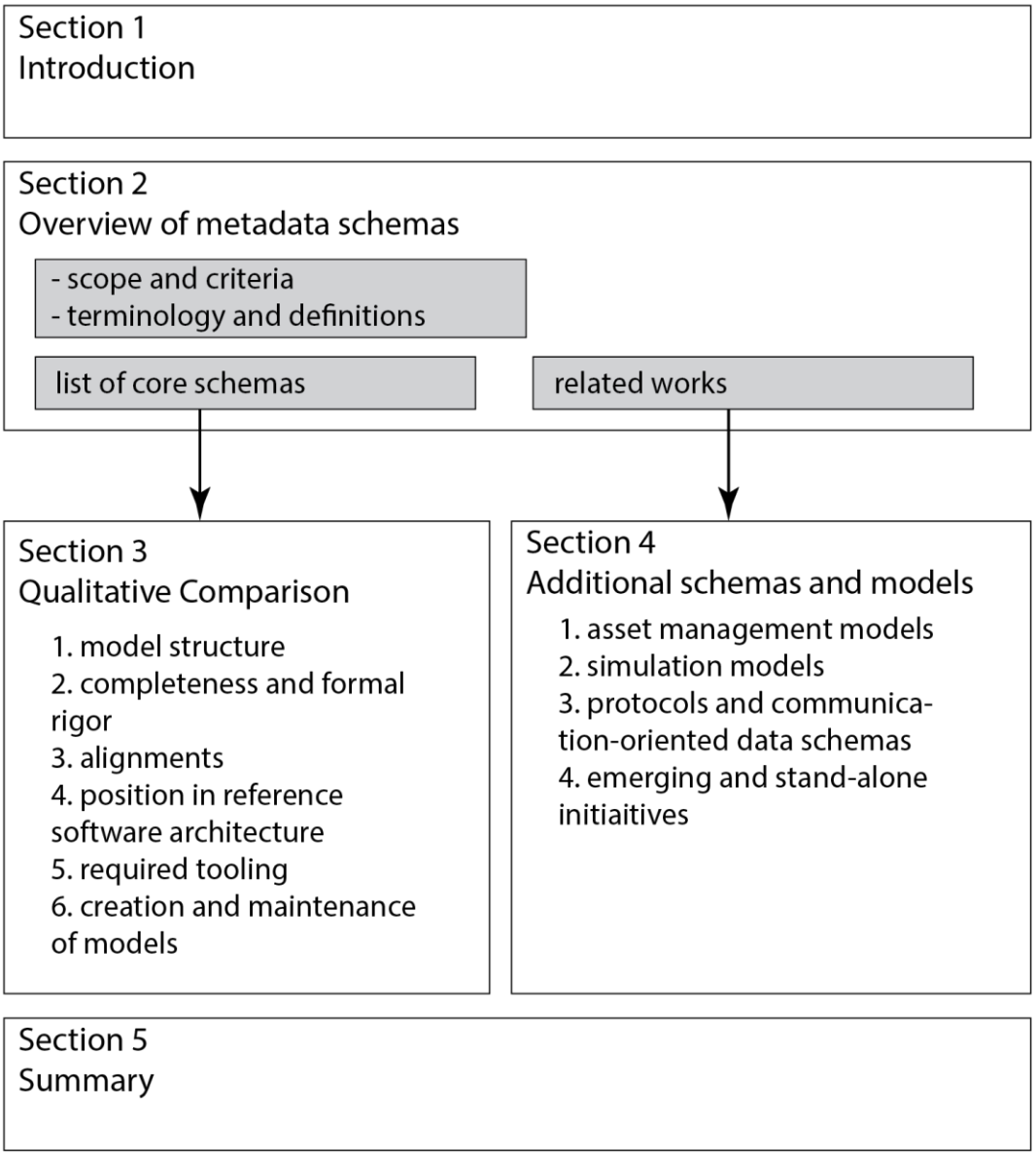


Figure 2: Survey structure.

2. Overview of metadata schemas

Data-driven buildings typically have sensor data streams at their core, which need to be annotated and given meaning. This can be done by attaching metadata to these data streams, which is slightly different from ontologies and vocabularies that fully start from semantic models that are enriched by pointers to sensor data streams. For simplicity, we refer to these as '**metadata schemas**', and not 'vocabularies', 'ontologies', 'metadata models' or 'schemas', because they incorporate solutions for organising building data, as well as, more traditional formal linked data models. This section lists and motivates the inclusion of the set of metadata schemas in this survey.

For each metadata schema, we enumerate the **purpose, origin, licensing and governance of the metadata schema**. This initial description includes a summary of known deployments or uses of the metadata schema, as well as how the originators or maintainers of the model see the model as contributing to a vision of data-driven buildings. The second part of the survey will **discuss** each of the salient **dimensions or properties of the metadata schemas** that are relevant to data-driven buildings and **compare and contrast** the different approaches.

2.1 Scope and criteria

We now describe the **qualifications for inclusion** in the metadata schema survey. To be included, a metadata schema must:

1. be relevant to the vision of enabling *data-driven smart buildings*, in terms of data management
2. have an *active authority* and owner in charge of further developments and maintenance
3. have experienced *adoption* by organisations or individuals outside of the maintainers of the model
4. be "*open*" in the sense of being permissively licensed, open-source or otherwise widely available
5. target the *operational phase* of the building life cycle

The set of metadata schemas included in this survey were narrowed down from a search of recent (last 10 years) academic literature, building industry publication venues, and the collective experience of the authors. We **categorise** the chosen metadata schemas by:

1. the extent to which *they "touch data*", meaning they are directly involved in the management, description or organisation of building telemetry (e.g. sensor data, equipment operational status or setpoint, etc)
2. the extent to which they describe the structure and composition of buildings, which can be used to *contextualise* data
3. *their nature* and key way of working / purpose, for example whether they target and enable mainly annotation and classification, or rather on semantic modelling.

2.2 List of core metadata schemas

In our summary and review, we discuss the following **core metadata schemes**.

1. Project Haystack

2. Brick Schema
3. Real Estate Core (REC)
4. BOT ontology and Linked Building Data (LBD)
5. SAREF (SAREF4BLDG)
6. SOSA / SSN
7. Google Digital Buildings

2.2.1 Project Haystack (PH)

Project Haystack is a project that has built semantic data models and web services to encode the data streams generated by smart devices in buildings. This project has a number of technical solutions that can be used at will. It focuses on adding meaning to data streams using tags and flexible annotations.

Developers: Developed by Project Haystack (non-profit organisation: 501c6 incorporated in the US).

Maintenance Model: Version 4 of Project Haystack is developed on GitHub. The definition of the metadata schema, build scripts and resulting documentation are hosted in a GitHub repository[4]. Edits to PH are performed through merging of Pull Requests or, more commonly, are requested through discussion on the GitHub issue tracker or the PH forum. Informal Working Groups[5] around specific features and topics are organised through the forum.

Governance Process: No official bylaws are available on the PH website. Direction for PH is set at Haystack Connect conference, with working groups attended by contractors and consultants who use PH in their work. PH has a board of directors represented by various companies in the HVAC industry.

Licensing Model: open source under the Academic Free Licence 3.0[6]

Key Dates: Founded in 2014

Regularity of Updates: PH adopts a rolling release model between major versions: bug fixes and small improvements or changes are merged directly into the latest online documentation and downloadable releases.

Required tooling, software, dependencies:

- Python libraries for Haystack (available tooling)
- ZINC-specific language -> parser
- Oriented towards SkyFoundry SkySpark (dependency)

Known/Existing Deployments: Deployed widely in modern BMS systems throughout the world since 2015, principally with product vendors such as Tridium/Honeywell, EasyIO and others. Principal reason for use is to create a metadata schema for the FDD software *SkyFoundry SkySpark*, and derivative products. While PH is free and open-source, *SkyFoundry SkySpark* is a commercial, proprietary software solution supported by a licensing model based on the number of points monitored. Instances of SkyFoundry SkySpark are typically owned and operated by independent HVAC controls contractors or consultants.

Support for Data-Driven Smart Buildings: PH defines a data model for exchange of building/IoT data. A PH instance consists of a set of linked key-value documents each corresponding to a site, space, equipment, point, device or a number of other entity types. Point documents, which represent data sources, can contain a reference to their current value or an annotation that historical data is available. Version 4 of PH

[4] https://github.com/Project-Haystack/haystack-defs
[5] https://project-haystack.org/forum/wg
[6] https://project-haystack.org/doc/docHaystack/License

incorporates some ontology features that provide relations between objects. Access to telemetry and the instance is performed by sending queries to an HTTP API server; queries against the metadata instance can resolve to historical telemetry or real-time data subscriptions.

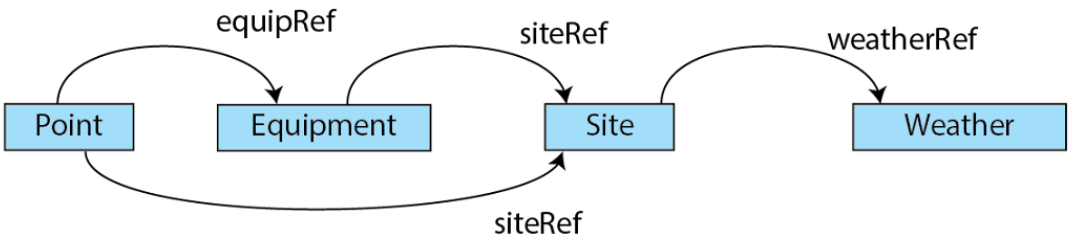


Figure 3: Backbone structure for a PH dataset (image in Pauwels et al., 2022[7]).

Example:

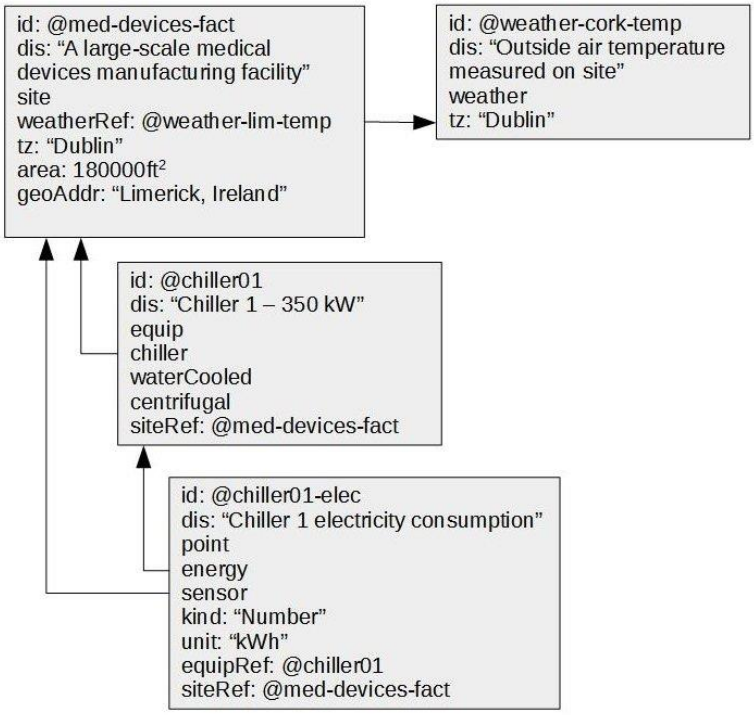


Figure 4: PH Example[8].

```
id:@24192ca1-0c85f75d "Headquarters"
site
area:140797ft²
tz:New_York
dis:Headquarters
geoAddr:"600 W Main St, Richmond, VA"
geoCoord:C(37.545826,-77.449188)
hq
metro:Richmond
primaryFunction:Office
yearBuilt:1999
```

```
_:24192ca1-0c85f75d
    a phIoT:site ;
    ph:hasTag site,
    phIoT:area 140797 ;
    ph:tz "New_York" ;
    ph:dis "Headquarters" ;
    ph:geoAddr "600 W Main St, Richmond, VA" ;
    ph:geoCoord "C(37.545826,-77.449188)" ;
    phIoT:primaryFunction "Office" ;
    phIoT:yearBuilt 1999 .
```

Figure 5: Example of JSON notation for PH dataset[9].

[7] Pauwels, P., Costin, A., Rasmussen, M.H. (2022). Knowledge Graphs and Linked Data for the Built Environment. In: Bolpagni, M., Gavina, R., Ribeiro, D. (eds) Industry 4.0 for the Built Environment. Structural Integrity, vol 20. Springer, Cham. https://doi.org/10.1007/978-3-030-82430-3_7.
[8] https://web.archive.org/web/20210908002902/
[9] https://project-haystack.org/example/bravo

2.2.2 Brick Schema

Brick is an open-source effort to standardise semantic descriptions of the physical, logical and virtual assets in buildings and the relationships between them. Brick consists of an extensible dictionary of terms and concepts in and around buildings, a set of relationships for linking and composing concepts together, and a flexible data model permitting seamless integration of Brick with existing tools and databases. Brick metadata schema uses the resource description framework (RDF) ontology that provides relations between objects in a subject-predicate-object (graph) model.

Developers: Brick was originally developed by representatives of UC Los Angeles, UC Berkeley, UC San Diego, Carnegie Mellon University, University of Virginia, University of Southern Denmark and IBM Research. It is now developed and maintained by the Brick Consortium (non-profit organisation; 501c6 incorporated in the U.S.fty).

Maintenance Model: Brick is developed on GitHub. The definition of the ontology and build scripts are hosted in a GitHub repository[10]. Documentation and other tools are also hosted in the BrickSchema GitHub organisation[11]. Changes to Brick are requested through the issue tracker or mailing list and are performed by merging Pull Requests to the primary Brick repository. The Brick Consortium organises 3 public working groups around specific ongoing technical tasks: Data Working Group (creating and curating public data sets), Ontology Working Group (furthering development of the ontology) and Tooling Working Group (developing open tools for the community).

Governance Process: The bylaws for the Brick consortium are available online[12]; they follow an IETF-style IP disclosure process in order to avoid the inclusion of protected intellectual property polluting the ontology and causing issues for adoption. Working groups are public and open to all individuals. Formal members of the Brick consortium may participate in and vote for members of various committees. The Technical Committee oversees development of the Brick ontology and votes to approve new minor and major releases.

Licensing Model: open source under the BSD-3-Clause licence[13]

Key Dates: Original ontology published in 2016; consortium launched in 2021

Regularity of Updates: Brick releases a minor version update roughly every 6 months, with patch versions being made available on a rolling basis. Rolling changes are made to the primary branch on GitHub and are made available as “Brick Nightly”[14].

Required tooling, software, dependencies: Brick relies on general purpose tools, such as triple stores, ontology editors, query engines, and so forth. It has little to no dependencies on available tooling.

Known/Existing Deployments: Brick has been deployed in several data platforms, including Mortar[15] and the Data Clearing House[16]. In addition, the Brick consortium counts Carrier, Johnson Controls, Clockworks Analytics, Mapped and Schneider Electric as members.

Support for Data-Driven Smart Buildings: Brick defines a data model for describing data sources in buildings and their characteristics and context. An instance of Brick is a directed labelled graph where nodes represent physical, virtual and logical entities (including equipment, locations and points) and edges represent directional relationships between entities (RDF graph). Point instances in a Brick graph can contain

[10] https://github.com/BrickSchema/brick/
[11] https://github.com/BrickSchema/
[12] https://brickschema.org/consortium
[13] https://github.com/BrickSchema/Brick/blob/master/LICENSE
[14] https://github.com/BrickSchema/Brick/releases/tag/nightly
[15] https://mortardata.org/
[16] https://www.dataclearinghouse.org/

"foreign key" properties that relate a data source to where the data may be found: historical storage, location in a BMS network or other digital or cyber-physical locations. Applications describe the properties and features of relevant data using queries against a Brick graph. The evaluation of those queries returns to the application the metadata required to fetch or subscribe to the telemetry.

Example:
Reference brick models are found at the Brick website. Figure 6 below illustrates a basic Brick model encapsulating an AHU, two VAVs, and a handful of points and rooms.

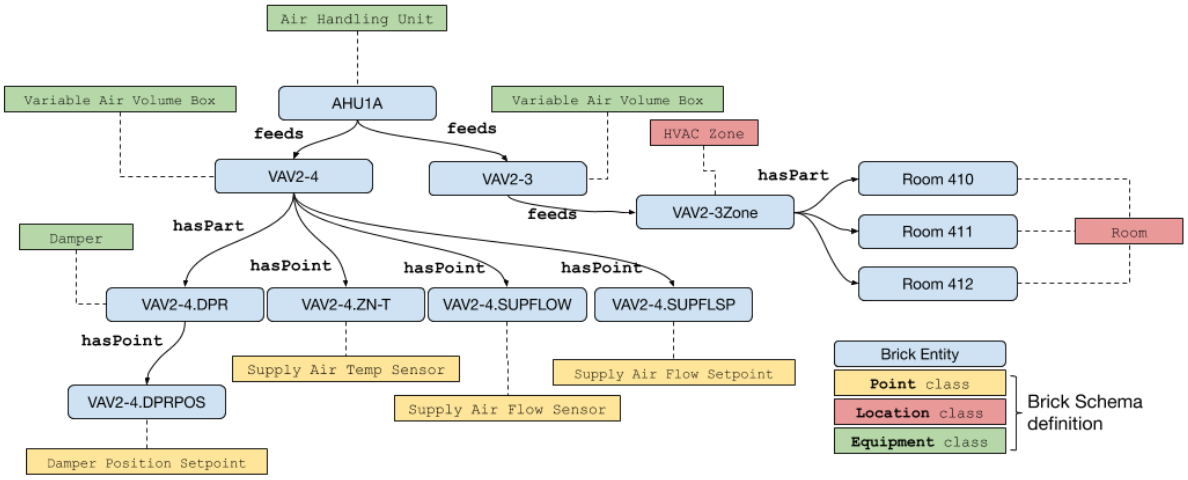


Figure 6: Example of Brick dataset (image from BRICK website[17]).

2.2.3 Real Estate Core (REC)

RealEstateCore is a metadata schema that is mostly focused on asset and property management. It is a modular ontology that consists of several smaller data schemas that describe concepts and relations for modelling buildings and building systems. RealEstateCore is not aiming to replace all standards, but rather intends to bridge existing standards and find the common denominators. RealEstateCore uses and maps existing standards in a pragmatic manner. RealEstateCore focuses on merging and bridging four domains: Business administration, Digital representation of the building's elements – BIM, Control and operation of the building BMS, IoT technologies. REC is an ontology defined in the resource definition framework (RDF).

Developers: REC is developed by the RealEstateCore Consortium, which was founded by Vasakronan AB, Akademiska Hus AB, Idun Real Estate Solutions AB, Willhem AB, RISE, and the School of Engineering at Jönköping University.

Maintenance Model: REC is developed on GitHub. The ontology definitions and documentation are hosted in a GitHub repository[18]. Changes to REC are requested through the issue tracker or Gitter chat channel and are performed by merging Pull Requests.

Governance Process: No official bylaws are available on the REC website. According to the primary publication by Hammar et al. (2019)[19], membership is 1000 Euro for corporations and free for everyone else.

Licensing Model: open source under the MIT licence[20]

[17] https://brickschema.org/
[18] https://github.com/realestatecore/rec
[19] Hammar, K., Wallin, E O., Karlberg, P., Hälleberg, D. (2019) The RealEstateCore Ontology In: C. Ghidini, O. Hartig, M. Maleshkova, V. Svátek, I. Cruz, A. Hogan, J. Song, M. Lefrançois & F. Gandon (ed.), The Semantic Web – ISWC 2019: 18th International Semantic Web Conference, Auckland, New Zealand, October 26–30, 2019, Proceedings, Part I (pp. 130-145). Cham: Springer Lecture Notes in Computer Science https://doi.org/10.1007/978-3-030-30796-7_9
[20] https://github.com/RealEstateCore/rec/blob/master/LICENSE.txt

Key Dates: Consortium founded in 2017

Regularity of Updates: REC experiences regular updates; a new minor version is released roughly every 6 months.

Required tooling, software, dependencies: Because REC is an RDF-based ontology, it is compatible with the majority of RDF tooling including triplestores and SPARQL query processors. In addition, REC also defines an OpenAPI specification for interacting with a REC model over HTTP.

Known/Existing Deployments: REC is used in a number of software platforms, in particular in the Azure Digital Twin system that is made available also through Willow.Inc . IDUN also has a REC-compliant platform called ProptechOS.

Support for Data-Driven Smart Buildings: REC implements a modular approach to ontology design: it defines many different components which can be combined to support the desired features. Several REC modules provide metadata for capturing the context of data and relevant features of the building and environment: Core, Agents, Building, Lease and Device. The Device, DataSchema and Actuation modules are relevant to data-driven buildings. Similar to Project Haystack and SSN/SOSA, REC supports embedding the present value of a point in the graph instance. REC also models digital interfaces to devices and actuators; the DataSchema module implements a basic Interface Description Language which describes the structure of the data to and from a given device. REC also supports storing BACnet, Modbus, KNX and other digital protocol configurations to enable interacting with digital building networks. Currently, REC does not directly support storing references to historical telemetry.

Example:

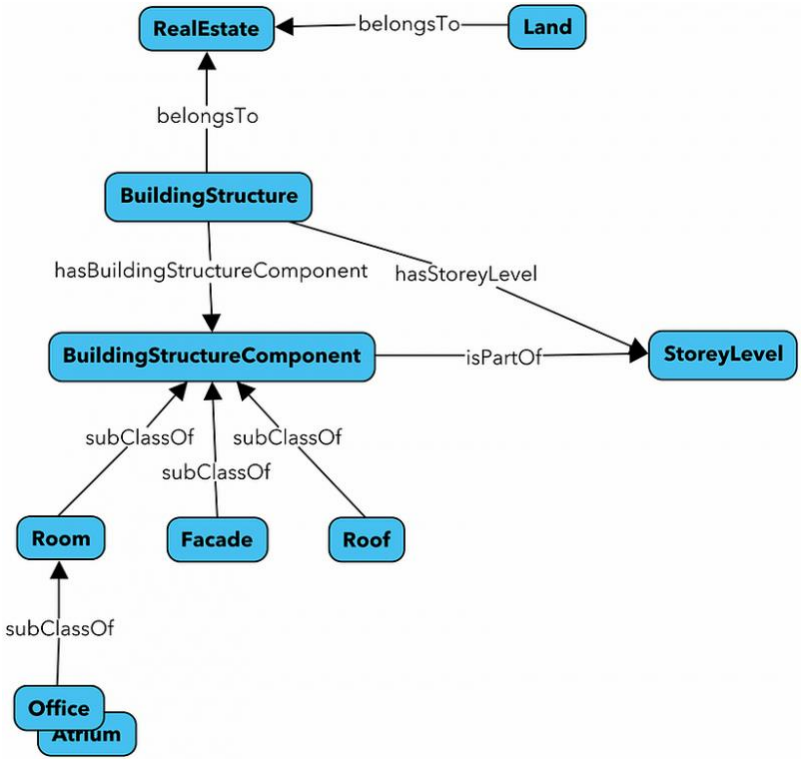


Figure 7: Core structure for REC metadata schema.

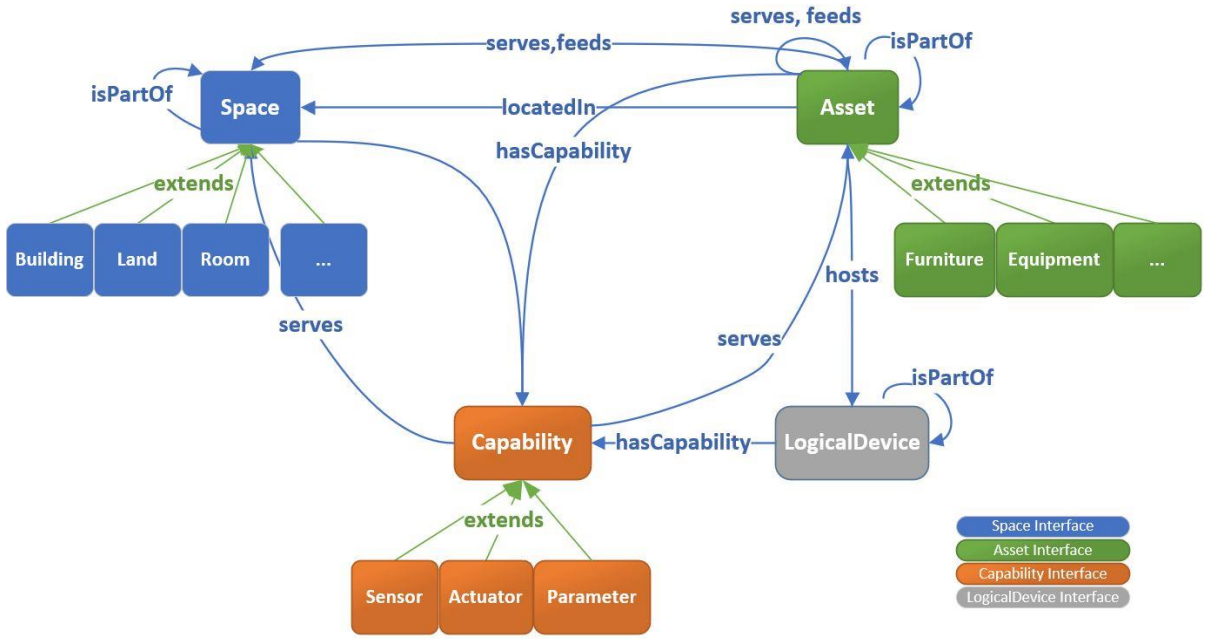


Figure 8: Representation of Space, Asset, Capability using REC metadata schema (image from the REC website[21]).

2.2.4 BOT ontology and Linked Building Data (LBD)

The Linked Building Data community group aims to make building data accessible over the web in a well-structured manner. This includes a BOT ontology that captures the building topology. This can be extended using several ontologies, which can include vocabularies to represent building systems, 3D geometry, telemetric data, and classification systems (tags). BOT ontology is defined in the resource description framework (RDF).

Developers: The BOT ontology was originally created by Mads Holten Rasmussen et al. (2017) [22], and is currently maintained by the W3C LBD Community Group[23]. Input for the BOT ontology has been provided by Pieter Pauwels, Maxime Lefrancois, and Georg F. Schneider[24]. It forms a part of an ecosystem of ontologies, which are lightly interlinked, and which are typically referred to as LBD ontologies (see diagram below).

Maintenance Model: The BOT ontology is maintained on GitHub[25] (last release Sept. 2020). The ontology definitions and documentation are hosted in a draft W3C report[26]. Changes to the BOT ontology can be suggested through the issue tracker on GitHub and are typically discussed first within the W3C LBD Community Group in order to achieve consensus and agreement with its owners.

Governance Process: No official bylaws are available on the BOT or LBD websites.

Licensing Model: The ontology is open source available, under the Creative Commons 4.0 licence[27], yet the work is also copyrighted under the W3C Community Contributor Licence Agreement[28] (CLA).

Key Dates: First publication and first release in 2017. A revised version, release and publication in 2020.

---

[21] https://www.realestatecore.io/getting-started/
[22] Mads Holten Rasmussen, Pieter Pauwels, Christian Anker Hviid and Jan Karlshøj (2017). **Proposing a Central AEC Ontology That Allows for Domain Specific Extensions**. *Lean and Computing in Construction Congress (LC3): Volume I Ð Proceedings of the Joint Conference on Computing in Construction (JC3), July 4-7, 2017, Heraklion, Greece, pp. 237-244*, https://dx.doi.org/10.24928/JC3-2017/0153
[23] https://www.w3.org/community/lbd/
[24] Rasmussen, M. H., Lefrançois, M., Schneider, G. F., & Pauwels, P. (2021). BOT: The building topology ontology of the W3C linked building data group. *Semantic Web*, *12*(1), 143-161. https://doi.org/10.3233/SW-200385.
[25] https://github.com/w3c-lbd-cg/bot
[26] https://w3c-lbd-cg.github.io/bot/#
[27] https://creativecommons.org/licenses/by-sa/4.0
[28] https://www.w3.org/community/about/agreements/cla/

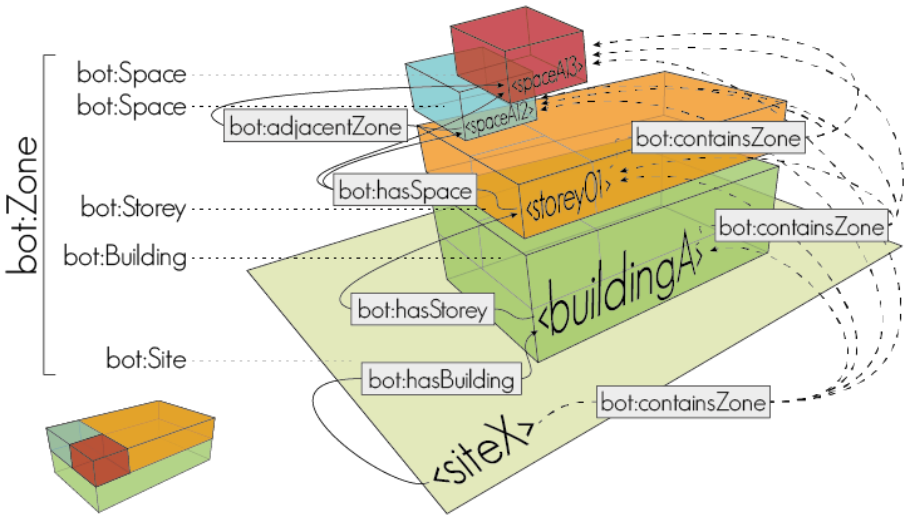
Regularity of Updates: BOT aims to be stable, releases are limited. Most effort currently goes to the definition and updates of the aligned ontologies (building systems, geometry, building elements, products, properties, etc.).

Required tooling, software, dependencies: Because BOT is an RDF-based ontology, it is compatible with the majority of existing semantic web tooling including triplestores and query processors. In addition, the LBD community has developed several tools for importing and exporting LBD models from IFC[29] and Revit, a popular commercial BIM tool.

Known/Existing Deployments: BOT is a part of set of LBD ontologies with an expanding size (e.g. damage ontologies, sensor-oriented ontologies, product ontologies, classification ontologies, 3D geometry, etc.). The focus is on the representation of building data at large. Existing deployments are situated mainly in the AEC industry, with diverse exporters and importers from and to BIM modelling software (Revit and IFC predominantly). No public list of commercial deployments is currently available. Yet, stand-alone tools are developed for building data management; and several ongoing ontology developments are inspired by the BOT ontology and LBD ontologies at large.

Support for Data-Driven Smart Buildings: The BOT ontology is a minimal and central ontology that is easy to use to define a building from scratch using a limited number of required classes (Fig. 9). This allows defining the topology of a building (spaces, zones, elements, interfaces) relatively quickly, which is the main purpose of the BOT ontology. It is crucially part of a set of LBD ontologies, which allow to further extend the knowledge representation for specific buildings in a relatively independent manner (modular and extensible). This includes for example BEO, MEP, DOT, BPO and FOG ontologies. Time series data are made available in LBD graphs (1) either as triples, e.g. through the use of the SSN/SOSA ontologies[30], or (2) as references to specific Ids in time series databases[31]. This allows the use of dedicated technologies (machine learning, statistics) for analysis of data from data-driven smart buildings.

Example:
A reference example is available at in the OpenSmartHomeData Github repository[32].



[29] Mathias Bonduel, Jyrki Oraskari, Pieter Pauwels, Maarten Vergauwen, Ralf Klein. The IFC to linked building data converter - current status. Proceedings of the 6th Linked Data in Architecture and Construction Workshop. London, UK, 2018. Pp. 34-43. https://ceur-ws.org/Vol-2159/04paper.pdf
[30] https://github.com/TechnicalBuildingSystems/OpenSmartHomeData
[31] Petrova, E. A., Pauwels, P., Svidt, K., & Jensen, R. L. (2019). In Search of Sustainable Design Patterns: Combining Data Mining and Semantic Data Modelling on Disparate Building Data. In I. Mutis, & T. Hartmann (Eds.), *Advances in Informatics and Computing in Civil and Construction Engineering: Proceedings of the 35th CIB W78 2018 Conference: IT in Design, Construction, and Management* (pp. 19-26). Springer. https://doi.org/10.1007/978-3-030-00220-6_3
[32] https://github.com/TechnicalBuildingSystems/OpenSmartHomeData

Figure 9: BOT allows the representation of the topology of a building: Site, Building, Storey, Space, Element, Interface.

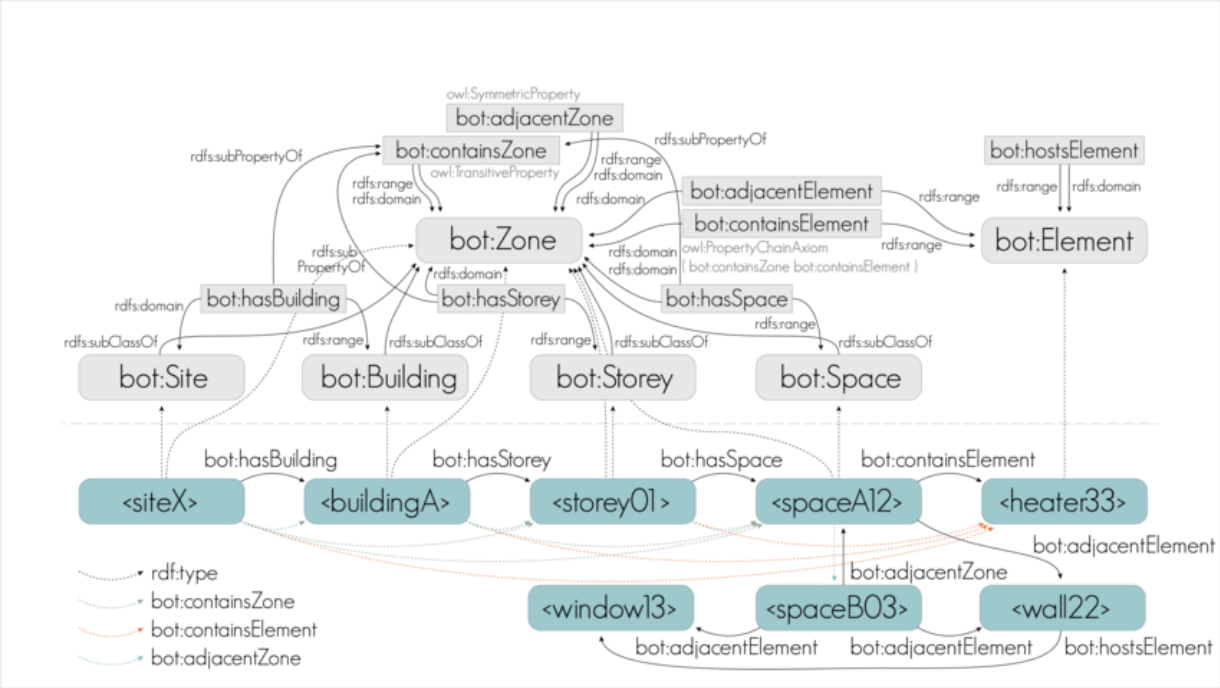


Figure 10: Example use of BOT for an example building.

2.2.5 SAREF (SAREF4BLDG)

The Smart Applications REFerence (SAREF) ontology is a shared model of consensus that facilitates the matching of existing assets (standards, protocols, data models, etc.) in the smart appliances domain. The SAREF ontology provides building blocks that *allow separation and recombination of different parts of the* ontology depending on specific application needs.

Developers: The SAREF ontology is originally created by a team at TNO in the Netherlands[33]. It has been standardised and maintained under ETSI – contributors include L. Daniele (TNO), R. Garcia-Castro and M. Poveda-Villalon (Universidad Politécnica de Madrid) and Maxime Lefrançois (MINES Saint-Étienne). All development of SAREF was informed and advised upon by a technical and domain expert group and consolidated knowledge acquired over several research and development projects.

Maintenance Model: The SAREF core ontology and its domain extensions are available on a local GitLab account[34]. The ontology definitions and documentation are hosted in the ETSI web portal for SAREF[35].

Governance Process: SAREF is governed under the ETSI standardisation body.

Licensing Model: SAREF is licensed under an ETSI licence[36] that points to a BSD-3-Clause open source licence.

Key Dates:

- First documentation[37]: 2013
- Last publication date: 2020-02-11

[33] https://sites.google.com/site/smartappliancesproject/project-team
[34] https://labs.etsi.org/rep/saref (last activity Aug. 2021)
[35] https://saref.etsi.org/
[36] https://forge.etsi.org/etsi-software-license
[37] https://sites.google.com/site/smartappliancesproject/documents

Regularity of Updates: Updates to SAREF ontologies are made regularly and has been extended to 11 relevant domains.

Required tooling, software, dependencies: Because the SAREF ontologies are built in RDF, they are compatible with the majority of existing RDF tooling including triplestores and query processors.

Known/Existing Deployments: A number of European Research projects contributed to the development and use of the SAREF core ontology and its domain extensions. The mapping of SAREF to the oneM2M core ontology, has resulted in testing and deployment in the context of oneM2M open source implementations.

Support for Data-Driven Smart Buildings: The SAREF core ontology is a general purpose ontology, focusing on the concept of device, of which the main schema can be seen below. It has been extended with 11 domain extensions, including SAREF4BLDG, SAREF4ENER and SAREF4CITY. Through the SAREF4BLDG ontology module[38], for example, a connection is made with BIM models, and in particular the relevant appliance classes and properties in IFC.

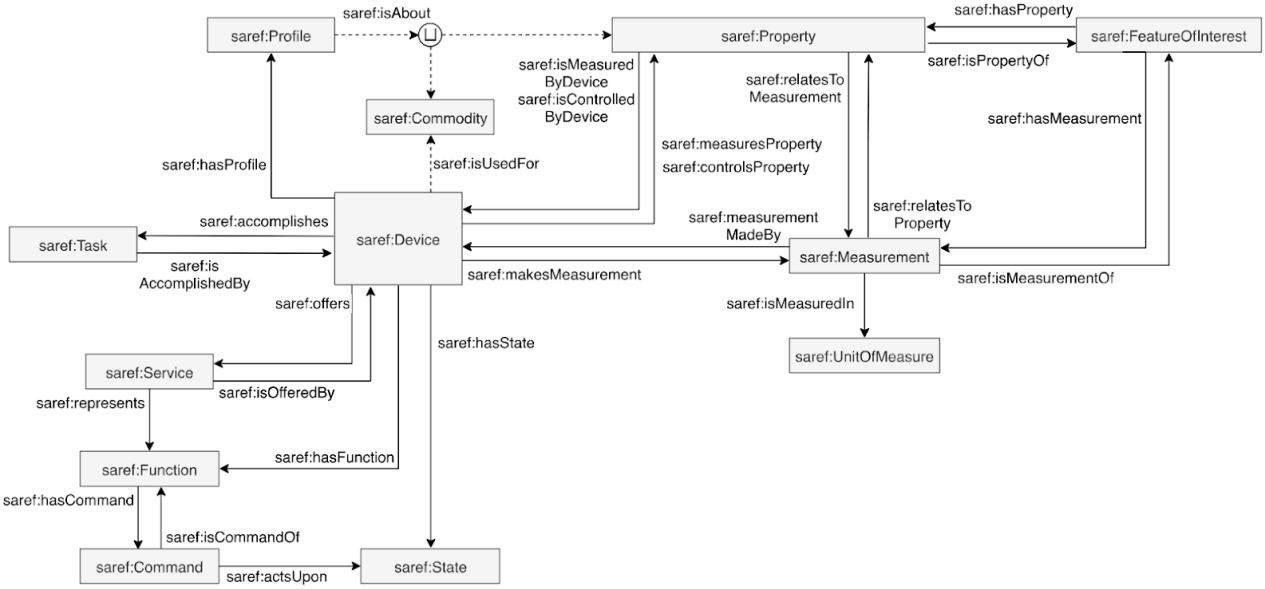


Figure 11: Part of the SAREF metadata structure that shows how Feature of Interest can be defined in relation to measurements and devices with specific services and commands and states (image from SAREF website[39]).

[38] https://saref.etsi.org/saref4bldg/v1.1.2/
[39] https://saref.etsi.org/

Example:

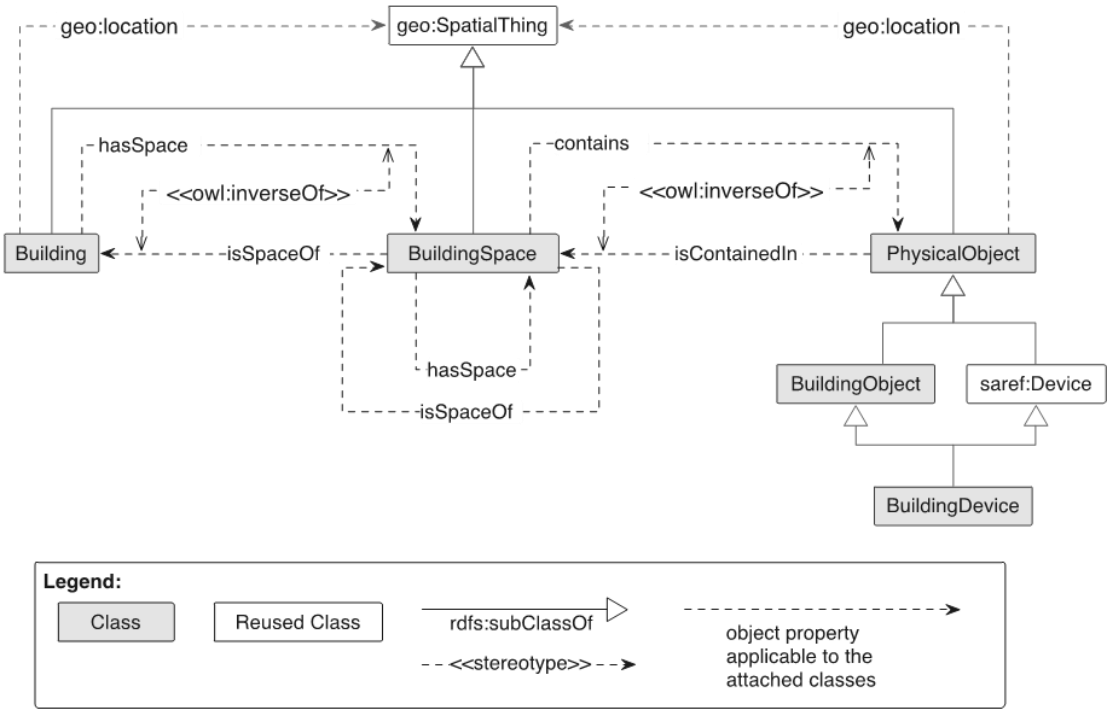


Figure 12: The topology structure used in SAREF as a composition of Building, BuildingSpace, PhysicalObject, and a set of subclasses under BuildingDevice (image from SAREF4BLDG documentation[40]).

2.2.6 SOSA and SSN

The Semantic Sensor Network (SSN) ontology is an ontology for describing sensors and their observations, the involved procedures, the studied features of interest, the samples used to do so, and the observed properties, as well as actuators. SSN follows a horizontal and vertical modularization architecture by including a lightweight but self-contained core ontology called SOSA (Sensor, Observation, Sample, and Actuator) for its elementary classes and properties.

Developers: SOSA/SSN is developed by the W3C and OGC organisations, originally by the Semantic Sensor Network Incubator Group and developers of OGC's Sensor Web Enablement. Further details on the history are provided in the SSN documentation[41].

Maintenance Model: SOSA/SSN is developed on GitHub via the Spatial Data on the Web Working Group[42]. Contributions are proposed as pull requests and later discussed and merged into subsequent releases.

Governance Process: SOSA/SSN is governed under the W3C policies for participation.

Licensing Model: SOSA/SSN is licensed under both the OGC licence which borrows from Apache 2.0 and the W3C licence[43]. Both are permissive open-source licences.

Key Dates: The initial version of SOSA/SSN was released in 2005.

Regularity of Updates: Regular updates have been conducted since the initial release with the latest in 2017. Updates are not as often as with other ontologies because SOSA/SSN is intended to be a stable ontology on which other more specific ontologies may be developed.

[40] https://saref.etsi.org/saref4bldg/v1.1.2/
[41] https://www.w3.org/TR/vocab-ssn/#Developments
[42] https://github.com/w3c/sdw
[43] https://www.w3.org/Consortium/Legal/2015/copyright-software-and-document

Required tooling, software, dependencies: Because the SOSA and SSN ontologies are built in RDF, they are compatible with the majority of existing RDF tooling including triplestores and query processors.

Known/Existing Deployments: SOSA/SSN are used widely to model the observations and actuations for sensors and actuators. Their concepts are explicitly incorporated into or are designed to complement existing ontologies such as Brick and SAREF4BLDG.

Support for Data-Driven Smart Buildings: SOSA/SSN support data-driven smart buildings by modelling the relationship between observable and actuable properties of the physical world, how those properties are observed and actuated, and what the observed or actuated values are. These perspectives are often implicit or ignored in many other systems and ontologies; by modelling them it is possible to remove ambiguity in, for example, the difference between the location of a sensor and the location of the property it is observing. Having this level of detail is important for advanced data-driven use cases.

Example:

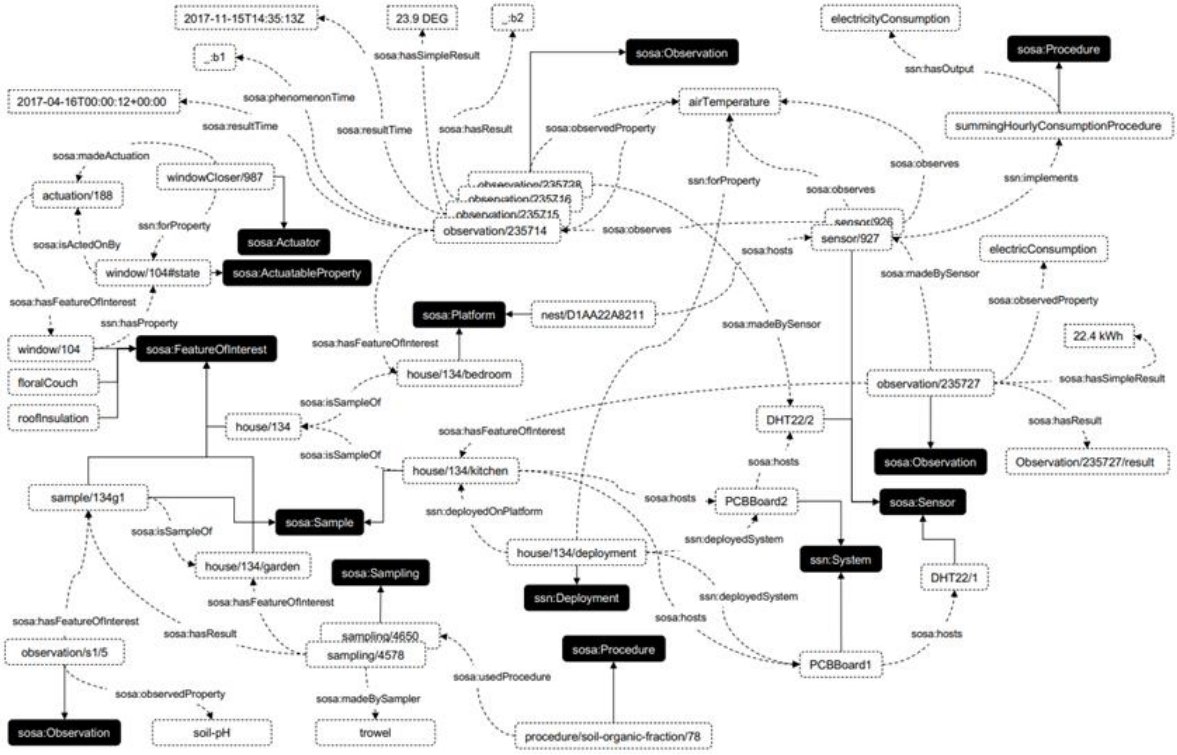


Figure 13: Overview example of how building data can be connected to observation and sensor data using SOSA/SSN (image from Haller et al., 2018[44]).

2.2.7 Google Digital Buildings

Google Digital Buildings is a data schema and toolset for representing structured data about buildings and equipment in buildings. It is being used internally at Google to manage the buildings in their portfolio. The metadata model is based around naming collections of points which are required for different applications; these collections can be composed to support larger suites of applications. The model captures minimal metadata on the topology of the building systems themselves. An OWL ontology of the model is made available.

[44] Armin Haller, Krzysztof Janowicz, Simon Jonathan David Cox, Maxime Lefrancois, Kerry Taylor, Danh Le Phuoc, Joshua Lieberman, Raúl García-Castro, Rob Atkinson, Claus Stadler. The Modular SSN Ontology: A Joint W3C and OGC Standard Specifying the Semantics of Sensors, Observations, Sampling, and Actuation. Semantic Web, vol. 10, no. 1, pp. 9-32, 2019. http://dx.doi.org/10.3233/SW-180320.

Developers: The Google Digital Buildings effort is created and maintained by a team at Google[45] and accepts contributions from an online community of developers.

Maintenance Model: Development of Google Digital Buildings is conducted online via GitHub. The GitHub repository contains all relevant definitions and tools for developing the schema and producing the OWL ontology. No formal releases are published online.

Governance Process: Contributions to Google Digital Buildings are performed via pull requests on GitHub and must be reviewed by the core developer team.

Licensing Model: Google Digital Buildings is licensed under Apache 2.0. Would-be developers must sign a Contributor Licence Agreement which gives the project the ability to use and redistribute any contributions made.

Key Dates: The earliest public work on Google Digital Buildings was published in 2019, though internal development likely started earlier.

Regularity of Updates: Google Digital Buildings is actively maintained by the core developer team and supporting community.

Required tooling, software, dependencies: Google Digital Buildings provides its own toolchain which supports the management and generation of the schema. The schema definitions are defined in YAML, and the data schema is compiled into Protobuf files. These are open standards with many supporting open-source libraries. However, at time of writing, no public platform has been released which supports the ingestion, transmission, analysis and/or manipulation of data expressed in the Google Digital Building schema.

Known/Existing Deployments: Google Digital Buildings is used by Google to manage their portfolio of buildings.

Support for Data-Driven Smart Buildings: Google Digital Buildings defines, to a greater extent than the other metadata models listed in this section, exactly how to present and serialise the telemetry about a building. The data messages are annotated directly with semantic types whose definitions are provided by the schema; this enables the interpretation of data by machines. Although the metadata model and data serialisation are currently used independent of other metadata models described in this document, Google Digital Buildings does maintain cross-compatibility and/or convergence with Brick and Project Haystack as long-term goals.

2.3 Tangentially relevant metadata schemas

The above section gave an overview of 6 key metadata schemas for representing data-driven smart buildings, namely Haystack, Brick, Real Estate Core, BOT and LBD, SAREF, and SSN/SOSA. These will be discussed and compared in more detail in Section 3. Before that, the below section aims to list relevant related standard schemas that are used in important reference domains, namely (1) asset management, (2) occupants and users, and (3) control logic. This section is deliberately kept short and more details and references can be found in Section 4.

[45] https://github.com/google/digitalbuildings/

2.3.1 Asset Management

Several standards and classification schemes have arisen out of the construction industry in order to organise and transmit information about related assets to the various stakeholders and information systems involved in a building's lifecycle.

Uniclass 2015 and Omniclass, the North American version of Uniclass, are equivalent classification schemas derived from ISO 12006-2. It consists of 15 tables that contain codes describing construction objects prevalent in construction documentation such as complexes, entities, activities, spaces, elements, systems, and products; covering all construction sectors and all phases of a project lifecycle. It is supported by the Construction Specification Institute[46] (CSI) and has been included in the National BIM Standard-United States (NBIMS-US)[47] released by the BuildingSMART Alliance.

VBIS[48] is a third-party classification schema, similar to Omniclass/Uniclass but geared towards asset management in buildings during the O&M phase. It defines a single structured tag separated into 4 entries describing Discipline, Product, Sub-type, and Sub-sub-type. It is designed to be easily searchable, and tags can be mapped to Omniclass/Uniclass codes.

COBie (Construction Operations Building Information Exchange), also published in the NBIMS-US standard, is a formal "Model View Definition" which defines a subset of IFC specifically tailored for data-exchange of facility asset information from BIM models in the design phase to FMIS in the O&M phase. COBie information is transmitted in a spreadsheet, containing information on building spaces/zones, and equipment geometries, locations, manufacturing, and performance data. COBie entities can be classified using Omniclass.

OpenMaint is an open source property and facility management application targeted at the O&M phase of the building lifecycle. The application has the ability to import IFC/BIM models to populate a relational database representing the physical assets and metadata associated with a building, site or portfolio. The Postgres database used to support the application can easily be queried and used to automate generation and maintenance of any metadata schema identified in this survey. Such an application also allows for easy integration of automated maintenance requests into workflow.

2.3.2 Occupant/User Perspective

obXML[49] is an XML schema that standardises the representation and exchange of occupant behaviour models for building performance simulation. It builds on the DNAS (drivers-needs-actions-systems) ontology to represent energy-related occupant behaviour in buildings. Drivers comprise environmental and other contextual factors that stimulate occupants to fulfill a physical, physiological, or psychological need. Needs include the physical and nonphysical requirements of occupants that must be met to ensure satisfaction with the environment. Actions are the interactions with systems or activities that occupants can perform to achieve environmental comfort. Systems refer to the equipment or mechanisms in the building that occupants may interact with to restore or maintain environmental comfort. A library of obXML files, representing typical occupant behaviour in buildings, was developed from the literature[50]. These obXML files can be exchanged between different building energy modelling programs, different applications, and different users. A recent extension to the obXML schema[51] enables the representation of occupants' demographic and social-economic attributes and contextual information. The impact and ramifications of storing occupant-centric data

[46] https://www.csiresources.org/standards/omniclass/standards-omniclass-background
[47] https://www.nationalbimstandard.org/about
[48] https://vbis.com.au/classification-and-tags
[49] Hong, T., S. D'Oca, W.J.N. Turner, and S.C. Taylor-Lange. 2015. An ontology to represent energy-related occupant behavior in buildings. Part I: Introduction to the DNAS framework. *Building and Environment* 92:764-777. https://doi.org/10.1016/j.buildenv.2015.02.019.
[50] Belafi, Z.D., T. Hong, and A. Reith. 2016. A library of building occupant behaviour models represented in a standardised schema. *Energy Efficiency* 12, 637–651 (2019). https://doi.org/10.1007/s12053-018-9658-0.
[51] Putra, H.C., T. Hong, C. Andrews. An ontology to represent synthetic building occupant characteristics and behavior. *Automation in Construction* 125, no. 103621, 2021. https://doi.org/10.1016/j.autcon.2021.103621.

on the security and privacy of individuals are currently a topic of active discussion in the wake of regulation like the General Data Protection Regulation (GDPR) and the California Consumer Protection Act (CCPA).

2.3.3 Auditing

buildingSync[52] is a common schema for organising energy audit data so it can be more easily analysed, stored, aggregated and exchanged between different software tools. buildingSync is distributed as an XML schema so it can be easily incorporated into existing workflows and can leverage a variety of existing tooling. buildingSync presents an abstracted view of a building focused on the energy producing and consuming elements, rather than the specific topology and composition of the system and the incorporated data producing elements.

2.3.4 Control and Automation

Control Description Language[53] is a recently developed declarative language for expressing control sequences using graphical programming (blocks) in a vendor-agnostic manner. CDL relies upon the Modelica simulation language[54] to specify control blocks that can be combined to describe low- and supervisory-level control sequences. These can then be used using co-simulation concepts with existing energy modelling tools. The premise is that these blocks can be translated to vendor-specific implementations and deployed in a unified matter to host hardware architectures. The main premise of CDL is to facilitate the delivery of building control systems, addressing a key challenge on capturing the design intent to actual operational practice. CDL includes support for annotations which may in the future contain Haystack tags, Brick models or other pieces of RDF which associate the inputs and outputs of CDL control sequences with the actual data sources and sinks in the building.

2.3.5 Unreleased Metadata Schemas

223P[55], titled “Semantic Data Model for Analytics and Automation Applications in Buildings”, is an RDF-based ontology being actively developed by the Semantic Interoperability Working Group associated with the BACnet subcommittee for the ASHRAE standards organisation. 223P aims to “formally define knowledge concepts and a methodology to apply them to create interoperable, machine-readable semantic models for representing building system information for analytics, automation, and control.” The current draft of the standard proposes a detailed topological model which takes inspiration from the SOSA/SSN and SAREF4SYS ontologies and delivers a finer level of detail in the model than what is provided by most of the schemas described above.

BDNS[56] is a recent W3C effort to define a standard naming scheme for point labels in buildings. The intended scheme will *“align with and complement other initiatives in the industry such as BRICK, Haystack, Omniclass, Uniclass, [and] IFC”* while providing an encoding of semantic information that can be expressed in existing building management systems without incorporating RDF or other non-flat data models.

[52] https://buildingsync.net/
[53] https://obc.lbl.gov/specification/cdl.html
[54] https://modelica.org/
[55] https://www.ashrae.org/about/news/2018/ashrae-s-bacnet-committee-project-haystack-and-brick-schema-collaborating-to-provide-unified-data-semantic-modeling-solution
[56] https://www.w3.org/community/bdns/

3. Qualitative Comparison

After the above survey of existing metadata schemas, it is clear that a diverse number of options are available for representing the building data in an operational building. Each metadata schema has specific features that characterise its purpose and method of use. In this section, we review and compare these features, aiming to provide a guideline to smart building data managers about:

- key features per metadata schema;
- primary intended purpose of use;
- ease of fit into a data-driven smart building

This comparison relies on the following qualitative aspects:

- Structure of the model: Review the benefits/concerns/drawbacks/advantages of data dictionaries, relational models, graph models, naming schemes, drawing from existing literature or experiences where appropriate. How does the choice of structure affect or influence its suitability for data-driven building processes?

- Vocabulary organization and completeness and strictness/rigor: How are concepts organised / defined in the model? Are they generic or specific? Structural vs nominal typing analogy.

- How is alignment handled: What are the ways in which metadata schemas align with one another? Direct lookup, OWL-style equivalency, external software translation, statistical processes, etc. How do the models in the survey align with others?

- What is the role of the model in a hypothetical or reference data-driven building process? How is metadata stored and accessed by software processes to support data-driven buildings? For example, do these processes access a metadata model through a database service, or through another method? How does incorporating a specific metadata schema influence the development, deployment, and management of data-driven building processes?

- Required tooling / software support / expertise: do models need proprietary software? What features are required by supporting platforms? What does this mean for its deployment for a data-driven building? Commercial support? How can above metadata representations be represented in BMS, building platforms?

- Creation / bootstrapping /maintenance: how do models get built and maintained? Automated / semi-automated / manual --- descriptions and examples of each. Who owns the metadata *instance*?

3.1 Model structure

Information models can be broadly categorised by the strictness or flexibility of how they organise information. This is a distinct quality from how the model is stored or accessed by programs; this will be discussed below. In general, stricter information models provide more consistency and validity guarantees to consumers of the model. This feature usually manifests as a lack of expressiveness in the model which can limit general applicability. The choice of an information model for data-driven smart buildings must take the structure of

the information model into account. To illustrate the design space, we examine four common data models: naming conventions, tags, relational/tabular and graph.

Table 1: Overview of metadata schemas and their model structure.

Metadata Schema	Naming Convention	Tags	Relational	Graph	RDF Ontology
Haystack	No	Yes	No	Yes	No
Brick	No	Yes	No	Yes	Yes
RealEstateCore	No	No	No	Yes	Yes
LBD	No	No	No	Yes	Yes
SAREF4BLDG	No	No	No	Yes	Yes
SSN/SOSA	No	No	No	Yes	Yes
Google Digital Buildings	Yes	No	No	Yes	Yes[57]

Naming Conventions are the dominant industry practice for capturing the metadata associated with the I/O points in a BMS/BAS/EMIS. They are prevalent mostly because historical memory constraints on embedded devices offered few opportunities for anything more expressive than a simple text label. Naming conventions are a way of encoding additional structure into such a flat string. Delimiters like +, :, - and / separate a point's name into several components which may indicate its behaviour or purpose, the device or space it is associated with, and what is upstream or downstream of the device that hosts the point. Most naming conventions are fully custom (often site- or vendor-specific), ad-hoc and poorly-enforced; however they are the most commonly available form of building metadata. Although Google Digital Buildings offers a point naming scheme, these are standardized because the labels are modeled with an ontology.

Tags are sets of atomic words and key-value pairs which together indicate the type, purpose and behaviour of an entity. Unlike naming conventions, tags do not have the limitation that they can only be associated with points in the BMS/BAS/EMIS. Importantly, tags can represent abstract concepts like an "HVAC Zone" or entities which are not points like devices or equipment.

Tag-based models, such as Project Haystack, offer an improvement over naming conventions.The set structure facilitates the extraction of the metadata components. Because naming conventions are not usually documented, it is often ambiguous what the "components" of the name are. The tag structure makes its components explicitly discrete. The incorporation of key-value pairs in the tag-based model also permits the modelling of relationships between entities; this is much more difficult to express in a flat string. Tag-based models are also very flexible: it is trivial to create new "tags" and associate them with an entity to describe it.

This flexibility also makes it difficult to consistently communicate semantic information. Without any rules that prescribe meaningful combinations of tags, users of tag models must invent and document their own interpretations (Mathes et al., 2004[58]; Fierro et al., 2019[59]).

[57] The Google Digital Buildings metadata schema defines an OWL ontology export but it is not the intended mode of interaction, and does not support all features of the metadata schema

[58] Adam Mathes. Folksonomies - Cooperative Classification and Communication Through Shared Metadata. Computer Mediated Communication, LIS590CMC (Doctoral Seminar) 2004. http://adammathes.com/academic/computer-mediated-communication/folksonomies.html

[59] Gabe Fierro, Jason Koh, Yuvraj Agarwal, Rajesh K. Gupta, and David E. Culler. 2019. Beyond a House of Sticks: Formalizing Metadata Tags with Brick. In Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities,

Relational or **tabular** models are ubiquitous in many data-oriented businesses and use cases. They provide an intuitive and flexible structure for data that can encode data integrity constraints and invariants in a way that can be enforced and validated automatically by the data management system. This stability contributes to the popularity of tabular models as the basis for many applications.

Tables are one way of capturing semantic metadata. The semantic meaning, properties of and relationships between entities can be encoded in a variety of ways; a “data dictionary” is a reference used to catalogue the meaning and structure of data. One example is the COBie specification for asset management. COBie defines, among other things, a standard spreadsheet (essentially a relational schema) defining what tabs (tables) should exist, what their columns (fields) are named and what the types of the data in those columns should contain.

While tabular data models can capture arbitrary semantic metadata, they are limited in their extensibility because any update to the tabular model requires a redefinition of the schema and the use of migrations to update existing data to that new structure. As a result, tabular metadata is best used when the scope and content of the metadata is well-understood and relatively static. Additionally, while the *structure* of relational schemas is self-describing, the *semantics* of the schema is not. Software must be written with the intended semantics in mind and avoid ambiguities or inconsistencies in the definition of core terms - this often proves to be a challenging task[60] [61].

Graph models are the most expressive method for capturing information among the alternatives discussed above. The expressive power is crucial for capturing semantic metadata in the heterogeneous environments typical of buildings; however, to express that metadata *consistently* requires the use of constraint and logic languages. There are many graph-based data models to choose from — including entity-relationship models, linked property graphs and RDF graphs — but fundamentally they all encode information as a network of nodes with labels and properties, connected by edges with labels and properties. Due to RDF-based ontologies featuring heavily in the set of metadata schemas above, this document will concentrate on the RDF data model.

RDF graphs are directed, labelled, multigraphs. This is a very flexible structure which can express each of the above data models. The fundamental structure in an RDF graph is the triple. A triple is a 3-tuple composed of a subject, predicate, and object. A triple indicates the relationship (predicate) between two nodes (subject and object). Through the use of ontologies, RDF graphs can leverage common definitions, properties, classes and other constructs to communicate meaning in a consistent manner.

RDF graphs are usually stored in specific kinds of graph databases called *triple stores*. These databases usually support storage of many graphs, indexed by some identifier. Access to graphs is handled through the SPARQL query language (see Section 3.5 for more information). Most of these graph databases use triple-specific storage formats and indexing structures, but there are examples of graph databases implemented over established relational databases. Although relational databases have good scaling properties (able to store billions of triples), the differences in query access patterns between the SQL and SPARQL languages means that relational-backed graph databases can suffer performance issues

and Transportation (BuildSys '19). Association for Computing Machinery, New York, NY, USA, 125–134. https://doi.org/10.1145/3360322.3360862

[60] Gabe Fierro, Marco Pritoni, Moustafa Abdelbaky, Paul Raftery, Therese Peffer, Greg Thomson, and David E. Culler. 2018. Mortar: An Open Testbed for Portable Building Analytics. In Proceedings of the 5th Conference on Systems for Built Environments (BuildSys '18). Association for Computing Machinery, New York, NY, USA, 172–181.

[61] Dimitris Mavrokapnidis, Gabe Fierro, Ivan Korolija, and Dimitrios Rovas. 2023. A Programming Model for Fault Detection and Diagnosis. In Proceedings of the 14th ACM International Conference on Future Energy Systems (e-Energy '23). Association for Computing Machinery, New York, NY, USA, 127-131.

Recall that, in general, an ontology is a formal structure of the knowledge in some domain. An RDF ontology encodes domain knowledge as a graph. The semantics of the domain can be expressed in a language — for example, OWL 2 RL, RDFS and SHACL — which consists of well-known terms and computational elements whose execution determines what knowledge is valid and what knowledge can be inferred. A discussion of how these ontology languages work, what they can express, and when to use one over the other is beyond the scope of this survey. Brick, RealEstateCore, LBD, SAREF4BLDG and SSN/SOSA are all examples of RDF ontologies. Project Haystack v4 models can be represented as an RDF graph, but do not contain the formal rules required to constitute an ontology. Google Digital Buildings can be expressed as an RDF ontology, but idiomatic use of the metadata model is not performed through this form.

Several of the following sections will discuss how differences in ontology design impact the utility and ease-of-use of these metadata models and how they fit into a data-driven smart building. Generally, we can recognize a spectrum of models from very flexible approaches (left in Fig. 14) towards more rigid and formally defined approaches (right in Fig. 14).

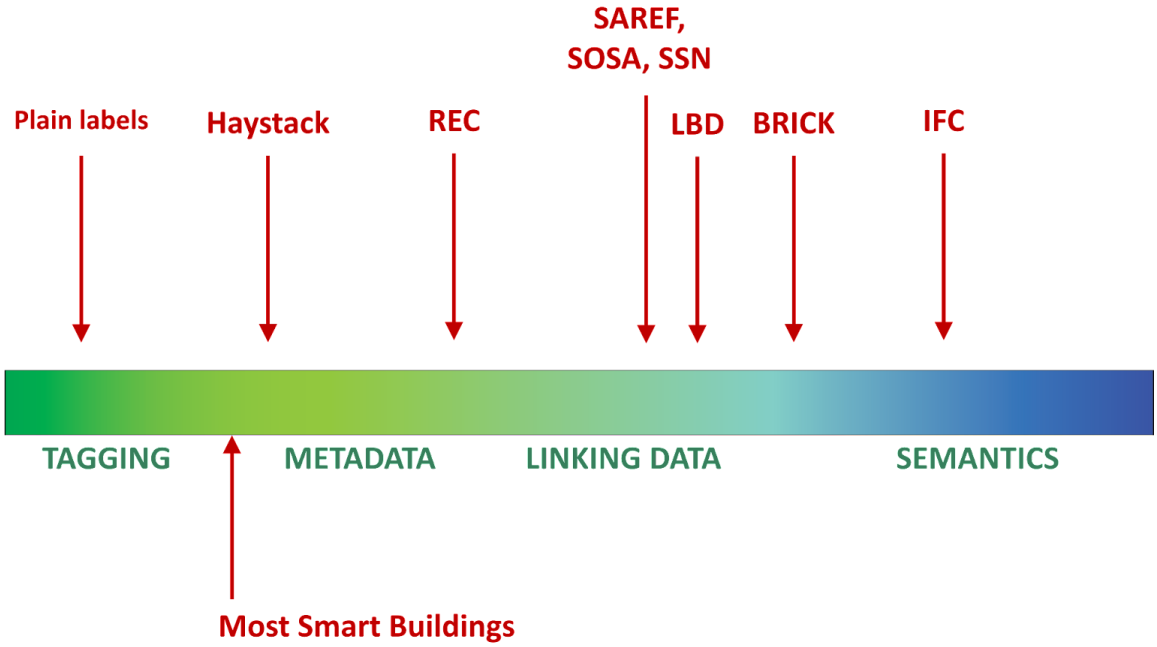


Figure 14: A spectrum of building data representation from more flexible and ad-hoc (leftmost) to more formal and semantically defined (rightmost); plus the estimated location on that scale for several existing metadata schemas (image inspired by Pauwels, 2021[62]).

3.2 Specificity and completeness

The specificity and completeness of a metadata schema is a crucial property for technical domains such as data-driven smart buildings.

Specificity is how precisely and in how much detail the metadata schema can express the type and properties of an entity. Being specific in the definition of an entity aids in the consistent communication of information because more of its properties are captured explicitly in the model. On the other hand, the choice of what *can* be specific within a metadata schema can limit interoperability between models. If certain detail is expressible in one metadata schema but not in another, then that detail is not interoperable.

[62] Pauwels Pieter, Data Integration for Smart Communication. Presentation for the FHI Bits, Bricks, Behaviour conference, Rotterdam, Netherlands. November 2021 https://research.tue.nl/en/activities/data-integration-for-smart-communication

Completeness is the proportion of the concepts in the target domain that are defined by a metadata schema. This is difficult to quantify in the building domain because there is no prescribed family of concepts — in fact, this is the very artifact several of the metadata schemas described above aim to produce!

One way to compare metadata schemas is via the specificity and completeness of the types they attribute to entities. Many metadata schemas define a set of types, classes and/or categories which group entities by their behaviour, purpose, origin or other features. These often range from the generic ("equipment", "device", "location") to the specific ("supply air temperature sensor", "electronically commutated exhaust fan").

BOT, SAREF (including SAREF4BLDG) and SOSA/SSN define *shallow* taxonomies of generic types. These generic types provide a common basis for defining extensive taxonomies which are more specific. For example, BOT defines several generic spatial concepts for buildings — Zone, Building, Storey, Space and Element — but does not define any types that communicate the purpose or design of any instance of those. It is up to a user of BOT to define the specific concepts required for a given application. In a data-driven smart building, these specific concepts may include various types of zones for each building subsystem: HVAC zones, daylighting zones, fire zones, etc. Likewise, Spaces and Buildings may have more specific types that are important to other applications. An Element may indicate *any other non-spatial entity* which is the purview of other building ontologies.

The upshot of shallow taxonomies is that they are straightforward to integrate with other ontologies: most building-oriented ontologies have a notion of a building, a space, a zone and so on. However, these shallow taxonomies do not capture the *specific* information that is actually required to author data-driven building applications. This information must be defined externally to that ontology, raising the possibility of divergent classifications and reducing the potential for interoperability of such information between those extensions. A great advantage lies however on the resulting modularity as well. By relying on several ideally standardised smaller-sized ontologies for specific domains (e.g. infra, hvac, mep, structure, damage), as advocated in the LBD initiative, data management becomes much more scalable and comprehensive, as a monolithic one-size-fits-all ontology like IFC is avoided.

Brick, REC and Project Haystack provide more detailed and specific taxonomies for classifying entities. These types typically match (and indeed are often derived from) the terminology used by practitioners when authoring data-driven applications for buildings. For example, consider how the concept of a sensor measuring the temperature of air flowing into a room may be captured across several ontologies (see Table 2).

Table 2: Overview of metadata schemas and their model structure.

Metadata Schema	**Temperature Sensor Representation**
Google Digital Buildings	`my-sensor:` `  type: supply_air_temperature_sensor`
Project Haystack	`id: my-sensor` `temp:` ✔ `supply:` ✔ `sensor:` ✔ `air:` ✔ `point:` ✔
Brick	`bldg:my-sensor a brick:Supply_Air_Temperature_Sensor .`

RealEstateCore	`bldg:my-sensor a rec:DryBulbTemperatureSensor ;` `rec:hasPlacementContext rec:SupplyAir .`

LDB (incl. BOT)	`bldg:my-sensor a bot:Element,`
	`mep:Sensor-TEMPERATURESENSOR ,`
	`mep:FlowInstrument-THERMOMETER .`

SSN/SOSA	`bldg:my-sensor a ssn:Sensor .`
SAREF4BLDG	`bldg:my-sensor a saref4bldg:Sensor .`

Only Brick, Haystack, Google Digital Buildings and REC capture *all* of the specific qualities of that concept: it is a sensor, it is measuring the dry-bulb temperature quantity of air, and that air is being supplied to some space in the building. While the BOT ontology itself clearly is insufficient and not meant to record all this information, the LBD structure easily allows to define the same detailed representation of the space, its surrounding and contained elements, including the sensor and flow terminal and their classifications / types. It is possible to express *some* of that information in other ontologies — SSN/SOSA permit the description of a temperature sensor by modelling the property being observed by the sensor.

There are several reasons why an adopter of a metadata schema may opt between a shallower or less-specific ontology and a more complex and more-specific ontology. Regardless, when picking a less specific ontology, it is clearly always necessary to supplement it with more specific ontologies that allow description of more specific features.

Usability and Approachability: An ontology may define a shallow class hierarchy, and require the use of other properties and ontologies to fully define the behaviour and purpose of a given entity. Other ontologies may instead define a deeper class hierarchy with many specific types which encapsulate or imply the properties which must be made explicit in another ontology. Classes are often easier to grasp for non-ontologists because they require very little knowledge about how the ontology actually works. Brick and Google Digital Buildings emphasise classes as the primary method for describing entities. RealEstateCore emphasises the use of properties (see Table 2).

Maintainability vs. Extensibility: While a deeper and more specific class hierarchy aids in discoverability — users can traverse the hierarchy to identify what classes are available — it is also more work to maintain. For that reason, such detailed class hierarchies are kept outside of the BOT ontology, also because the AEC industry has a plethora of such class hierarchies (e.g. OmniClass, UniClass, NL/SfB, and many more). This makes these classifications maintainable, and they can easily be plugged and unplugged in the used semantic structure (e.g. MEP ontology in Table 2). Brick and Google Digital Buildings, on the other hand, each define hundreds of classes within their core, where other ontologies only contain a few dozen. Very importantly, if an ontology only defines high-level concepts (such as “Sensor”), then it is unlikely that two users of the ontology will extend those concepts in the same manner. This can impede interoperability. In the experience of the authors, it is easier to ensure interoperability between models by using specialized domain ontologies that are more specific and complete. There is less of a chance that a modeler will need to extend the ontology with their own concepts. For example, extending the high-level BOT ontology with site-specific types will make the model harder to interpret than if the modeler used existing fine-grained LBD ontologies (e.g., DOT, BPO, MEP, and BEO).

3.3 Alignments

As the set of available metadata schemas for data-driven smart buildings grows, it is important for those schemas to specify how they will interact or relate to one another. Users of metadata schemas should be interested in whether a given set of schemas are compatible or interoperable in some way. Different trades, engaged at different stages of the building life cycle, use different metadata schemas to represent the digital information necessary for their roles. There is non-trivial overlap in the concepts modelled by the ontologies and other metadata schemas described above. Any producer or consumer of metadata should understand how the metadata schemas used by a building can "collaborate" to provide a more complete and richer representation of the building. In this section, we describe several philosophies and technical approaches towards *alignment* or *integration* of metadata schemas and provide concrete information on how the metadata schemas above relate to one another (at time of writing).

Table 3: Alignment / transformation table, with the rows representing the source models, the columns the destination models, and in all cells the existing translation methods or alignment.

	Brick	**Haystack**	**Google**	**REC**	**LBD**	**SSN/SOSA**	**Point Labels**	**IFC**
Brick		Inference, Structured	n/a	Alignment	Alignment	n/a	n/a	Structured
Haystack	n/a		Alignment via OAP project	n/a	n/a	n/a	n/a	n/a
Google	n/a	Alignment via OAP project		n/a	n/a	n/a	n/a	n/a
REC	Alignment	n/a	n/a		Alignment	n/a	n/a	n/a
LBD	Alignment	n/a	n/a	Alignment		n/a	n/a	n/a
SSN/SOSA	n/a	n/a	n/a	n/a	n/a		n/a	n/a
Point Labels	Inference	Some community tools	unknown	n/a	n/a	n/a		n/a
IFC	Structured	n/a	n/a	n/a	Alignment, Structured	n/a	n/a	

Foreign key / external reference: A metadata schema can embed a reference to an entity or metadata which is represented in an external store or model. A consumer of the metadata schema must interpret the reference to determine how to access and navigate the external source to find the intended data. The external reference can take many forms:

- a RDF-based metadata schemas can use URIs (Uniform Resource Identifiers -- a generalization of the URLs used to identify web pages) to express the logical location of a resource. Interpreting the URI can inform software what protocol, network location and other parameters are required to access the data stored at that resource

    - Refer to a BACnet object on a network: bacnet://123/analog-input,3/present-value
    - Refer to an XML file on a fileshare: ftp://1.2.3.4/my_devices/vav.xml
- more complex objects (a URI can just be represented as a string) can encode references to arbitrary pieces of information.
    - In Brick, the External Reference object can encode pointers to time series data stored in external databases, entities in IFC models, classes in an asset management system or pub-sub topics containing live data.
    - In BOT, the bot:has3DModel property allows references to an IRI that can take any form (often binary or file-based representation) and holds a representation of the related 3D shape of an object.
- Unstructured references — for example the *rdfs:seeAlso* property in RDF — can also point human consumers to external sources of information; examples of this include a Wikipedia reference (found in many Project Haystack definitions). This is also used to point from the MEP and BEO ontologies to the originating documentation in IFC.

Ontology-based Alignment: One advantage of structuring information with an ontology is the ability to formally define how concepts, properties and relationships in one ontology correspond to another. This correspondence is often called an “alignment”.

The goal of an alignment — from a data-driven smart building perspective — is two-fold. The first goal is to understand entities across a variety of contexts and perspectives and to allow consumers of metadata to make use of the union of metadata from several schemas. The second goal is interoperability between metadata schemas defined with different schemas.

There is a rich body of academic work exploring techniques for (semi-)automatically producing an ontology alignment. Most of these build on the same set of underlying techniques:

- OWL-based class alignment: Pairs of classes from different ontologies can be *symmetrically* aligned by asserting the *owl:equivalentClass* relationship between them. This expresses that every instance of one class is necessarily an instance of its equivalent class. This technique is most appropriate when two ontologies model the same semantic concept. Alignments between Brick and REC are built on this technique.
- RDFS-based class/property alignment: Pairs of classes or properties from different ontologies can be aligned *asymmetrically* by asserting the *rdfs:subClassOf* / *rdfs:subPropertyOf* relationship between them (respectively). This expresses that every instance of the subclass or every use of the subproperty also implies the superclass or superproperty. This does not hold in the other direction: instances of the superclass are not necessarily instances of the subclass, and uses of the superproperty do not imply that the subproperty also holds. This technique is most appropriate when one ontology describes more specific concepts than another. For example, many Brick and REC concepts are subclasses of generic BOT and SSN/SOSA classes.
- SHACL-based class/property alignment: SHACL rules can generate subgraphs in a target ontology from subgraphs in a source ontology. This is a more powerful technique than RDFS- or OWL-based alignment because the execution of these rules can be conditioned on complex and closed-world predicates, and the output of those rules can be arbitrarily complex subgraphs. This enables SHACL to align ontologies that may not be based on the same ontology design principles, or which model different perspectives of the same entities.

Another way to consider ontology alignment is via the lens of *instance-level alignment* versus *schema-level alignment.* An example of instance alignment is given in the below diagrams, where multiple schemas (BOT, DogOnt, SOSA, Schema.org, etc.) are used and combined in an instance model (RDF graph). As the alignments are not fully defined, this instance linking method allows to endlessly add further schemas, provided that they are not ‘semantically conflicting’. Hence this instance-level alignment method mainly

requires making good agreements about data modelling best practices for specific data and specific cases, which is not a light task. This instance-based linking approach is one of the cornerstones behind Linked Building Data; and this approach is considered there because ontology alignments are often lacking and/or disputed, and then considered insufficient.

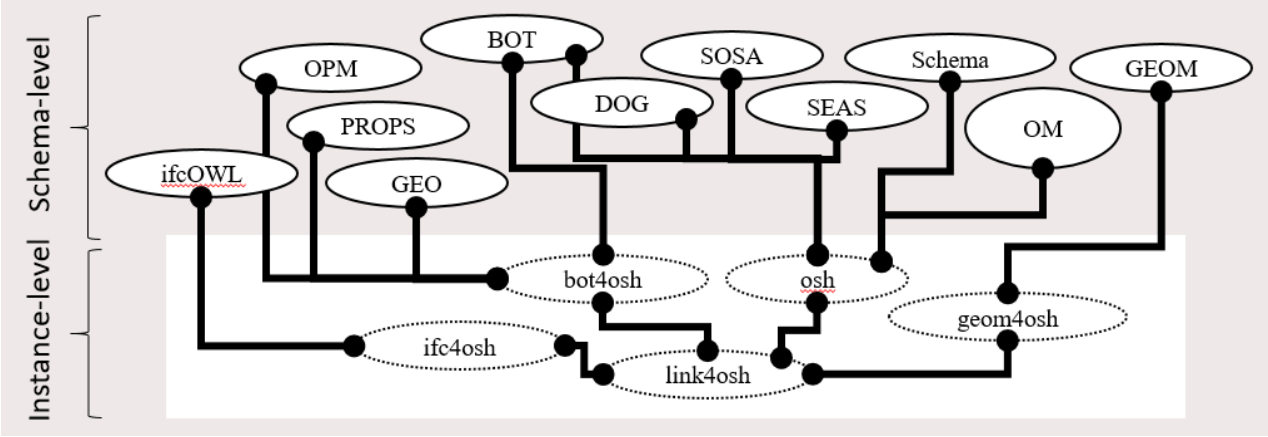


Figure 15: Instance-based linking (image from Schneider et al., 2018[63]).

Structured Transformation: It is not always possible to formally define an alignment between two metadata schemas, especially when at least one of those schemas is not based in an ontology. However, if the metadata schemas contain consistent and structured information (such as in a relational data model), it can be possible to develop a *structured transformation* between the two schemas. These often take the form of rule-based (e.g. RML, R2RML) or procedural frameworks (Java, C#, Python code) with embedded domain knowledge.

Several examples of structured transformations exist in the literature and in practice. IFC models are a common source of formal information about the architecture and construction of buildings which are not easily expressed as RDF graphs. As a result, a family of tools has emerged for consuming the EXPRESS-based representation of IFC models (or COBie spreadsheets) and transforming them to BOT-based graphs and Brick-based graphs, as well as mixes of BOT and Brick. Brick also defines a framework for translating structured VBIS tags to Brick classes.

Some metadata schemas are designed to be standalone; they intend to be a “one-stop shop” for all metadata needs relating to data-driven building applications. Project Haystack and the Google Digital Buildings efforts are two examples of this philosophy. As a result, integration between these metadata schemas and others must depend on some sort of transformation. Google Digital Buildings enumerates a set of well-defined concepts and can make use of structured transformation. Project Haystack neither enumerates nor enforces a set of well-defined concepts, meaning that alignments with Project Haystack must depend on custom transformations, typically implemented in procedural code (see below).

Lastly, it is possible to adopt a structured transformation technique by codifying naming conventions or otherwise imposing regular structure on a metadata model where one does not previously exist.

Custom Transformation: When a source of metadata is unstructured or lacks a formal specification, it is difficult to establish a direct alignment between the source and target schemata. This is because there is no stable representation of concepts that can be mapped into the target schema. In such a case, a custom

[63] Schneider, G. F., Rasmussen, M. H., Bonsma, P., Oraskari, J., & Pauwels, P. (2018). Linked building data for modular building information modelling of a smart home. In J. Karlshøj, & R. Scherer (Eds.), *eWork and eBusiness in Architecture, Engineering and Construction: Proceedings of the 12th European Conference on Product and Process Modelling (ECPPM 2018), September 12-14, 2018, Copenhagen, Denmark* (pp. 407-414). CRC Press. https://doi.org/10.1201/9780429506215-51.

transformation is needed, in which a wide variety of techniques can be applied, most of them very ad-hoc. Some common approaches include:

- heuristic-based transformation: As documented in Fierro et al.[64], Brick uses distance metrics and keywords to predict the most likely set of Haystack tags for a given Brick class, and vice-versa rather than persisting a lookup table which associates sets of tags with a Brick class.
- human-in-the-loop/active learning: At a high level, this technique involves giving sample inputs to human experts which inform the system what metadata can be extracted and how. Over time, the system learns how to extract metadata from particular unstructured sources. This technique is most often used for extracting metadata from unstructured text such as BMS point labels. However, it can be brittle because the learned techniques are difficult to generalise.

In conclusion of this Section, Table 3 shows an overview of the different alignments and transformations that may be made between the mentioned metadata schemas.

3.4 Position in a reference software architecture

The prior sections have established for each metadata representation how it represents buildings, building components, data sources and other salient entities and properties. This section looks beyond the representation of a building to examine how each metadata model interacts with and supports a hypothetical “data-driven smart building” platform.

Due to the immense variability in the capabilities, purpose, implementation and documentation of software platforms for buildings, we will focus on the broad design choices informing how metadata can be used in such platforms. We define three categories: (a) data-oriented metadata models which prescribe a particular data platform API, (b) data-oriented metadata models which are agnostic to the data platform, and (c) metadata models which support other metadata models but do not directly “touch” the data.

Both Project Haystack and Google Digital Buildings are opinionated about the capabilities of the underlying platform. Project Haystack models are most easily queried with the custom Axon language which operates directly on top of the document-oriented tag-based Haystack metadata. Additionally, Project Haystack defines a “curVal” value tag which, when accessed, must return the current state of the corresponding data source. Most deployments of Project Haystack use the commercial backend provided by Skyspark. Google Digital Buildings defines messages for groups of points that correspond to sets of application requirements. While most deployments of either metadata model are built on proprietary platforms, neither of these metadata models actually depends on proprietary software to operate.

Brick and RealEstateCore avoid requiring specific APIs or data formats from the underlying platform, but still go to lengths to provide software access to building data via the metadata model. The two approaches differ primarily in the level of abstraction of the underlying data sources. RealEstateCore models generic data schemas[65] that can represent arbitrary structured data payloads (with a focus on telemetry). This means that a RealEstateCore model can tell software *how* to find the relevant fields in whatever networked protocol the building speaks. Brick stops short of defining protocol-agnostic schemas. Instead, it standardises the sets of parameters required to access external data sources and outsources interpretation of those parameters to existing protocol client libraries. Any entity in a Brick model can point to itself in other digital representations.

[64] Fierro G, Koh J, Nagare S, Zang X, Agarwal Y, Gupta RK and Culler DE (2020) Formalizing Tag-Based Metadata With the Brick Ontology. *Front. Built Environ.* 6:558034. https://www.frontiersin.org/articles/10.3389/fbuil.2020.558034/full
[65] https://doc.realestatecore.io/3.3/dataschemas.html

Metadata models like the LBD family, SSN/SOSA and SAREF4BLDG do not “touch” data in the same way as the above models. In particular, LBD and SAREF4BLDG avoid modelling building telemetry and instead focus on the more semantic features, arguing that telemetry data should ideally not be stored in graph databases for performance requirements. Hence, the system architecture for an LBD-based dataset is assumed to take the form of what is displayed in Fig. 16. Graph-based data, supported by the documented metadata schemas, is stored in graph databases, while telemetric data and file-based data is stored in their own specialised data storage solutions (time series databases, key-value stores, etc.). Integration across data sets is realised using a system integration layer that ties together the different APIs. This system integrator layer also serves as the location for storing the instance-based links between data stores (see previous section and Figure 15), as well as for securing the framework and data stores with independent authentication and security protocols.

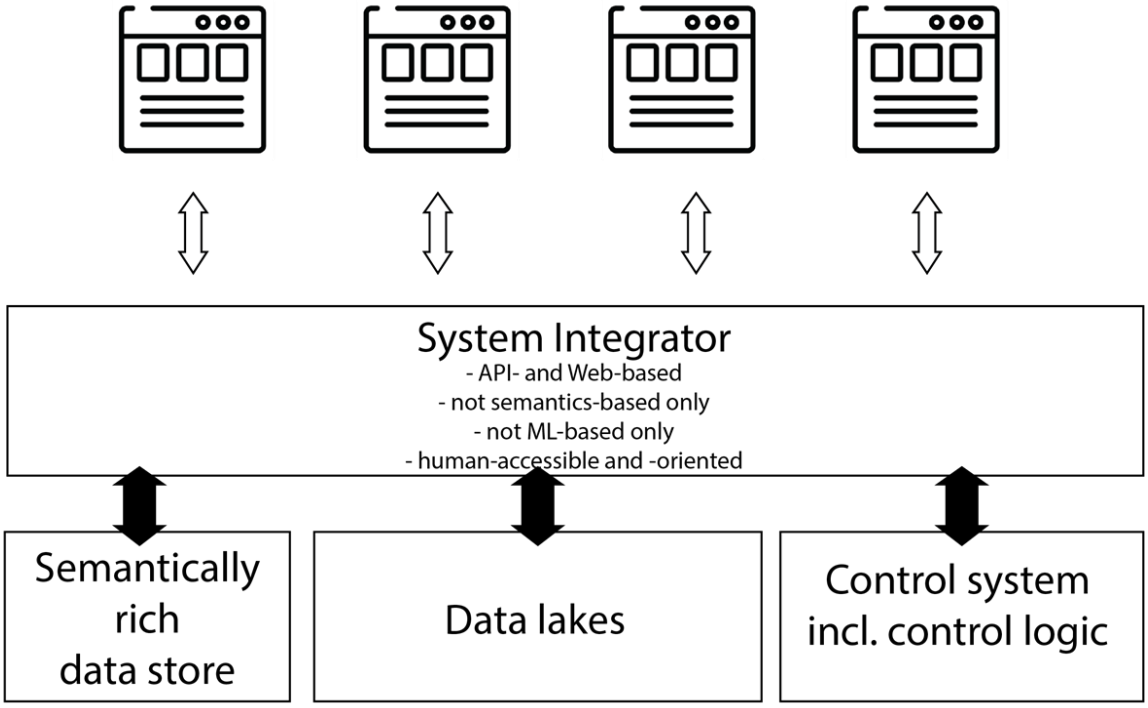


Figure 16: Schematic system architecture diagram with the graph data in the bottom left data store, telemetric data in the bottom middle (data lakes), and control logic separated as well (bottom right). Data is integrated across APIs using a system integrator layer that also regulates access and security (image from Pauwels, 2021[66]).

SSN/SOSA is often used in combination with a LBD graph, and it provides a domain-agnostic model for sensors, actuators and their respective observations and actuations. While it is possible to store sensor data (telemetry) directly into this SSN/SOSA graph, the data is often kept separate, following the architecture specified in Fig. 16.

Note that LBD, SAREF4BLDG, and SSN/SOSA require other software to populate and maintain the data in the graph. Their use thus hinges on the development of such software, or the willingness of existing software providers to build connectors to these ontologies. The LBDServer initiative is one initiative that aims to build such a new interface and tooling.

[66] Pauwels Pieter, Data Integration for Smart Communication. Presentation for the FHI Bits, Bricks, Behaviour conference, Rotterdam, Netherlands. November 2021 https://research.tue.nl/en/activities/data-integration-for-smart-communication

3.5 Required tooling

In order to enable data-driven smart building applications, metadata must ultimately be consumed by software. The structure and form of the metadata schema used informs the features that must be supported by the software platform supporting the application. While the exact software features are specific to each metadata schema, we can identify two major themes.

- **Storage and Access**: Where is the metadata stored? How is the metadata schema accessed? How can applications interact with the stored metadata?
- **Inference and Validation**: Does the metadata schema have the ability to *imply* information, and how is that implied information computed? How is use of the metadata schema validated?

Storage and Access: Data that follows RDF-based metadata schemas — including Brick, RealEstateCore, LBD, SAREF4BLDG and SSN/SOSA — can be stored in any database that stores RDF. The most common varieties of these are "triplestores" and "quadstores" which store triples for a single and multiple RDF graphs, respectively. There are many open source and commercial triplestores and quadstores available which are actively maintained. RDF graphs can be accessed in several different ways: as textual serialisations (e.g. Turtle), web serialisations (e.g. JSON-LD) and through query languages like SPARQL. RDF graphs may also be stored in relational databases or alternative graph databases (e.g., Neo4J) but these may not provide first-class support. In particular, the SPARQL query language for RDF graphs provides many RDF-specific features which can be difficult to emulate in other query languages.

Examples: Allegrograph, Blazegraph, OntoText GraphDB, Virtuoso, Oxigraph, RDFlib, Stardog

It is important to mention here that the actual metadata schema (OWL ontology in the above cases) should be published separately from the RDF store, following best practices for publishing these in a dereferenceable manner[67]. Namely, the ontology should be available in human-readable form (HTML) as well as machine-readable form (RDF) on the location of the ontology URI. RDF ontologies are commonly published with hash URIs using the 3rd 'recipe' in the best practices specification[68]. Many of the mentioned ontologies do not follow these best practices at the time of writing, e.g. Google Digital Buildings, and only make a file available on GitHub.

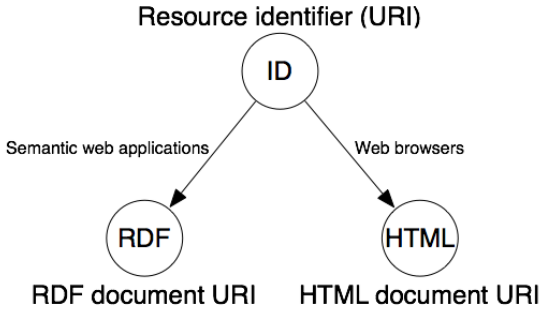


Figure 17: Dereferencing URIs using 303 content negotiation into RDF and HTML URIs (image from W3C[69]).

Although Project Haystack's data model is graph-like, most Haystack models are stored in document databases (MongoDB) where it is simple to represent each entity as a document. Document stores can easily add, remove and query marker tags and value tags on these entities. Project Haystack defines a custom query language named Axon. Several open-source implementations of Project Haystack storage and API servers exist, but the most fully featured implementation is the one maintained by SkySpark.

[67] https://www.w3.org/TR/swbp-vocab-pub/
[68] https://www.w3.org/TR/swbp-vocab-pub/#recipe3
[69] https://www.w3.org/TR/cooluris/

Examples: *SkySpark, Haystack, Shaystack, J2 Innovations*

As stated above, Google Digital Buildings does not have many public details about existing underlying database support. The RDF-encoding of the model means that it can take advantage of the same kind of tooling as the RDF ontologies above; and so it has a similar approach towards data storage and access compared to LBD, Brick, SSN/SOSA, REC, etc. The Protocol Buffer encoding of the data messages can be used by many platforms such as Google's Cloud Pub/Sub offering.

Inference and Validation: An important feature of a metadata schema is to what extent its use can be validated. Validation helps ensure that metadata schemas are used consistently and correctly, which are crucial to broader goals of interoperability and the value of standardising metadata. Related to validation is *inference.* Metadata schemas which support inference are able to imply information which is not explicitly provided by the creator of a building model. Implied information can be computed and then explicitly materialised in the model. This can simplify the development and usage of the model.

Most of the RDF ontologies covered above make some use of existing validation and inference technologies. RealEstateCore, LBD, SSN/SOSA and SAREF4BLDG are all built on the OWL ontology language and incorporate common inference rules around subclasses and the domain and range of properties (Description Logic). The LBD BOT ontology defines some classes as mutually disjoint, which discourages misuse of incompatible classes. Brick and the pending 223P standard both incorporate the newer SHACL standard for specifying constraints on RDF graphs. Many kinds of rules and consistency checks which are difficult or even impossible to express in OWL become straightforward in SHACL. It is nevertheless recommended to keep such SHACL constraints outside of the OWL ontology, as this has a very different purpose and nature (OWL for open-world inference; SHACL for closed-world constraint-checking and validation). As such, external SHACL-based projects have been performed for the other metadata schemas as well (REC, LBD, SSN/SOSA, etc.).

The lack of validation rules in Project Haystack version 3 has been documented in the literature[70] and attempts to provide this feature are part of the nascent version 4 development. However, it remains the case that the lack of a specification for the Haystack data model impedes development of tooling which can ensure that tags are used appropriately. Google Digital Buildings uses the Protobuf definition files and custom open-source tooling to provide validation of usage.

In conclusion, Table 4 gives a brief consolidated overview of the above sections (storage and access, as well as inference and validation).

Table 4: Overview table for the techniques used for storage and access, as well as inference and validation.

	Brick	Haystack	Google	REC	LBD	SSN/SOSA	SAREF4BLDG
Storage and Access	Triplestores, SPARQL	Document/NoSQL database	Protobuf database	Triplestores, SPARQL	Triplestores, SPARQL	Triplestores, SPARQL	Triplestores, SPARQL
Inference and Validation	SHACL	n/a	OWL	OWL	OWL	OWL	OWL

[70] Gabe Fierro, Jason Koh, Yuvraj Agarwal, Rajesh K. Gupta, and David E. Culler. 2019. Beyond a House of Sticks: Formalizing Metadata Tags with Brick. In Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys '19). Association for Computing Machinery, New York, NY, USA, 125–134. https://doi.org/10.1145/3360322.3360862

3.6 Creation and maintenance of data

The point of this last section is to document the ways in which the actual data is meant to be created and maintained. One thing is the metadata schema and its structure, or how it fits into a software architecture, a whole other thing is the creation of data for specific buildings and its maintainability. As most of the metadata schemas covered by this document rely on RDF databases or triple stores, this section will naturally focus on RDF tools as well as ontology-specific methods.

Creation and maintenance of schemas: While new metadata schemas may be created at any point in time, there is a need for stability and standardisation if one aspires these metadata schemas to be used reliably. Hence, the creation of new models is not expected at large scale in the future, especially considering that several good metadata schemas already exist and are being used. Yet, it is of relevance to evaluate how the different metadata schema came into existence.

A good metadata schema is a metadata schema that responds to specific project objectives that can be commercialised and generate value. Many of the documented metadata schemas, however, are built by researchers and data scientists with an eye for the data schema rather than the potential commercial value and implementation specifics. As a result, many of the available schemas do not necessarily have the application support to provide intrinsic commercial value, and instead are standards under development that may be *used for reference* by commercial parties in their in-house implementations (usually proprietary).

As a result, there is a dichotomy between the above outlined metadata schemas and proprietary data models deployed by commercial software. These naturally develop alongside each other, since a commercial company needs to provide its own added value in order to distinguish itself on a market.

Still, many of the metadata schemas documented above have strong support by commercial players for their development. The key value here lies in the creation of an open and neutral format that can be used for *external data exchange*. For example, Project Haystack coupled with AFDD software from Skyfoundry Skyspark is one of the dominant metadata schemas utilised in the real world, to date. Also the Brick metadata schema has strong support by direct commercial members in its Brick consortium. IFC has been built by a consortium of software vendors, and also the LBD ontologies have a community group of AEC-specific software vendors in support of these ontologies.

With significant overlap and few fully formalised standards, it is unclear which metadata schema at this point, if any, will become the standard or de-facto standard for data-driven smart buildings. Moreover, the available metadata standards have significant diversity, and therefore, the chances are very high that there will not be one de-facto standard in the near future.

Creation and maintenance of instance data

In addition to all the specifics about the metadata schema, any implementer of a metadata schema needs to know how to create instance data according to this schema, what method to apply, and what tooling is required to do so. The below paragraphs list some of this information.

Many of the mentioned metadata schemas orient towards the use of RDF and OWL. Therefore, most instance data can simply be created and maintained using standard and generic RDF tooling[71]. This typically includes the following types of tools:

- Ontology editors (e.g. Protégé): allow the creation or maintenance of a new metadata schema as an OWL ontology. Such tools are recommended for editing ontologies only, not the actual instance data.

[71] https://www.w3.org/wiki/SemanticWebTools

- Triple stores (e.g. Stardog, OntoText GraphDB, Virtuoso): allow to store the instance data in what looks like a database management system (DBMS) for graph data. Such tools are recommended for storing actual instance data, not the OWL ontologies that should be published according to best practices[72].
- Dedicated software libraries (e.g. RDFLib, OWLAPI, Jena): allow to handle RDF data programmatically (parse, (de)serialise, query, write, create). These tools are crucial in working with the data in a data-driven smart building, as they enable the required level of automation.

Creation and use of the data has to be intuitive and fast, yet accurate, given the vast quantity of data points within large buildings. Many manual tools exist (e.g. ontology editors), but these are not scalable, and automated tools need to be used instead (dedicated software libraries). Available software libraries for RDF can easily be re-used within existing software environments, as they are available in most major software languages (Java, Javascript, Python, C#, etc). Hence, it is expected that creation of data occurs mostly in newly developed software or in existing software solutions. A particular example in this regard is the creation of connectors from BIM software (e.g. Revit) directly to LBD graphs, REC graphs (MS Azure), Brick graphs, and so forth. Also BMS software is typically quite well compatible with for example BMS graphs (data ingestion and extraction). *Dominant software libraries (e.g. RDFLib, OWLAPI, Jena) as well as good development practices by large software vendors are key here for the creation and maintenance of good and reliable data.*

Further to these generic tools, the various maintainers of the metadata schemas as listed above provide software libraries and applications to facilitate the creation and maintenance of models for each instance. This is in particular the case for Brick, which comes with software solutions of various kinds[73], for example the Brick Viewer in Figure 18.

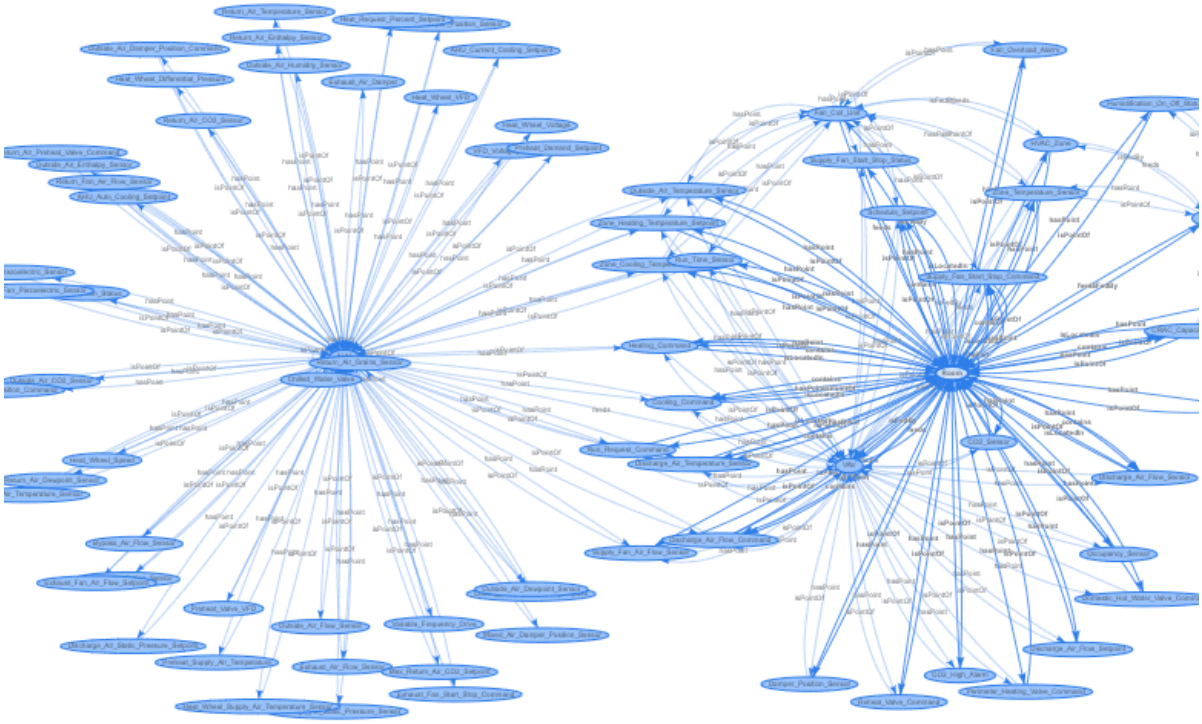


Figure 18: Brick Viewer (image from Brick Consortium[74]).

[72] https://www.w3.org/TR/swbp-vocab-pub/
[73] https://brickschema.org/tools
https://docs.brickschema.org/lifecycle/creation.html
[74] https://brickschema.org/tools/BrickViewer

In the specific case of the AEC domain and its associated BIM data, much focus has been placed on the conversion of IFC and gbXML files to the various operationally focused metadata schemas discussed in this survey. In particular for the LBD ontologies, several converters and transformers are available. The challenge for these conversion tools is to strip excess information and convert only specific objects into the targeted metadata schema, without losing too much meaning.

Efforts have been taken by Pauwels et al. to create an IFC OWL ontology, which is simply an OWL representation for the IFC model[75]. This IFC OWL ontology comes with an IFC-to-RDF converter tool[76], built in Java, that allows transforming any IFC-SPF file into RDF triples. A successor of this work is represented by the LBD ontologies, which aims to be much more modular, extensible, simple, and RDF-oriented (instead of EXPRESS-based). Also in this case, more than one IFC-to-LBD convertor is available, with diverse routines and purposes, including the main LBD CG IFC-to-LBD convertor by Jyrki Oraskari[77], a fork by Pieter Pauwels[78], and a NPM JS package[79]. Similar to the IFC-to-LBD convertors, Python scripts exists that allow to convert IFC to Brick/REC & others[80]. Excellent documentation is available for how to create such models programmatically[81]. Similarly, tools are available to transform gbXML and Revit models to Brick data.

[75] Pauwels, Pieter, and Walter Terkaj. "EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology." *Automation in construction* 63 (2016): 100-133. https://doi.org/10.1016/j.autcon.2015.12.003.
[76] https://github.com/pipauwel/IFCtoRDF
[77] https://github.com/jyrkioraskari/IFCtoLBD
[78] https://github.com/pipauwel/IFCtoLBD
[79] https://www.npmjs.com/package/ifc-lbd
[80] https://github.com/gtfierro/brick-ifc-convert
[81] https://docs.brickschema.org/lifecycle/creation.html

4. Additional schemas and models

While the above sections focus on a number of specific schemas and models, there are several other information representations for smart buildings that have adoption in industry, but do not support data-driven operation and thus do not meet the critera set out in Section 1. In aiming for completeness, this section outlines these schemas and models, using a number of categories that loosely group them.

4.1 Asset Management and AEC

4.1.1 VBIS (Virtual Building Information System)

VBIS is a class-based nomenclature, designed to link operational technology objects and systems together through key/pair tagging. VBIS is designed to provide the level of granularity required in asset classification for operational requirements, with a focus on maintainable assets and providing a syntax for labelling objects in a consistent manner that supports common applications and services. VBIS objects[82] can be mapped to Omniclass and Uniclass objects.

Using a common nomenclature, VBIS postulates and provides a common structure to search asset databases & documentation repositories. A URI standard can be developed using VBIS, allowing for static/dynamic references to drawings or BIM documents.

4.1.2 Industry Foundation Classes (IFC)

Building Information Modelling (BIM) is supported and developed by a range of vendors and is central to products such as Archicad, Autocad, Revit, Navisworks, Aurora & SketchUp, amongst others. Each party in a construction project will use the BIM software best suited to their application or requirements. The IFC schema was developed to provide interoperability between software vendors and therefore the various parties involved in a construction project.

IFC was designed with construction project workflow in mind. For example, how does an architect transmit a design to the hydraulic engineer without inadvertent changes to the architecture occurring without the architect's knowledge? IFC provides assurances that the hydraulic engineer cannot alter the architecture of the building - they can only add a layer of hydraulic services on top of the architectural layers. Changes to architecture to support engineering requirements would require a change request from the consulting engineer to the architect, not the engineer changing the architectural layer themselves, respecting the roles and responsibilities of parties involved in construction.

IFC is a metadata schema acting as a common file interchange format, particularly suited for use on the design and construction phases of a project, with intent to support the operational phase in future, via "6D BIM" (facilities management) and "xD BIM" (performance evaluations). IFC enables file interchange between the vast majority of BIM vendors and products.

Criticism of IFC includes that it is very voluminous and complex, therefore breaking W3C best practices to keep metadata schemas simple for easy maintenance. Other criticism is due to the origin of IFC in the closed-source world, causing duplication or redefinition of already existing W3C standards, or worse, contradicting existing W3C standards on fundamental concepts such as time, location, units of measure. Also, IFC is found

[82] e.g. https://uri.com/?project=1234&type=drawing&label=chiller123&label=chiller345

to be not very web-oriented or web-friendly, as no OWL ontology or JSON schema is included in the ISO standard and a web-incompatible identification mechanism is used (dollar sign not supported in URLs, but commonly present in IFC's shortened GUIDs). As indicated in the previous section, IFC OWL ontology as well as LBD development efforts aimed to respond to these limitations in the IFC metadata schema. To some extent, the added value of IFC is already covered in the form of the LBD alternative, expanded with Brick and Project Haystack graphs and data.

To be complete, it is important to indicate that IFC is useful to the operational phase of a smart building in several ways:

- Import IFC data to an Asset Management System (AMS) or Building Management System (BMS). Examples include Archibus/Sysfm (commercial) or OpenMaint (open source). An AMS is used to raise and track work-orders, using the imported BIM model to reference all assets within the building. A foreign key can be used to reference operational objects for automation of work orders (UUID, VBIS, others). A BMS often houses the space and zone data, as well as a simplified floor plan that can be used to manage the building to various extents (building use and systems).
- Generate an optimised model (data) for the operational phase of the building lifecycle by converting an IFC file to a RDF graph according to any of the mentioned metadata schemas (predominantly Brick and LBD) using the available transformers outlined in the previous section (e.g. brick-ifc-convert[83]). Such transformers filter out 'unnecessary' information as needed. For example, the Brick-IFC converter referenced above currently extracts HVAC equipment, but can cover other operational technologies contained in the IFC metadata schema. The optimised schema (e.g. Brick) is designed to directly support operational activities such as controls inefficiency identification and optimisation, fault detection and diagnostics, benchmarking or other applications and services that require a consistent and dynamic metadata schema.

As can be seen above, the use of IFC in the operational phase of the building remains limited to its use as a data carrier. The transported data is best immediately transformed into more agile data formats and more appropriate metadata schemas. Hence, we have excluded IFC from the main survey in Section 1 and 2, arguing that it is primarily targeted at the AEC industry, rather than the operational and management phase of the building lifecycle.

4.2 Protocols and communication-oriented data schemas

A wide family of communication protocols, each with their own data schemas and formats, exist for networking the cyber-physical components of smart buildings. These include BACnet, LonTalk, KNX, OPC-UA and ModBus. Several of these protocols also define so-called "application profiles" which name groups of I/O points to be used for particular applications. More recent efforts such as Web of Things seek to provide protocol-agnostic representations of the data provided by networked systems, including modern protocols such as MQTT.

[83] https://github.com/gtfierro/brick-ifc-convert

5. Conclusions

This document has presented a structured survey of existing metadata schemas for data-driven smart buildings. By concentrating on schemas which focus on supporting data-driven use cases during the operational stage of a building, this survey aims to provide a practical guide to the range of design decisions, features, supporting tooling and other dimensions of this growing landscape.

We identify seven metadata schemas fulfilling our requirements for inclusion in the survey (see Table 5). While the majority are built on semantic web technologies such as RDF, OWL, and SHACL, others have developed custom data formats and formalizations which require specialised tooling. All metadata schemas are freely available, permissively licensed, and supported by one or more large international companies or standards bodies.

Table 5: Summary: overview of metadata schemas and their model structure.

Metadata Schema	Naming Convention	Tags	Relational	Graph	RDF Ontology
Haystack	No	Yes	No	Yes	No
Brick	No	Yes	No	Yes	Yes
RealEstateCore	No	No	No	Yes	Yes
LBD	No	No	No	Yes	Yes
SAREF4BLDG	No	No	No	Yes	Yes
SSN/SOSA	No	No	No	Yes	Yes
Google Digital Buildings	Yes	No	No	Yes	Yes[84]

The schemas differ primarily in how they support data-driven smart buildings (Fig. 19). Several schemas — Project Haystack, Brick, RealEstateCore and Google Digital Buildings — deal directly with the management and organisation of telemetry about the building. Project Haystack and Google Digital Buildings explicitly define the format of the data and how it is accessed. Brick and RealEstateCore define more generic structures which can be incorporated into a variety of APIs and software platforms. Other schemas — LBD (incl. BOT) and SAREF4BLDG — provide contextual information about the building which can assist applications in finding relevant data, and they typically focus more on asset management rather than telemetric data. These metadata schemas are much closer to the AEC domain and BIM data as a result. Finally, SSN/SOSA provides all needed mechanisms to represent sensor data and actuator data on a large and detailed scale, and leaves the representation of actual building data to other ontologies like Brick, LBD, and SAREF.

[84] The Google Digital Buildings metadata schema defines an OWL ontology export but it is not the intended mode of interaction, and does not support all features of the metadata schema

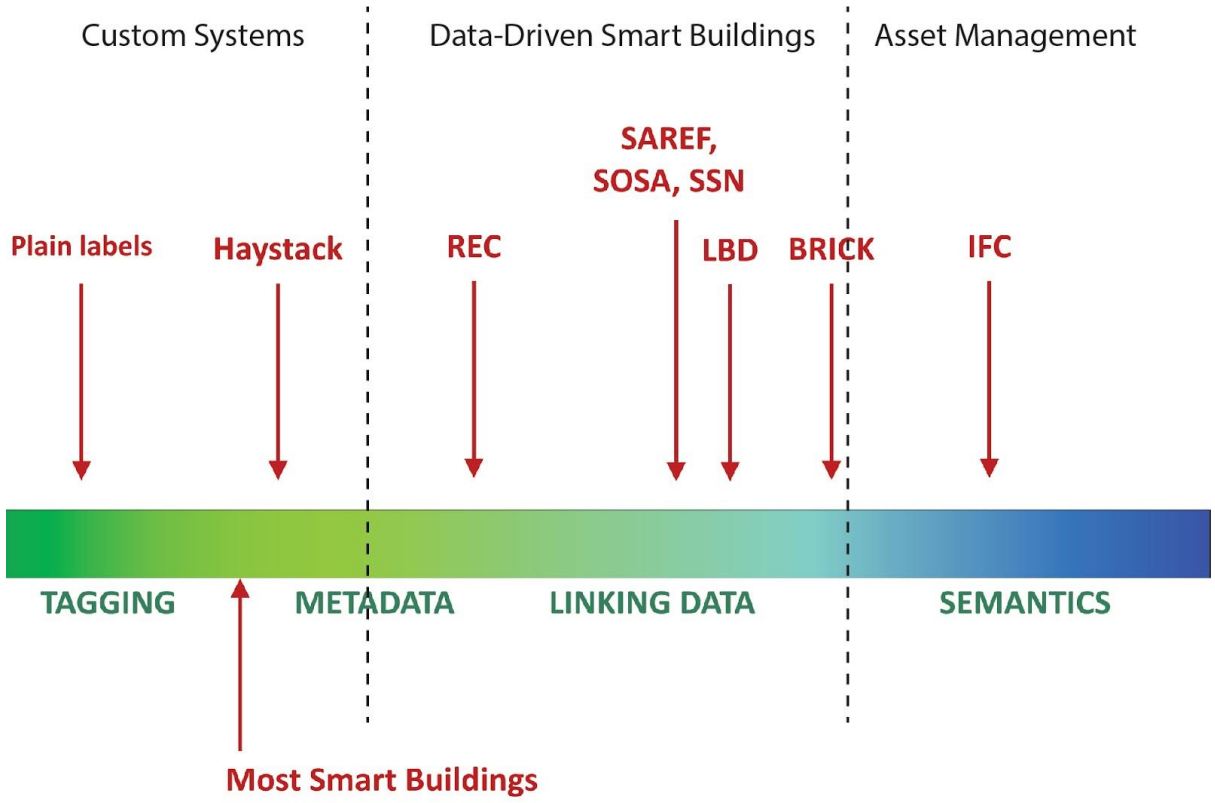


Figure 19: Spectrum of metadata schemas, ordered according to main characteristics (tagging, metadata, linking data, semantics), as well as primary application domains (custom systems, data-driven smart buildings, asset management). (image inspired by Pauwels, 2021[85]).

Among the data-oriented schemas, there is variety in what perspectives of the building are modelled, as well as the consistency and specificity of those perspectives. Brick and Project Haystack model many common building subsystems including HVAC, lighting and electrical. Project Haystack's tagging model affords a great deal of flexibility in describing these systems at the cost of consistency across Haystack models. In contrast, Brick prescribes more of the model structure in exchange for a consistent modelling and querying experience for the consumer of the model. Google Digital Buildings focuses primarily on collections of data coming out of the building, rather than the topology and composition of the building subsystems. RealEstateCore is similar to Brick, but focuses more on the property management aspects and includes a shallower hierarchy of equipment and data source types. Finally, LBD approaches tend to focus much more on asset management and description of the building itself, with much less focus on HVAC systems or their telemetric data logs. Other surveys[86] discuss how the use of these models compares in the context of a specific building.

In our survey, it became also clear that these data models and metadata schemas are and will remain part of a larger software (and hardware) architecture (Fig. 20); and the connections with prevalent commercial software platforms is *absolutely crucial* for the development, usefulness and adoption of these metadata schemas. Commercial software platforms will keep their own data structures, yet they are ideally aligned with the outlined metadata schemas. Furthermore, they need to be combined with other technologies (relational time-series DBs, key-value stores, dedicated algorithms) as well as an integration layer that secures data access, authorisation and security, as shown in Fig. 20.

[85] Pauwels Pieter, Data Integration for Smart Communication. Presentation for the FHI Bits, Bricks, Behaviour conference, Rotterdam, Netherlands. November 2021 https://research.tue.nl/en/activities/data-integration-for-smart-communication
[86] Pritoni, Marco, Drew Paine, Gabriel Fierro, Cory Mosiman, Michael Poplawski, Avijit Saha, Joel Bender, and Jessica Granderson. "Metadata schemas and ontologies for building energy applications: a critical review and use case analysis." *Energies* 14, no. 7 (2021): 2024.

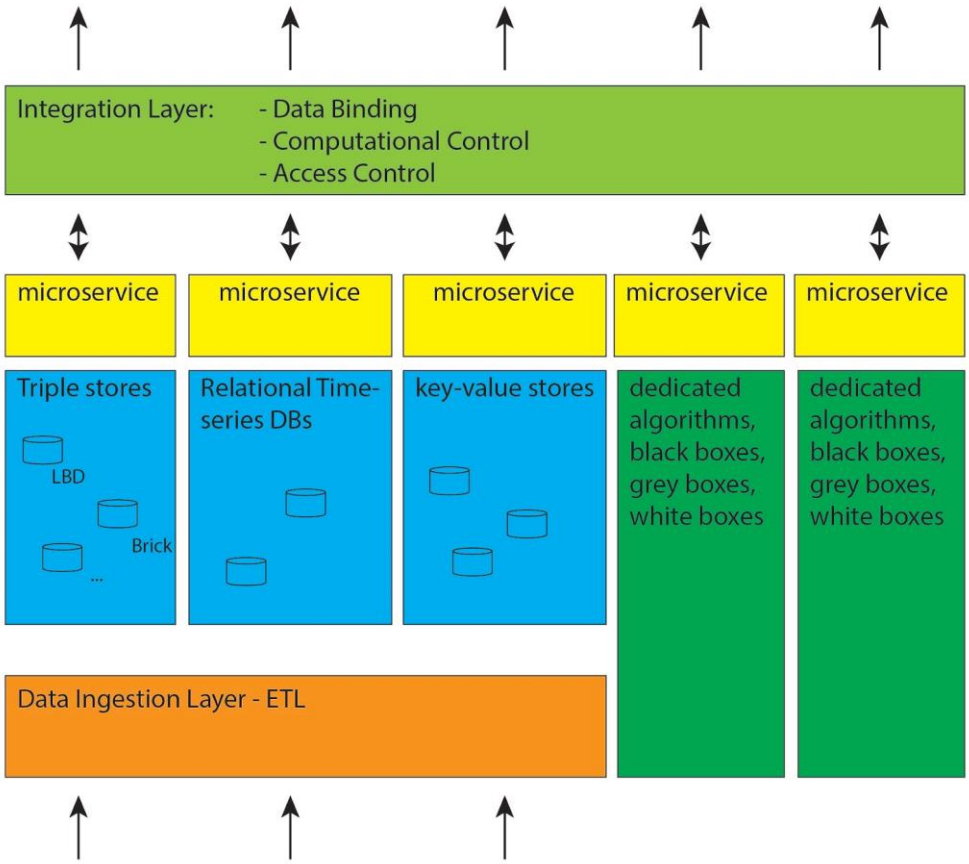


Figure 20: Schematic structure of the commonly expected software architecture for data-driven smart buildings. Data-driven processes interact with the building and its data through the integration layer.

Despite the diversity of approaches and stakeholders for each metadata schema, there is a growing theme of unity and alignment emerging from the various groups. We predict, hope, and recommend that future editions of most metadata schemas will focus more on complementing each other through reductions in scope, rather than expanding the modelling scope to compete on other perspectives of data-driven buildings. We also see RDF-based metadata schemas emerging as the dominant modelling approach. These demonstrate the highest degrees of interoperability and reusability compared to other proprietary models. New tools will emerge that raise the level of abstraction for interacting with RDF-based metadata schemas, ultimately democratising the use of rich metadata in data-driven smart buildings.

ANNEX 81

EBC

Energy in Building and Communities Programme

www.iea-ebc.org