Received 13 March 2026; accepted 29 April 2026. Date of publication 11 May 2026;
date of current version 29 May 2026. The review of this article was arranged by Associate Editor Erman Ayday.

Digital Object Identifier 10.1109/TP.2026.3692127

SmartData Interlock: An Access Control Architecture and Data Exchange Model for Secure Data Ecosystem in Smart Buildings

RAVI SHARMA (Member, IEEE), AVINASH RAMACHANDRUNI (Graduate Student Member, IEEE), EKATERINA PETROVA, AND PIETER PAUWELS

Department of the Built Environment, Eindhoven University of Technology, 5612, AZ Eindhoven, The Netherlands

CORRESPONDING AUTHOR: RAVI SHARMA. (e-mail: r.sharma@tue.nl)

This work was supported by European Union under Horizon Europe Research and Innovation Programme - Project WILSON under Grant Agreement 101147267.

ABSTRACT Smart buildings increasingly rely on heterogeneous data sources, including real-time Internet of Things (IoT) streams, point cloud scans, and Building Information Modelling (BIM), to optimise energy consumption, safety, and operational efficiency. However, the integration of these data sources introduces significant challenges, particularly in terms of ensuring user privacy and data sovereignty across distributed systems. Building operators, tenants, service providers, and emergency responders often require access to overlapping datasets with different legal, organisational, and contextual constraints. Without appropriate governance mechanisms, data sharing risks violating privacy regulations and exposing sensitive information. To address these challenges, this paper first discusses the development of Personalised Data Hubs (PDHs) based on the International Data Spaces (IDS) concept, which integrates multiple data sets into a unified, compliant framework. Building on PDHs, we present the SmartData Interlock (SDI) model, a graph-based, sovereignty-aware access control and data exchange architecture that enables controlled data sharing, allows users to define their own access policies, and ensures regulatory compliance. SDI extends Category-Based Data Access (CBDA) models with purpose binding, delegation semantics, contextual constraint enforcement, and executable policy graphs for lifecycle-aware smart building governance. Using a realistic smart building policy graph, this work evaluates SDI performance and shows that secure data exchange can be achieved with bounded computational overhead, confirming its practical viability in smart building applications and urban planning.

INDEX TERMS Smart buildings, internet of things, data sovereignty, international data spaces, privacy-preserving data sharing.

I. INTRODUCTION

The smart building industries are experiencing a significant digital transformation known as digitalisation, which combines the Internet of Things (IoT), Building Information Modelling (BIM), and advanced analytics to drive innovation and efficiency. However, integrating heterogeneous data streams into smart building ecosystems presents significant challenges, particularly balancing data utility with user privacy and sovereignty [1]. Traditional centralised systems often fail to address the dynamic, distributed nature of IoT data, complicating compliance with evolving general data protection regulation (GDPR) in the EU and EEA, and exposing confidential information to misuse [2]. For example, CO_2 measurements and occupancy traces can reveal indirectly identifiable behavioural patterns, which fall under GDPR's definition of personal data when linked to spatial and temporal metadata.

Smart building systems rely on interconnected devices and applications that generate voluminous, high-velocity data streams. To ensure seamless data exchange among these heterogeneous sources, such as Heating, Ventilation, and Air-Conditioning (HVAC) systems, occupancy sensors,

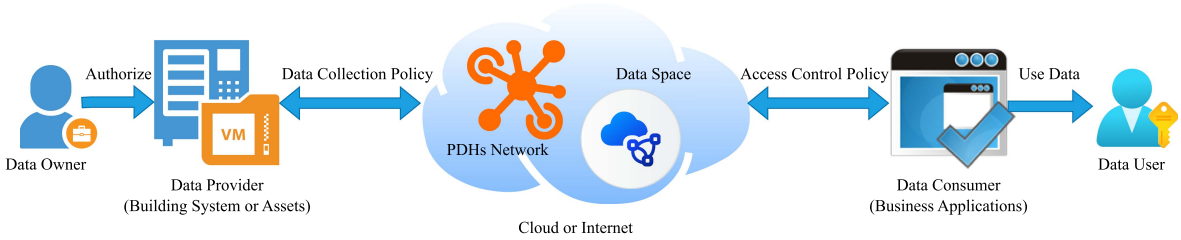

FIGURE 1. **Data dissemination and access control according to the IDS architecture.**

and energy management tools, standardised protocols, such as those based on RESTful APIs and semantic metadata models, must be adopted, which enable interoperability across devices and platforms without assuming globally harmonised international standards [3]. For instance, centralised repositories may inadvertently expose confidential operational data to third-party applications, violating privacy standards [4]. Additionally, the lack of dynamic policy enforcement mechanisms restricts the capability of stakeholders to adapt access rules to changing contexts, such as shifting user roles or regulatory updates [5].

Traditional access-control schemes (e.g., static role-based access control (RBAC) [6] or centralised databases) struggle to address the complexity of modern building environments. Static permission systems do not account for contextual factors such as time-sensitive data sharing or device-specific privacy requirements. Meanwhile, centralised architectures introduce single points of failure and trust, which contradict the decentralised nature of IoT ecosystems [7]. Recent initiatives, like the International Data Spaces (IDS) framework, fill these gaps by emphasising data sovereignty and decentralised governance. The IDS initiative captures these principles in the IDS Reference Architecture Model (IDS-RAM) [8], which establishes standards for trusted data sharing through enforceable usage policies. IDS prioritises accountability and technical enforcement of data-sharing agreements over legal contracts. However, their implementation frequently necessitates significant customization to meet the specific needs of smart building stakeholders, such as facility managers, tenants, and service providers [8]. Structurally, these models separate data collection from policy enforcement, preventing lifecycle-aware inheritance of constraints across raw, processed, and shared data categories.

The IDS framework emphasises data sovereignty, the principle that data originators retain control over its usage and dissemination, through decentralised governance and standardised data exchange protocols (see Fig. 1). While IDS aligns with regulatory mandates like GDPR, its implementation in smart buildings requires customizations to address domain-specific challenges, such as balancing data utility with anonymisation requirements [2]. Edge computing, for example, can reduce risks by processing confidential data locally, but it comes at a cost in terms of real-time analytics and data optimisation [9]. These challenges highlight the significance of architectures that strike a balance between technical feasibility as well as legal and ethical obligations.

Smart buildings involve highly heterogeneous data and multiple dynamic participants. Data flows must be controlled both technically (encryption, filtering) and legally (consent, audit) [10]. We observe that (a) edge processing can reduce privacy risks by anonymizing data on-site, but it necessitates strong local governance, and (b) fine-grained policy models (incorporating roles, device attributes, and context) are required to capture complex sharing rules in construction or project scenarios [11]. Existing work (e.g., RBAC, blockchain-based logging) addresses some aspects of the problem, but a comprehensive framework for smart buildings is lacking [6].

The main contributions of this work are:

1) A sovereignty-aligned PDH architecture integrating edge preprocessing with IDS-compliant governance;
2) The SmartData Interlock (SDI) model extending Category-Based Data Access (CBDA) [12] with roles, purpose binding, and delegation;
3) A graph-executable policy representation enabling constrained authorisation paths;
4) An analytical complexity and scalability evaluation based on a realistic HVAC case study.

The proposed framework bridges critical gaps in smart building ecosystems by combining PDHs' decentralised data governance with SDI's granular access control. It uses state-of-the-art techniques like differential homomorphic encryption to secure data flows while maintaining seamless integration [13]. Finally, this work advances secure, collaborative data ecosystems, allowing stakeholders to leverage IoT-driven insights without compromising privacy or regulatory compliance. The proposed architecture establishes the foundation for scalable, user-centric decision-making in a smart ecosystem, transforming data sharing and utilisation in the built environment.

The remainder of this paper is structured as follows. Section II discusses previous work on access control and privacy in IoT-enabled smart buildings, while Section III describes the architectural foundations and the CBDA-based data governance model. Section IV describes the privacy-preserving PDH architecture, and Section V introduces the SDI model for dynamic, context-aware access control. The SDI policy graph is presented in Section VI, and Section VII

discusses policy analysis techniques. Section VIII concludes the paper by outlining future research directions.

II. LITERATURE REVIEW

The rapid digitisation of smart buildings within the broader building digitisation paradigm has increased the demand for architectures that effectively balance privacy protection with data sharing across heterogeneous systems. Existing Cloud-IoT architectures (see [14] for an overview) typically use layered structures with access control mechanisms such as access control lists (ACL) [15] and attribute-based access control (ABAC) [16]. ABAC, in particular, supports fine-grained, context-aware policies based on user, resource, and contextual attributes (e.g., time, location). Despite its adaptability, ABAC's policy complexity can make it difficult to manage and scale in dynamic environments of connected buildings. Furthermore, these architectures largely overlook data sovereignty, a critical requirement in industries dealing with confidential, distributed data.

Some IoT privacy-preserving frameworks have emerged to address this gap. For example, Databox [17] provides an edge-based gateway architecture with isolated containers for drivers (data sources), data stores, and applications, allowing for localised data control. Similarly, Personal Data Vaults (PDVs) [18] offer a mobile/IoT data store architecture with TraceAudit mechanisms for recording data usage and adaptive filters for limiting outgoing data streams. These systems improve privacy management in general IoT contexts, but they are limited in their ability to build lifecycle workflows.

The Architecture, Engineering, Construction, and Operations (AECO) presents unique challenges, including fragmented stakeholders, transient project phases, and multimodal data sources (e.g., BIM models, point clouds, IoT sensor networks). These complexities necessitate advanced, context-sensitive solutions that go beyond the capabilities of current Cloud-IoT systems. Although Databox emphasises localised data governance through edge hardware and supports service-level negotiation, it lacks built-in edge preprocessing capabilities, which are critical in smart buildings where raw IoT streams (e.g., occupancy sensor data) must be anonymised prior to cloud transmission to meet regulatory requirements [2]. Similarly, while PDVs provide centralised storage, they do not allow for dynamic policy adaptation based on real-time context, such as construction phases or stakeholder roles. PDVs, for example, are not equipped to redact subcontractor financial data from BIM models during public audits, which could expose sensitive business information that should be kept confidential. Importantly, neither framework integrates lifecycle-aware policy inheritance across data categories, which is essential in construction and operations contexts where data sensitivity evolves over time.

Policy management frameworks designed for consumer domains, such as the Personalised Privacy Assistant [18] and OpenPDS [19], have included mechanisms for user-friendly policy design and privacy-preserving data analytics. However, their use in the building domain is limited due to its inherent complexity. OpenPDS' question-and-answer paradigm, for instance, does not support multi-party workflows governed by contractual obligations like non-disclosure agreements with suppliers.

Existing policy specification languages, such as LaSCO [20] and Miro [21], use graph-based models to represent system states and access controls. However, these frameworks are primarily intended for static access control in file systems and databases, with no support for temporal constraints (e.g., revoking access after project completion) and real-time IoT streams. Similarly, rule-based standards such as XACML and RuleML provide insufficient support for dynamic, context-aware, and semantic policy enforcement [22]. XACML defines a standard policy language and decision architecture for attribute-based access control. While expressive, XACML assumes centralised policy decision points and does not natively address decentralised data sovereignty or edge-level data transformation. Integrating XACML in smart buildings typically necessitates extensive custom extensions to handle streaming IoT data and lifecycle-driven policies.

Next Generation Access Control (NGAC) introduces graph-based policy representation and supports flexible relationships between users, objects, and attributes [23]. However, NGAC primarily governs access to stored resources and does not natively integrate data collection pipelines, edge filtering, or sovereignty-preserving usage control across distributed systems. OAuth 2.0 focuses on delegated authorisation for API access [24]. It is well suited for web services but does not express data semantics, privacy constraints, or transformation policies. OAuth 2.0 controls "who can call an API," but not "how data may be collected, processed, filtered, and shared" across heterogeneous building data sources.

To overcome these limitations, we present PDHs, a hybrid framework that combines distributed *Data Pocket* components [2] at the edge with secure cloud repositories. Unlike conventional PDVs, PDHs enable fine-grained policies for both data providers and data consumers. This dual capability is critical for developing lifecycle workflows, where data ownership is frequently distributed among contractors, suppliers, and regulatory bodies.

At the core of our system is the SDI model, which extends CBDA [12] by incorporating attribute-driven, context-adaptive policies specific to the building lifecycle. Fernandez et al. demonstrated that CBDA and its variant A-CBDA [2] can express dynamic, high-level policies for IoT data collection and sharing. For instance, during the early design phase, raw sensor streams may be marked as "restricted," whereas aggregated analytics may be marked as "public." Our SDI framework goes beyond ABAC [16] by integrating Industry Foundation Classes (IFC) and JSON-LD schemas, resulting in lossless data exchange across BIM metadata, IoT streams, and project documentation.

TABLE 1. Comparison of SDI with representative access control frameworks

Feature	XACML	OAuth 2.0	NGAC	SDI
Decentralized	×	×	Partial	✓
Sovereignty	×	×	Partial	✓
Graph-based	×	×	✓	✓
Transformation-aware	×	×	×	✓
Context-aware	Partial	×	✓	✓
Smart building fit	Low	Low	Medium	High

SDI extends beyond these approaches (see Table 1) by governing data collection, categorization, transformation, and service access through a unified policy graph. Unlike XACML and OAuth 2.0, SDI integrates policy enforcement into PDHs, allowing for privacy-preserving preprocessing. Compared to NGAC, SDI explicitly models IoT streams, BIM semantics, and lifecycle workflows in smart buildings. As a result, SDI is particularly useful when decentralising, data sovereignty, context-aware enforcement, and cross-domain interoperability.

III. ARCHITECTURAL FOUNDATIONS AND DATA GOVERNANCE MODELS

This section introduces the fundamental concepts that underpin our access control architecture and data exchange model. We first review the fundamental structure of cloud-IoT architecture and then detail the CBDA [12] framework, which provides the rigorous policy management required for secure data handling in PDHs.

A. CLOUD-IOT ARCHITECTURAL FOUNDATIONS

Modern Cloud-IoT architectures have become the foundation of digital transformation across industries, allowing for seamless data acquisition, processing, and dissemination [10]. These architectures are designed to handle distributed data flows. They are structured into three core layers: the object or device layer, middleware, and application layer [10], [12]. Each layer serves an important purpose, such as managing devices, aggregating data, and providing services. However, adjusting these architectures to the dynamic and diverse workflows of smart buildings requires significant refinement. These architectures are traditionally divided into three layers (see Fig. 2), each serving a distinct function:

1) PERCEPTION LAYER (EDGE-CENTRIC DATA GENERATION)

This foundational layer is responsible for discovering, registering, and managing IoT devices, serving as the primary interface for capturing raw environmental data. Apart from data acquisition, it also performs initial preprocessing at the network edge, filtering or aggregating data to reduce latency and protect confidential data before transmission. This layer integrates diverse data sources, including LiDAR scans, BIM metadata, environmental sensors, and point cloud data. Unlike conventional IoT deployments, smart building

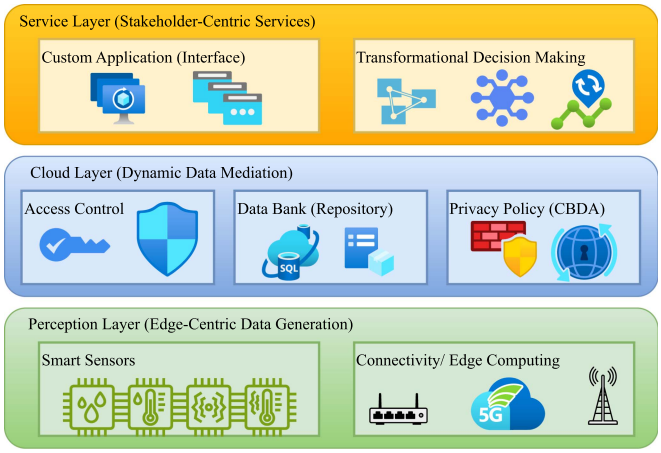

FIGURE 2. Three-layer architecture for data bank.

environments require localised data storage, edge-resident modules that preprocess raw data streams (e.g., masking personally identifiable information (PII) in point cloud scans) before transmission [18]. Traditional architectures often lack such edge-filtering capabilities, posing risks of noncompliance with data regulations when raw data is sent directly to centralised cloud systems [11].

2) CLOUD LAYER (DYNAMIC DATA MEDIATION)

This middleware layer connects edge devices and applications by managing data storage, transformation, and access control. In smart building workflows, this layer must handle transient data lifetimes (e.g., temporary project repositories) and enforce context-aware policies (e.g., revoking subcontractor access post-project). While traditional middleware is based on static databases (e.g., SQL/NoSQL), smart buildings require PDHs, ephemeral repositories that cache data for individual assets (e.g., HVAC or BMS system) and serve as both data producers and consumers. PDHs integrate with the IDS-RAM [8], enabling secure data exchange through RESTful APIs for semantic data exchange.

3) SERVICE LAYER (STAKEHOLDER-CENTRIC SERVICES)

The top layer connects users to the underlying data ecosystem, providing tailored services and analytical insights. It communicates with the middleware through standardised APIs, enabling data-driven decision-making, real-time visualisation, and control mechanisms. Additionally, this layer includes advanced privacy controls, allowing users to define and enforce access policies in accordance with the regulatory frameworks established by the Data Space Architecture. This layer provides specialised decision-support tools to stakeholders. Key services include clash detection in BIM models, predictive maintenance for machinery, and safety monitoring based on smart sensors data. Unlike consumer-oriented platforms (e.g., social media), these applications require role-based semantic data exchange to ensure stakeholders receive information

in industry-specific formats (e.g., IFC schemas for BIM software) [6].

While cloud-IoT architectures provide scalable data management solutions, their centralised nature frequently impedes fine-grained control. Unrestricted data collection can inadvertently lead to privacy flaws, emphasising the need for multiple stakeholders who combine centralised capabilities with dynamic, user-centred enforcement.

B. CATEGORY-BASED DATA GOVERNANCE MODELS

To meet the aforementioned requirements, category-based models like the CBDA framework [12] (an instance of the Category-Based Metamodel [25]) provide granular control over data flows. CBDA's hierarchical categorization of devices, data, and services enables scalable policy enforcement. While the CBDA model provides a structured framework for categorising data, devices, and services, its static category assignments are incompatible with the dynamic, context-sensitive workflows of smart buildings. To address these limitations, we extend CBDA into the SDI model, which introduces attribute-driven, role-based policy enforcement tailored to the evolving needs of smart building ecosystems.

1) ENTITIES AND DATA SOURCES

The model defines a structured set of entities, including IoT devices, raw data streams, and processed data items. Each device (sensor, aggregator) generates unprocessed data, which is then transformed and stored as processed data that can be shared with downstream applications. This clear distinction enables precise control of data at various stages of its lifecycle.

- *Data Sources (Devices):* Consider D as a countable set of devices, where each $\{\mathsf{d}_i\}_{i=1}^{n} \in \mathsf{D}$ includes IoT devices and BMS systems that collect real-time operational data, such as temperature sensors, IP cameras, and smart energy meters. The term *device* refers to various data sources.
- *Data Items (streams):* Let DI represent a countable set of data items d_i. The set is divided into two mutually exclusive subsets:
  - *Unprocessed Data:* Each $\{\mathsf{ud}_i\}_{i=1}^{n} \in \mathsf{UD}$ is raw data (e.g., raw sensors data or LiDAR scans) generated by a device, which often includes contextual metadata like time, location, or environmental parameters.
  - *Stored Data:* $\{\mathsf{sd}_i\}_{i=1}^{n} \in \mathsf{SD}$ denotes processed or transformed data (e.g., anonymised occupancy analytics) that is stored in a repository and made available for subsequent operations.
- *Services:* Consider S to be a countable set of services, with each $\{\mathsf{s}_i\}_{i=1}^{m} \in \mathsf{S}$ representing an application or system component that consumes or processes stored data (e.g., IFC schemas for BIM software or structural integrity analysis).
- *Categories:* Let C be a countable set of categories divided into three subsets:
  - *Unprocessed Data Categories:* $\{\mathsf{udc}_i\}_{i=1}^{p} \in \mathsf{UDC}$ represents the categories that classify raw data items based on sensitivity or contextual attributes (e.g., `restricted` or `open`).
  - *Stored Data Sharing Categories:* $\{\mathsf{dsc}_i\}_{i=1}^{q} \in \mathsf{DSC}$ represents categories of processed data that regulate sharing permissions (e.g., `restricted` vs. `open`).
  - *Service Categories:* $\{\mathsf{sc}_i\}_{i=1}^{r} \in \mathsf{SC}$ represents categories that group services based on their functional or operational domains (e.g., `design_services`, `regulatory_services`).
- *Actions:* Let A be a countable set of actions, which include:
  - *Data Collection Actions:* $\{\mathsf{da}_i\}_{i=1}^{s} \in \mathsf{DA}$ that transform unprocessed data (e.g., encrypt, aggregate) into stored data. If an action da is performed on a raw data item ud to create a stored data item sd, the result is denoted by $(\mathsf{da}, \mathsf{ud}, \mathsf{sd}) \in \mathsf{O_{pd}}$, where $\mathsf{O_{pd}}$ is the set of data collection operations transforming unprocessed data into stored data.
  - *Service Actions:* $\{\mathsf{sa}_i\}_{i=1}^{t} \in \mathsf{SA}$ that operate on stored data items, such as viewing, transferring, or sharing.

2) DATA CATEGORISATION AND POLICY ASSIGNMENT

Data items are organised into categories based on confidentiality and intended exposure. For example, unprocessed data can be labelled based on its inherent confidentiality (e.g., reclassifying BIM metadata from `Restricted` to `Public` post-approval), whereas processed data is grouped under data sharing categories that dictate access privileges. Similarly, services are assigned to categories based on their operational domains, ensuring that data access rights are consistent with both confidentiality and service requirements.

The model defines several fundamental relationships between these sets:

- *Device-Data Assignment:* $\mathsf{DUA} \subseteq \mathsf{D} \times \mathsf{UD}$, where a pair $(\mathsf{d}, \mathsf{ud}) \in \mathsf{DUA}$ indicates that the unprocessed data item ud was generated by the device $\mathsf{d} \in \mathsf{D}$. For the sake of simplicity, we assume that each unprocessed data item is associated with a single device.
- *Data Item-Category Assignment:* The relation $\mathsf{DICA} \subseteq \mathsf{DI} \times \mathsf{C}$ is partitioned as follows:
  - *For Unprocessed Data:* $(\mathsf{ud}, \mathsf{udc}) \in \mathsf{DICAU}$ indicates that the raw data ud belongs to the unprocessed data category $\mathsf{udc} \in \mathsf{UDC}$.
  - *For Stored Data:* $(\mathsf{sd}, \mathsf{dsc}) \in \mathsf{DICAS}$ indicates that the stored data sd falls under the data sharing category $\mathsf{dsc} \in \mathsf{DSC}$.

3) OPERATIONAL ACTIONS AND INTER-ENTITY RELATIONSHIPS

The CBDA framework introduces two main types of operations:

- *Action-Category Assignment:* $\mathsf{ACA} \subseteq \mathsf{A} \times \mathsf{C} \times \mathsf{Cond}$ is divided into:

– *Data Collection Actions:* These operations (e.g., encryption, aggregation, and filtering) transform raw data into a format suitable for storage and controlled sharing. $(\mathsf{da}, \mathsf{udc}, \mathsf{dsc}) \in \mathsf{ACAD}$ indicates that the data collection action da can be applied to unprocessed data in category $\mathsf{udc} \in \mathsf{UDC}$ to create stored data in category $\mathsf{dsc} \in \mathsf{DSC}$.
– *Service Actions:* These are the operations performed on stored data by various services, such as data viewing, transfer, and analytical processing. $(\mathsf{sa}, \mathsf{dsc}, \mathsf{sc}) \in \mathsf{ACAS}$ indicates that the service action sa is applicable to stored data of category dsc when performed by services in the service category $\mathsf{sc} \in \mathsf{SC}$.

The model establishes explicit relationships between these actions and their corresponding data categories, ensuring that only authorised operations are performed.

4) HIERARCHICAL POLICY ENFORCEMENT

The CBDA model uses axioms to enforce hierarchical relationships between data categories [12]. Permissions defined for higher security categories propagate to lower ones under specified conditions, allowing for dynamic and context-aware policy management. This inheritance mechanism is critical for meeting the diverse and changing privacy needs of smart building environments.

- *Service-Category Assignment:* $\mathsf{SCA} \subseteq \mathsf{S} \times \mathsf{C}$, a pair $(\mathsf{s}, \mathsf{sc}) \in \mathsf{SCA}$ indicates that the service s is associated with the service category $\mathsf{sc} \in \mathsf{SC}$.
- *Authorised Operations:* There are two additional relations that govern authorised actions within the system:
  – *Authorised Data Collection:* $\mathsf{ADC} \subseteq \mathsf{A} \times \mathsf{UD} \times \mathsf{SD}$; For any $(\mathsf{da}, \mathsf{ud}, \mathsf{sd}) \in \mathsf{ADC}$, the data collection action da is authorised to produce stored data sd from unprocessed data ud.
  – *Authorised Data Access:* $\mathsf{ADS} \subseteq \mathsf{A} \times \mathsf{SD} \times \mathsf{S}$; Any $(\mathsf{sa}, \mathsf{sd}, \mathsf{s}) \in \mathsf{ADS}$ authorises the service action sa on stored data sd for the service s.

C. AXIOMATIC ENFORCEMENT

The CBDA model uses axioms to ensure dynamic, hierarchical policy inheritance. Specifically, two axioms—referred to as (da1) for data collection and (da2) for data access—ensure that permissions granted at higher levels are transferred to lower levels. Formally, the subset relation $\mathsf{c} \subseteq \mathsf{c}'$ represents the hierarchy among categories, with lower categories implying less restrictive access (c is less protected than c', as demonstrated by the conditions $\mathsf{udcsubsetequdc}'$, $\mathsf{sc} \subseteq \mathsf{sc}'$, $\mathsf{dsc} \subseteq \mathsf{dsc}'$ in axioms da1 and da2):

(*da*1) *Data Collection : unprocessed data → stored data*

$$\forall \mathsf{ud} \in \mathsf{UD}, \forall \mathsf{sd} \in \mathsf{SD}, \forall \mathsf{da} \in \mathsf{DA},$$
$$(\exists \mathsf{d} \in \mathsf{D}, \exists \mathsf{udc}, \mathsf{udc}', \mathsf{dsc}, \mathsf{dsc}' \in \mathsf{C},$$
$$(\mathsf{d}, \mathsf{ud}) \in \mathsf{DUA} \wedge (\mathsf{ud}, \mathsf{udc}) \in \mathsf{DICAU} \wedge$$
$$\mathsf{udc} \subseteq \mathsf{udc}' \wedge (\mathsf{da}, \mathsf{udc}', \mathsf{dsc}') \in \mathsf{ACAD} \wedge$$
$$(\mathsf{da}, \mathsf{ud}, \mathsf{sd}) \in \mathsf{O}_{\mathsf{pd}} \wedge \mathsf{dsc} \subseteq \mathsf{dsc}' \wedge$$
$$(\mathsf{sd}, \mathsf{dsc}) \in \mathsf{DICAS}) \Leftrightarrow (\mathsf{da}, \mathsf{ud}, \mathsf{sd}) \in \mathsf{ADC})$$

(*da*2) *Data Access : stored data → services*

$$\forall \mathsf{sd} \in \mathsf{SD}, \forall \mathsf{sa} \in \mathsf{SA}, \forall \mathsf{s} \in \mathsf{S},$$
$$(\exists \mathsf{dsc}, \mathsf{dsc}', \mathsf{sc}, \mathsf{sc}' \in \mathsf{C}, (\mathsf{sd}, \mathsf{dsc}) \in \mathsf{DICAS} \wedge$$
$$\mathsf{dsc} \subseteq \mathsf{dsc}' \wedge (\mathsf{s}, \mathsf{sc}) \in \mathsf{SCA} \wedge \mathsf{sc} \subseteq \mathsf{sc}' \wedge$$
$$(\mathsf{sa}, \mathsf{dsc}', \mathsf{sc}') \in \mathsf{ACAS}) \Leftrightarrow (\mathsf{sa}, \mathsf{sd}, \mathsf{s}) \in \mathsf{ADS})$$

Definition 1: (CBDA Policy) A CBDA policy is defined by the tuple $\langle \mathtt{E}, \mathtt{Rel} \rangle$, where:

- $\mathtt{E}$ refers to the set of entities (devices, data items, services, categories, and actions).
- $\mathtt{Rel}$ refers to all defined assignment and authorisation relations.

The policy is valid if all the axioms are met.

While the CBDA model offers a structured and scalable foundation for categorising data, devices, and services, its static policy enforcement limits its applicability in dynamic smart building environments. Building upon this foundation, our SDI model, as detailed in Section V, extends CBDA with attribute-driven, context-aware access control mechanisms, enabling secure, fine-grained management of the entire data lifecycle, from device-level collection to cloud-based sharing.

IV. PRIVACY-PRESERVING ARCHITECTURE FOR PERSONALISED DATA HUBS IN SMART BUILDINGS

This section presents our (novel) architecture for Personalised Data Hubs (PDHs), which are designed to empower data owners with fine-grained control over data flows from IoT devices to cloud services while ensuring robust privacy preservation and semantic integration, as described in section III-A. Our framework integrates dynamic policy enforcement with advanced cryptographic techniques to accommodate the transient, decentralised nature of data in smart buildings. As illustrated in Fig. 3, the system architecture integrates PDH and SDI to enable seamless, semantically aligned, and policy-compliant data exchange between data providers and consumers. Conceptually, the PDH acts as a sovereignty boundary that mediates all cross-organisational data exchanges. No unauthorised raw data leaves this boundary. All outbound data flows are transformed, filtered, or encrypted according to SDI-enforced policies before external dissemination.

A. DATA SPACE-COMPLIANT PDH ARCHITECTURE SPECIFICATION

The proposed PDH framework achieves secure data exchange within data spaces by bringing together technical, semantic, and organisational alignment. We design and analyse a three-layer architecture (see Fig. 4), extending previous work

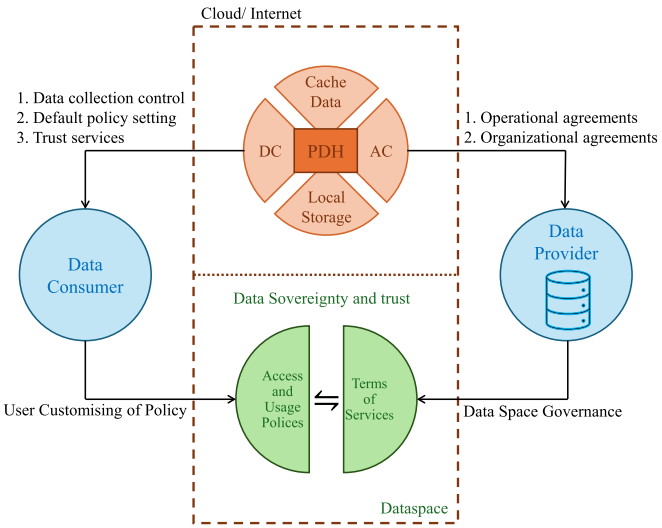

FIGURE 3. Deployment architecture of PDH and SDI for secure smart building ecosystems.

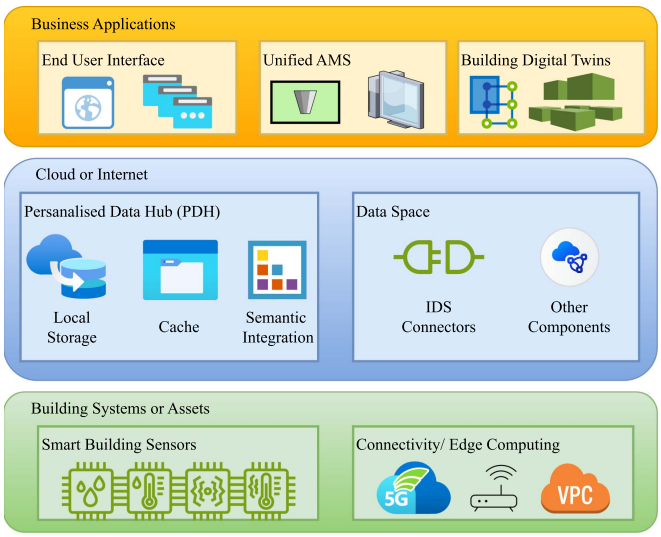

FIGURE 4. Three-layer PDH architecture.

from [26], with key improvements inspired by the model in [2]. Within this architecture both the PDH and IDS concepts are deployable over cloud or internet-based infrastructures. An integrated encryption technique enforces access control, ensuring that data is only accessible based on predefined user roles and permissions.

1) BUSINESS APPLICATION LAYER

This layer provides SDI role-specific interfaces for both data owners and third-party service consumers, and it manages data-sharing workflows using dedicated APIs. This layer translates user-defined policies into executable rules and routes service requests to the middleware layer for enforcement, while PDHs align with the IDS-RAM to ensure three dimensions of integration: technical integration through standardised APIs and data format normalisation (e.g., Brick Schema for building metadata [27]); semantic compatibility through JSON-LD contextualisation and ontology alignment (e.g., IFC schemas for construction); and organisational integration by dynamically mapping role-based data exchange policies for stakeholder roles to ensure regulatory compliance [28].

2) CLOUD OR INTERNET LAYER

This layer is at the core of the architecture, containing critical components that enforce data access control and secure data storage. This layer serves as the PDH network in the IDS ecosystem, allowing for secure data exchange through standardised protocols and cryptographic safeguards. Importantly, the PDH enforces a strict sovereignty boundary: no raw or unprocessed data is transmitted outside the PDH unless explicitly authorised and transformed according to SDI policies. All outbound data flows are subject to filtering, anonymization, encryption, and category validation prior to external dissemination. Key Modules in the Cloud Layer:

- *Access Control:* Validates service requests against a unified SDI policy, ensuring that data sharing adheres to user-defined privacy constraints. Uses dynamic attribute provisioning service (DAPS) to validate participant credentials (e.g., eIDAS-compliant identities).
- *Data Repository:* A multi-modal database (NoSQL/SQL) that supports both structured (BIM metadata) and unstructured (sensor logs) data. The repository supports integration with external cloud storage while implementing encryption and filtering mechanisms in accordance with the data collection policy.
- *Privacy-Utility Mechanism:* Dynamically balances user-defined privacy settings with the potential benefits of services, utilising negotiation models to optimise data sharing while maintaining confidentiality.
- *Auditing:* Maintains detailed logs of all data transactions, access attempts, and policy enforcement events to aid regulatory compliance and post-hoc analyses. Temporal constraints can be implemented using blockchain-based smart contracts (e.g., Hyperledger Fabric), enabling automated permission revocation and auditability [29].

3) BUILDING SYSTEM OR ASSETS LAYER

The foundational layer consists of the actual IoT devices installed on smart building sites. This layer consists of smart sensors and other devices that collect diverse data streams (e.g., environmental sensors). Communication between these physical devices and the PDH is facilitated by standardised protocols, which ensure that only pre-processed data is transmitted upstream [9].

B. SECURE PDHS NETWORK

Our secure data exchange implementation creates a comprehensive framework for data management across smart building ecosystems. By combining IDS-compliant architectures, advanced cryptographic techniques, and dynamic,

context-aware policy enforcement, the proposed PDH system ensures secure and seamless data sharing that is both scalable and regulatory compliant. Our PDH architecture is distinguished by the flexibility with which secure data repositories can be implemented. To accommodate varying operational needs, our design supports both on-premise and cloud-based deployments.

- *Local Deployment:* Organisations with dedicated technical resources can deploy the data locally, giving them complete control over confidential information. Integration with diverse external data sources is facilitated by IDS connectors [30], which translate local policies into standardised usage control expressions.
- *Cloud-Based Deployment:* In scenarios where local infrastructure is not feasible, both static and dynamic data repositories can reside in the cloud.

C. ARCHITECTURE VARIANTS FOR SCALABILITY

Our PDH framework supports a variety of deployment models to ensure adaptability across diverse smart building projects.

- *On-Premise PDHs:* This model is ideal for highly secure projects because it ensures that both the edge and core layers are hosted locally, keeping the data isolated from external networks.
- *Hybrid Cloud PDHs:* Edge caching for transient data is combined with centralised cloud analytics, which uses IDS-compliant platforms for long-term storage and processing.

V. SMARTDATA INTERLOCK MODEL

This section introduces the SmartData Interlock (SDI) model, our proposed framework for secure and scalable data governance in smart buildings. SDI expands on the A-CBDA model [2], which was influenced by ABAC [16] and evolved from the CBDA model [12]. SDI builds upon these foundations by providing dynamic, context-aware access control tailored to the needs of smart building ecosystems. Unlike traditional ABAC and CBDA, SDI interlocks data providers and consumers using graph-based, executable policies that govern data collection, transformation, and service access throughout the building lifecycle. The SDI model integrates role hierarchies, attribute-driven conditions, and semantic data exchange to enforce fine-grained, context-aware policies across diverse stakeholders while ensuring compliance with IDS architecture.

A. ENTITIES FOR SDI

The SDI model expands on the generic entity set $\mathtt{E}$ (defined in Section III-C) and distinguishes its elements. $\{\mathsf{r}_i\}_{i=1}^{r} \in \mathsf{R}$ defines predefined roles for stakeholders, including `FacilityManager`, `HVAC_Technician`, `SecurityOperator`, and `TenantAdmin`, which were selected because they represent core stakeholder groups in smart building ecosystems. Each has distinct responsibilities and contextual needs for accessing building data. DI are divided into two categories: $\{\mathsf{od}_i\}_{i=1}^{n} \in \mathsf{OD}$ *operational data*, which are transient, real-time streams like occupancy levels and energy consumption, and $\{\mathsf{ad}_i\}_{i=1}^{n} \in \mathsf{AD}$ *archival data*, which are long-term records such as maintenance logs and BIM data. OD corresponds to UD defined in Section III but is distinguished here to emphasise real-time characteristics. Set S refers to data-consuming applications such as energy analytics platforms, emergency response systems, and building automation controls. The precise set At of *attributes* are contextual properties describing entities, such as `BuildingZone`, `SecurityClearanceLevel`, `EmergencyStatus`, and others. These attributes are the minimum set required to capture the spatial, hierarchical, and temporal dimensions of access control in smart building environments. They are modelled as functions mapping entities to values in their domain. Set C is defined by Boolean conditions over attributes, such as a `Restricted` category determined by a `SecurityClearanceLevel` less than a specified threshold. For each entity $e \in \mathtt{E}$, an attribute tuple $\ell_e \in (\mathsf{At} \times \mathsf{V})^*$ is maintained to ensure that dynamic changes are immediately reflected in policy evaluations. The notation Z^* refers to a sequence of zero or more elements in Z.

B. HIERARCHICAL SDI RELATIONSHIPS

To enforce the principle of least privilege and ensure dynamic access control, the SDI model adds role-based assignments and hierarchical relationships to the A-CBDA category framework [2].

- *Device-Attribute Assignment:* $\mathsf{DAtA} \subseteq \mathsf{D} \times (\mathsf{At} \times \mathsf{V})^*$, a tuple $(\mathsf{d}, (a_1, v_1), \ldots, (a_n, v_n)) \in \mathsf{DAtA}$ indicates that device d has attributes $a_1, \ldots, a_n$ with corresponding values $v_1, \ldots, v_n$.
- *Data-Item-Attribute Assignment:* $\mathsf{DIAtA} \subseteq \mathsf{DI} \times (\mathsf{At} \times \mathsf{V})^*$ relation is divided into assignments for $\mathsf{DIAtAOD}$ (operational data) and $\mathsf{DIAtAAD}$ (archival data).
- *Service-Attribute Assignment:* $\mathsf{SAtA} \subseteq \mathsf{S} \times (\mathsf{At} \times \mathsf{V})^*$ Each tuple in SAtA identifies a service and its characteristic attributes.
- *Role-Attribute Assignment:* This relation, $\mathsf{RAA} \subseteq \mathsf{R} \times (\mathsf{At} \times \mathsf{V})^*$, associates each role with its defining attributes. For instance, an HVAC technician may have an entry: (`HVAC_Technician`, {(`Zone`, "EastWing"), (`Certification`, "REHVA")}). Only roles whose attribute tuples satisfy the Boolean conditions defining a category are eligible to inherit its associated permissions.
- *Role-Category Assignment:* In $\mathsf{RCA} \subseteq \mathsf{R} \times \mathsf{C}$, a tuple $(\mathsf{r}, \mathsf{c}) \in \mathsf{RCA}$ indicates that role r has access to data categorised as c. For instance, (`SecurityOperator`, `SurveillanceData`) $\in \mathsf{RCA}$ allows security personnel to access surveillance feeds.
- *Category-Attribute Assignment:* In $\mathsf{CAtA} \subseteq \mathsf{C} \times \mathsf{Cond}$, a pair $(\mathsf{c}, k) \in \mathsf{CAtA}$ indicates that category c is defined by the Boolean condition k, which is evaluated over entity attributes.

A reflexive-transitive relation $\subseteq$ is defined between categories to capture hierarchical access rights. For categories c, $\mathsf{c}' \in \mathsf{C}$, $\mathsf{c} \subseteq \mathsf{c}'$ means that the Boolean condition defining c

implies that of c'.

$$(ac1)\quad \textit{Access Control}$$
$$\forall \mathsf{r} \in \mathsf{R},\ \forall \mathsf{c} \in \mathsf{C},\quad (\mathsf{r}, \mathsf{c}) \in \mathsf{RCA} \iff \exists k;$$
$$\text{s.t. } (\mathrm{c}, \mathrm{k}) \in \mathsf{CAtA},\ \ell_{\mathsf{r}} \vdash \mathrm{k},$$

where ℓ_{r} denotes the attribute tuple associated with role r.

The dynamic nature of the SDI model is realised through a set of axioms that link attribute values to category membership. These axioms ensure that an entity is assigned to a category only if its attributes satisfy the defining condition of that category.

$$(ac4)\quad \textit{Category Inheritance}$$
$$\forall \mathsf{c}, \mathsf{c}' \in \mathsf{C}, \forall \ell \in (\mathsf{At} \times \mathsf{V})^*,$$
$$\mathsf{c} \subseteq \mathsf{c}' \iff \textbf{if } \ell \vdash k_{\mathsf{c}} \textbf{ then } \ell \vdash k_{\mathsf{c}'}$$

The categories c and c' are defined by the conditions k_{c} and $k_{\mathsf{c}'}$, respectively. This axiom ensures that the hierarchical relationship between categories is preserved based on their defining conditions.

C. DYNAMIC POLICY SPECIFICATION

SDI policies dynamically adjust to real-time conditions in smart buildings. This is accomplished using a decentralised policy engine that constantly monitors both entity attributes and environmental context. For instance, in emergency situations (e.g., a fire alarm in `ZoneB`), the policy engine can automatically update role-category assignments:

$$\texttt{EmergencyStatus} = \text{Active} \Rightarrow$$
$$\mathsf{RCA} \leftarrow \mathsf{RCA}$$
$$\cup \{\texttt{SecurityOperator}, \texttt{EmergencyExitData}\},$$

during emergency situations (e.g., a fire alarm in `ZoneB`), the policy engine can dynamically update role-category assignments, such as elevating a `SecurityOperator`'s access to critical data like emergency exit routes, while temporarily expanding an HVAC technician's permissions to include manual override capabilities for smoke extraction systems. Temporal constraints are enforced using smart contracts, which automatically update or revoke permissions as project phases evolve.

Definition 2 (SDI Data and Service Axioms):

$$(ac2)\quad \textit{Data Item Categorisation}$$
$$\Leftrightarrow \forall \mathsf{d} \in \mathsf{DI}, \forall \mathsf{c} \in \mathsf{C}, (\mathsf{d}, \mathsf{c}) \in \mathsf{DICA}$$
$$\exists \ell_{\mathsf{d}}\ ((\mathsf{d}, \ell_{\mathsf{d}}) \in \mathsf{DIAtA} \wedge$$
$$\exists k\ ((\mathsf{c}, k) \in \mathsf{CAtA} \wedge \ell_{\mathsf{d}} \vdash k))$$

$$(ac3)\quad \textit{Service Authorisation}$$
$$\forall \mathsf{s} \in \mathsf{S}, \forall \mathsf{c}' \in \mathsf{C}, (\mathsf{s}, \mathsf{c}') \in \mathsf{SCA} \Leftrightarrow \exists \ell_{\mathsf{s}};$$
$$((\mathsf{s}, \ell_{\mathsf{s}}) \in \mathsf{SAtA} \wedge \exists k'; ((\mathsf{c}', k') \in \mathsf{CAtA} \wedge \ell_{\mathsf{s}} \vdash k'))$$

D. CONTROL MECHANISMS IN SDI

SDI also introduces some semantic extensions, such as obligations, contextual constraints, delegation, and purpose binding. These additions allow for more precise, condition-aware, and purpose-specific access control decisions.

Let P be the set of principals (e.g., users, services), and O be the set of obligations (conditions that must be met before or after access). We define the obligation assignment relation $\mathsf{OB} \subseteq \mathsf{P} \times \mathsf{A} \times \mathsf{D} \times \mathsf{O}$. This relation states that when p performs a on d, the obligation o must be met. The permission assignment relation $\mathsf{PAR} \subseteq (\mathsf{P}, \mathsf{A}, \mathsf{D})$ indicates that p can perform a on d. PAR is the fundamental relationship that determines whether a principal has the authority to access a specific data item for a specific action. It encapsulates the final authorisation decision reached after considering all policies, obligations, context constraints, and purpose bindings.

$$(o1)\quad \textit{Obligation Fulfillment}$$
$$\forall \mathsf{p} \in \mathsf{P}, \forall \mathsf{a} \in \mathsf{A}, \forall \mathsf{d} \in \mathsf{D} : (\mathsf{p}, \mathsf{a}, \mathsf{d}) \in \mathsf{PAR}$$
$$\Rightarrow [\forall \mathsf{o} : (\mathsf{p}, \mathsf{a}, \mathsf{d}, \mathsf{o}) \in \mathsf{OB} \Rightarrow \text{satisfied}(\mathsf{o})]$$

Authorisation is granted only when all the obligations associated with the access tuple $(\mathsf{p}, \mathsf{a}, \mathsf{d})$ are fulfilled.

Let X denote the set of contextual attributes (e.g., time, location, device), V the set of possible attribute values, and S the optional set of context-affected services. We define the context assignment relation $\mathsf{CX} \subseteq (\mathsf{P} \cup \mathsf{D} \cup \mathsf{S}) \times \mathsf{X} \times \mathsf{V}$. This defines the context conditions that attribute-value bindings impose on principals, data items, or services.

$$(cx1)\quad \textit{Contextual Validity}$$
$$\forall \mathsf{p}, \mathsf{a}, \mathsf{d} : (\mathsf{p}, \mathsf{a}, \mathsf{d}) \in \mathsf{PAR}$$
$$\Rightarrow \texttt{ContextConstraintsHold}(\mathsf{p}, \mathsf{a}, \mathsf{d})$$

Access is conditionally granted based on the validity of all relevant contextual constraints at the time of request.

We define the delegation relation as $\mathsf{DEL} \subseteq \mathsf{P} \times \mathsf{Q} \times \mathsf{A} \times \mathsf{D}$. Here, $(\mathsf{p}, \mathsf{q}, \mathsf{a}, \mathsf{d} \in \mathsf{DEL})$ indicates that principal p delegates permission for action a on data item d to principal q.

$$(del1)\quad \textit{Transitive Delegation}$$
$$(\mathsf{p}, \mathsf{q}, \mathsf{a}, \mathsf{d}) \in \mathsf{DEL} \wedge (\mathsf{p}, \mathsf{r}, \mathsf{a}, \mathsf{d}) \in \mathsf{DEL}$$
$$\Rightarrow (\mathsf{p}, \mathsf{r}, \mathsf{a}, \mathsf{d}) \in \mathsf{DEL}$$

Delegation preserves access semantics through transitive propagation.

$$(del2)\quad \textit{Delegation-Based Authorisation}$$
$$(\mathsf{p}, \mathsf{a}, \mathsf{d}) \in \mathsf{PAR} \wedge (\mathsf{p}, \mathsf{q}, \mathsf{a}, \mathsf{d}) \in \mathsf{DEL}$$
$$\Rightarrow (\mathsf{p}, \mathsf{a}, \mathsf{d}) \in \mathsf{PAR}$$

Delegation permits effective access transfer to delegatees.

Let U denote the set of access purposes (e.g., research, diagnostics, compliance). We define two relations:

- $\texttt{IntPurp} \subseteq \mathsf{D} \times \mathsf{U}$: Intended Purpose Mapping defines the allowable purposes for each data item.

- $\texttt{AccPurp} \subseteq \mathsf{P} \times A \times \mathsf{D} \times \mathsf{U}$: Access Purpose Declaration, which represents the purpose declared by the principal when requesting access.

$$(pb1)\quad \textit{Purpose Compliance}$$
$$\forall \mathsf{p}, \mathsf{a}, \mathsf{d}, \mathsf{u} : (\mathsf{p}, \mathsf{a}, \mathsf{d}, \mathsf{u}) \in \texttt{AccPurp}$$
$$\Rightarrow \exists \mathsf{u}' : (\mathsf{d}, \mathsf{u}') \in \texttt{IntPurp} \wedge \mathsf{u} \subseteq \mathsf{u}'$$

Access is granted only when the declared purpose u matches the intended purpose of the data item d.

$$(pb2)\quad \textit{Extended Authorisation}$$
$$(\mathsf{p}, \mathsf{a}, \mathsf{d}) \in \mathsf{PAR} \Leftrightarrow$$
$$\left[\begin{array}{c} \exists \mathsf{c_p}, \mathsf{c_d} \in \mathsf{C} : (\mathsf{p}, \mathsf{c_p}) \in \mathsf{PCA} \wedge (\mathsf{d}, \mathsf{c_d}) \in \mathsf{DDICA} \\ \wedge (\mathsf{a}, \mathsf{c_d}, \mathsf{c_p}) \in \mathsf{ASCA} \\ \wedge \forall \mathsf{o} : (\mathsf{p}, \mathsf{a}, \mathsf{d}, \mathsf{o}) \in \mathsf{OB} \Rightarrow \text{satisfied}(\mathsf{o}) \\ \text{Context\_Constraints\_Hold}(\mathsf{p}, \mathsf{a}, \mathsf{d}) \end{array}\right]$$

Authorisation is granted if category mappings are satisfied, all obligations are fulfilled, and contextual conditions apply.

Example 1 (Smart Building HVAC System): Consider a medium-sized commercial smart building with 15 functional zones (offices, conference rooms, corridors and plant rooms) and 120 HVAC-related IoT sensors. These include 45 temperature sensors, 30 occupancy (PIR) sensors, 20 CO_2 sensors, 15 humidity sensors, and 10 airflow/pressure sensors. However, disclosing this information to third-party service providers (such as energy analysts or maintenance teams) raises privacy and sovereignty concerns. For instance, occupancy patterns in private offices may reveal sensitive tenant behaviour, whereas energy usage metrics may reveal operational details about a facility. To address this, we establish the SDI framework for regulating data flows.

Role: The SDI model specifies a set of pre-defined roles for several stakeholders, each with specific responsibilities and permissions. These roles include `FacilityManager`, who is responsible for monitoring the overall efficiency of the building; `HVAC_Technician`, who handles diagnostics and repairs related to heating, ventilation, and air conditioning systems; `SecurityOperator`, who manages surveillance feeds and emergency protocols; and `TenantAdmin`, who oversees tenant-specific settings and preferences. Each role is linked to a specified degree of access and qualifications. A tuple like (`HVAC_Technician`, {(`Zone`, "East Wing"), (`Certification`, "REHVA")}) indicates that the HVAC technician has a professional REHVA certification and permission to run the HVAC systems in the East Wing, therefore guaranteeing competence in managing HVAC-related activities in that area. These role-attribute associations are initially defined by system administrators or integrated IAM systems using organisational policies and verified credentials (e.g., REHVA certification). The SDI policy engine dynamically enforces and updates these associations in response to contextual changes, such as emergency status or evolving project phases.

Data Items: OD refers to transient, real-time streams that provide immediate information about building conditions and usage patterns. This includes temperature data, which captures raw temperature readings with associated timestamps; occupancy data, which consists of motion sensor logs indicating statuses such as "occupied" or "vacant" and anonymised visitors counts; and energy usage, which records energy consumption aggregated hourly. A data tuple such as (`TemperatureData`, {(`Timestamp`, "2025-03-01T 14:00"), (`Value`, 22°C)}) represents a real-time temperature reading, whereas (`OccupancyData`, {(`Timestamp`, "2025-03-01T14:00"), (`Status`, "Occupied")}) and (`EnergyUsage`, {(`Timestamp`, "2025 - 03 - 01T14: 00"), (`Value`, 150 kWh)}) show simultaneous occupancy and energy usage records. AD on the other hand, refers to long-term records used for historical analysis and system documentation. This includes maintenance logs that detail HVAC system diagnostics and repairs—such as (`MaintenanceLog`, {(`Date`, "2025-02-28"), (`Issue`, HVAC Filter Replacement)}) and BIM Metadata, which describes structural and operational aspects of the building, such as (`BIM_Metadata`, {(`BuildingID`, "Building_ 01"), (`Zone`, "Floor_5/Conference_Room")}).

Services: These data streams and records are consumed by a variety of services that enable intelligent building operations. The facility management system (FMS) monitors overall building efficiency, the energy analytics platform optimises HVAC performance using aggregated data, and the maintenance portal provides HVAC repair diagnostics. A service tuple, (`FMS`, {(`ServiceID`, "FMS-01"), (`Feature`, "Energy Optimisation")}), identifies a specific FMS instance that optimises energy usage.

Environment: The SDI model's Environment governs contextual constraints, primarily through time-based policies and dynamic condition management. Policies, for example, may require that maintenance access be granted only during predefined times, reducing operational disruptions while maintaining access control. Additionally, the environment can respond to changing circumstances, such as emergencies, by dynamically updating role-category assignments, allowing for timely and appropriate responses. When the EmergencyStatus is set to Active, the system modifies RCA by adding relevant roles and data access privileges, such as ($\mathsf{RCA} \leftarrow \mathsf{RCA} \cup$ {`SecurityOperator`, `EmergencyExitData`}), allowing `SecurityOperator` to access critical emergency-related data like exit routes. This adaptive behaviour ensures that relevant stakeholders have timely access to the information needed for an effective emergency response.

We categorise data into distinct levels based on confidentiality and intended access. Public data includes aggregated energy usage metrics (e.g., total daily kWh for Floor 5), which can be easily shared. Restricted Data refers to sensitive operational inputs, such as raw occupancy logs from high-security zones like executive offices, that require controlled access. Personalised inputs, such as tenant-specific temperature setpoints, are considered confidential data and must be kept private.

To enforce data protection, the model employs a combination of policy enforcement techniques. Edge filtering allows for configurable data sanitization, such as removing timestamps or location identifiers based on data category, user role, and contextual policies, prior to transmission to the PDH. In addition, raw temperature data from restricted zones is stored locally unless explicitly shared. Dynamic access control ensures that only authorised roles can access relevant data. The FMS system gives Facility Managers access to aggregated energy data and HVAC diagnostics; Maintenance Teams have access to HVAC error logs during scheduled maintenance windows through time-based role-specific access; and Energy Analysts can use anonymised kWh metrics but not view occupancy trends or tenant-specific settings.

The overall outcome is a system that follows IDS standards by requiring data anonymization and role-based access. For example, if the temperature in a conference room rises above 25 °C, the FMS will start cooling while masking occupancy data to protect tenant privacy. Meanwhile, energy analysts can improve building performance by using high-level metrics that do not reveal sensitive or operational information.

The model includes role hierarchies and attribute assignments to allow for such fine-grained control. Devices are assigned attributes, such as (HVAC − 101, {(Location, "Floor_5/Conference_Room"), (SecurityLevel, "Restricted")}), and data items are similarly classified, for example, (TemperatureData, {(Category, "Restricted"), (Timestamp,"2025-03-01T14:00")}). Services like the FMS are defined with access levels and operational zones, such as (FMS, {(AccessLevel, "Aggregated"), (Zone, "Floor_5")}), whereas roles like HVAC Technicians are tied to specific data categories and zones, e.g., (HVAC_Technician, {(Category, "Restricted"), (Zone, "EastWing")}). Restricted can inherit permissions from Confidential by meeting certain attribute-based conditions, such as $\texttt{Restricted} \subseteq \texttt{Confidential}$. This enables flexible but secure access delegation.

Finally, Dynamic Policy Specification ensures that access rules are updated in real-time. During an emergency, the policy engine updates RCA with relevant data ($\texttt{Emergency Status} = \texttt{Active} \Rightarrow \mathsf{RCA} \leftarrow \mathsf{RCA} \cup \{\texttt{Security Operator}, \texttt{EmergencyExitData}\}$). Furthermore, temporal constraints are enforced through blockchain-based smart contracts, which automatically adjust or revoke permissions based on building operations or project phases, ensuring accountability and flexibility.

VI. GRAPH-BASED POLICY REPRESENTATION

We present a graph-based policy representation framework for SDI that enforces dynamic, context-aware access control in smart building ecosystems, drawing on [31], [32]. This framework represents policies as labelled graphs, with nodes representing entities (e.g., IoT devices, data items, roles, and services) and edges defining permissible interactions between them. By incorporating hierarchical relationships and RBAC [6], this model ensures not only intuitive visualisation and machine-executable enforcement, but also compliance with IDS principles [2].

A. SDI POLICY GRAPHS

SDI policy graph is a formal model for expressing access control policies using a labelled, undirected graph $\mathcal{G} = (V, E, l_v, l_e)$, where V represents various entities in the system and the edges E capture the allowed interactions among them. Each node and edge is labelled with functions l_v and l_e, which assign rich metadata such as attributes, categories, contextual constraints, and roles, allowing for scalable and fine-grained access control enforcement.

Definition 3 (Record): A record is a structured set of attributes and values. Each attribute name a_i is drawn from a finite set, as is each value t_i. A record $\mathsf{R} \in \texttt{REC}$ has the form $\{\mathsf{a}_1 = \mathsf{t}_1, \ldots, \mathsf{a}_n = \mathsf{t}_n\}$, where each attribute appears exactly once. We denote attribute selection by $\mathsf{R}.\mathsf{a}$ and if a record R needs to be updated, the function $\texttt{update}(\mathsf{R}, \mathsf{a}, \mathsf{t})$ assigns the value t to the attribute a, either modifying an existing field or creating one if it does not exist. Every record has a mandatory attribute ent that specifies which entity it is associated with.

Example 2 (HVAC Sensor Record): For instance, an HVAC sensor might have the record $\mathsf{R} = \{\texttt{ent} = \texttt{HVAC} - \texttt{101}, \texttt{type} = \mathsf{D}, \texttt{Location} = \text{"Lobby"}, \texttt{SecurityLevel} = \text{"High"}, \ldots\}$, allowing queries like $\mathsf{R}.\texttt{ent} = \texttt{HVAC} - \texttt{101}$, and $\texttt{update}(\mathsf{R}, \texttt{SecurityLevel}, \text{"Medium"})$ results in the new record $\{\texttt{ent} = \texttt{HVAC} - \texttt{101}, \texttt{type} = \mathsf{D}, \texttt{Location} = \text{"Lobby"}, \texttt{SecurityLevel} = \text{"Medium"}, \ldots\}$.

Definition 4 (Policy SDI$_\mathcal{G}$): The SDI policy graph is formally defined as a tuple $\mathcal{G} = (V, E, l_v, l_e)$, where V is a finite set of vertices (nodes) and E is the set of undirected edges connecting these nodes. Using the labelling function $l_v : V \to \texttt{REC}$, each node $v \in V$ is assigned a record structure and metadata. Similarly, each edge $e \in E$ has a labelling function $l_e : E \to \texttt{REC}$, which captures the properties of the interaction between two adjacent nodes v_1 and v_2 such that $l_e(e).\texttt{adj} = v_1, v_2;\ v_1 \neq v_2$. Importantly, each node label contains the attribute ent, which denotes the entity associated with the node.

Nodes are typed using a field called type, and their values fall within the predefined set

$$\{\mathsf{D}, \mathsf{OD}, \mathsf{AD}, \mathsf{C}, \mathsf{A}, \mathsf{S}, \mathsf{R}\}$$

such that $l_v(v).\texttt{type} = \mathsf{D}$ if $l_v(v).\texttt{ent} = \mathsf{d} \in \mathsf{D}$.

The type of each edge is determined by the types of its connected nodes. If an edge e connects nodes v_1 and v_2, its type is a pair $(l_v(v_1).\texttt{type},\ l_v(v_2).\texttt{type})$. Because the graph is undirected, edge types are assumed to be symmetric; for example, (D, C) and (C, D) are treated as equivalent.

Definition 5 (Hierarchy): Categories form a hierarchy $\mathsf{C}_1 \subseteq \mathsf{C}_2$, where C_1 inherits permissions from C_2 if $\texttt{Cond}(\mathsf{C}_1) \subseteq \texttt{Cond}(\mathsf{C}_2)$. Similarly, roles form a hierarchy $\mathsf{R}_1 \subseteq \mathsf{R}_2$, where R_1 (e.g., HVAC_Technician) inherits permissions from R_2 (e.g., FacilityManager).

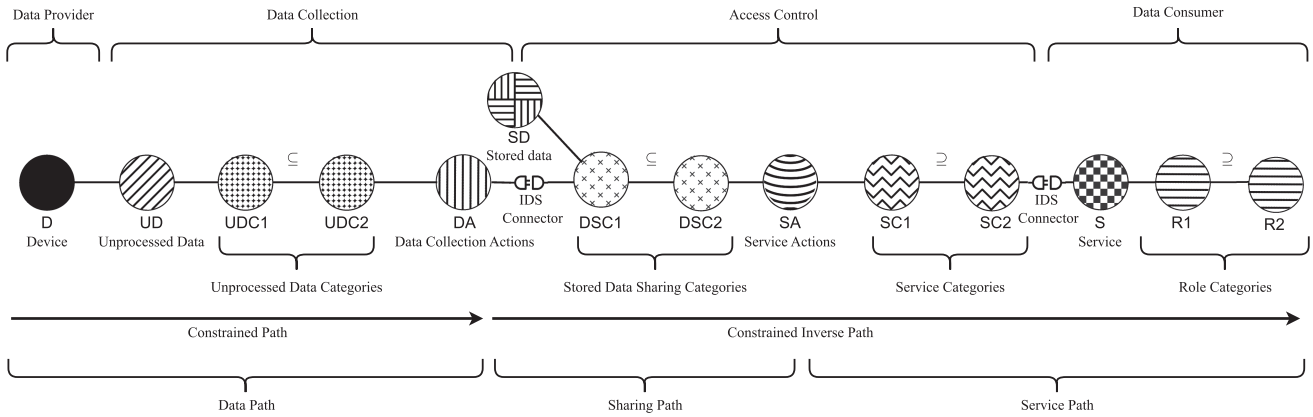

FIGURE 5. Authorisation and data dissemination path in SDI policy graph.

B. AUTHORISATION PATHS IN SDI

Authorisation is enforced through constrained paths that traverse nodes and edges while adhering to role-based and hierarchical constraints.

Definition 6 (Authorisation Path): Fig. 5 shows an authorisation path describes the end-to-end flow of data access and usage through four interconnected segments: data path, sharing path, service path, and role-data path. The data path describes how raw data, such as an occupancy sensor D, flows from its source as operational data OD, is classified into a category C_{OD} as "Restricted", undergoes a corresponding authorisation action A encrypted, and results in derived or anonymised data DA. The sharing path involves the further processing of anonymised data DA into a classified version C_{DA} as "Public," followed by an authorised sharing action A may be shared through a "Visualise". The service path connects the authorisation action A to the target service S, which may grant access to a downstream service like an Energy Analytics platform. Finally, the role-data path connects a role R "HVAC_Technician" to a data category C, which can be associated with the "Restricted" category through an authorised action A that allows it to perform the "Encrypt" action.

Example 3 (HVAC Data Path): An authorisation path may begin with a node of type D (HVAC sensor), followed by OD (temperature data), classified as "Restricted", encrypted through A (Encrypt), and stored as DA (encrypted temperature log).

C. ROLE-BASED SDI POLICIES

Roles are treated as first-class nodes in the policy graph and are linked to data categories and actions, allowing for dynamic permissions. Fig. 6 shows the SDI policy graph for a typical smart building ecosystem. A Role Node, like `HVAC_Technician`, can have permissions Encrypt, Share that are explicitly associated with a Category Node, like "Restricted". In dynamic situations, such as an emergency, the policy engine can assign roles accordingly. For example, if `EmergencyStatus` = "Active", an edge between `SecurityOperator` and `EmergencyExitData` is created to allow access.

VII. SDI POLICY ANALYSIS

Policy queries are essential tools for administrators to analyse and validate policies [33], particularly in a complex SDI model architecture. We focus on policy content queries, which ensure the structural integrity of policy graphs, and policy effect queries, which examine behavioural properties such as completeness and conflict-free configuration (a similar analysis can be seen in [34]). We demonstrate how to use graph-theoretic techniques on SDI policy graphs to perform these analyses effectively.

A. POLICY CONTENT QUERIES

Content queries in SDI assess the internal consistency and structural correctness of policy graphs. The following are typical queries about the components of the SDI policy framework, which are intended to ensure that the policy is well-formed, comprehensive, and aligned with data governance requirements in smart building environments.

- $Q1$: *(Data Categorisation)* Ensure that each data element is assigned to at least one category in C. Specifically, each OD node must connect to a c node through an edge labelled ODC and similarly, each AD node must connect through an edge labelled ADC.
- $Q2$: *(Action Association)* Verify that each category leads to valid actions. If c is a data category (e.g., "Restricted"), there must be a valid path to an action node (e.g., "Encrypt") in the A (e.g., collection set). If c is a service category (e.g., "Public"), A (e.g., Share) should provide a path to a sharing-related action (e.g., "Visualise").
- $Q3$: *(Category Membership)* Retrieve which data items belong to a given category c. This involves tracing from c backward to all OD/AD nodes connected through category edges.
- $Q4$: *(Role-Based Access)* Determine allowed actions for a role r. In the graph, trace paths $r \rightarrow c \rightarrow a$ for a role

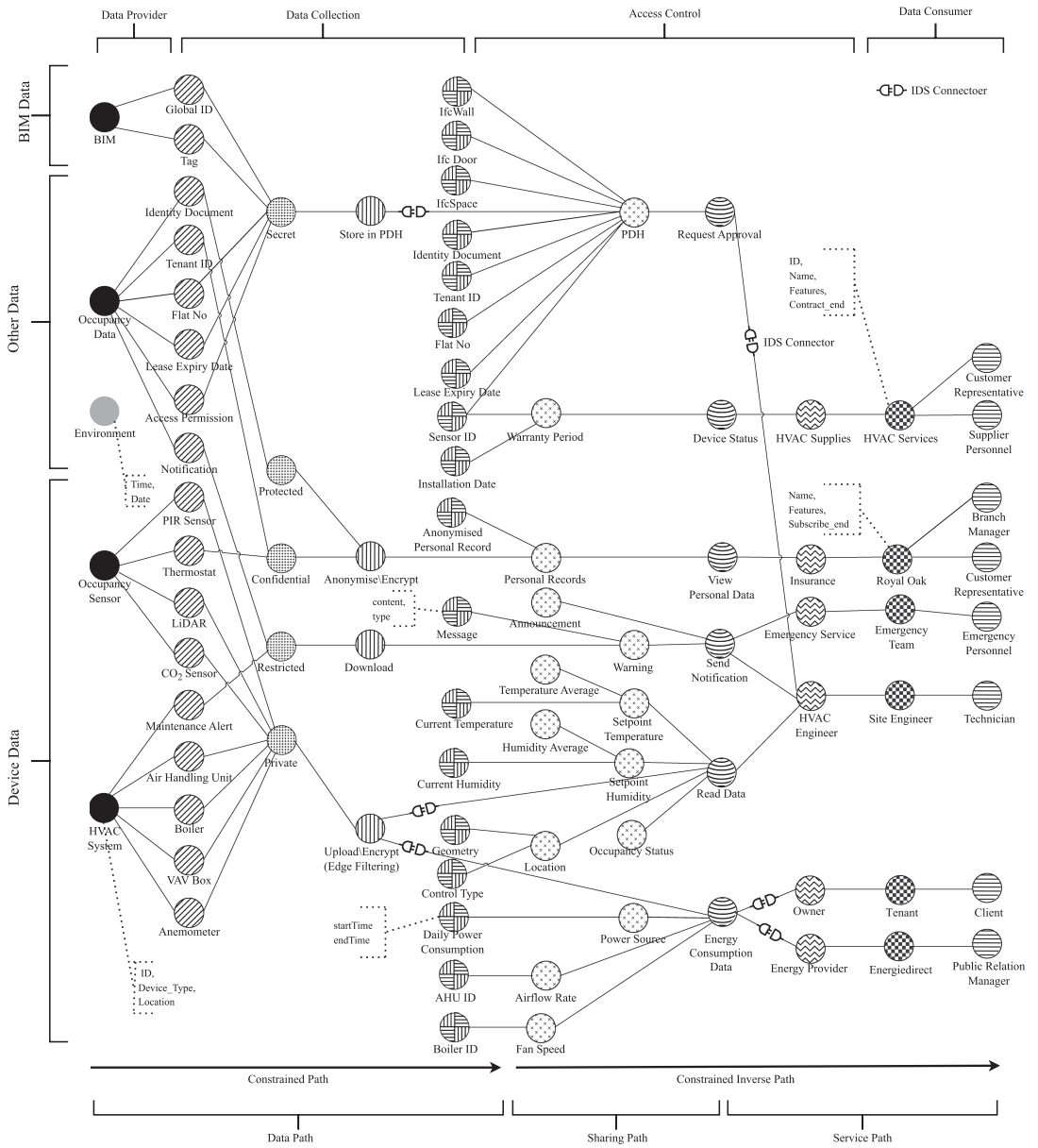

FIGURE 6. **An SDI policy graph for a smart building ecosystem.**

$\mathsf{r} \in \mathsf{R}$ (e.g., `HVAC_Technician`), where C is the category (e.g., "Restricted") and A is the allowed action (e.g., "Encrypt").

$\mathcal{Q}5$: *(Role Hierarchy Consistency)* Check that role-category edges respect the category hierarchy. When adding or removing edges between roles and categories ($\mathsf{R} \rightarrow \mathsf{C}$), the ordering $\mathsf{c}_1 \subseteq \mathsf{c}_2$ must be preserved. This means that the conditions associated with c_1 should logically imply those of c_2.

Example 4 (Emergency Role Activation): In emergency scenarios with `EmergencyStatus` = "Active", a query validates that a temporary link (e.g., `SecurityOperator` $\rightarrow \mathsf{C}\_$ `EmergencyExitData`) meets the emergency category condition.

Scalability Note: All content queries execute in polynomial time in relation to the size of the involved graph portion, making them computationally efficient even for large-scale policies.

B. POLICY EFFECT QUERIES

Effect queries are concerned with the accuracy and security of policy configurations. They help to ensure that the graph enforces all expected behaviours and does not contain conflicting permissions.

Definition 7 (Completeness: Totality): An SDI policy is total if it provides a definitive authorisation decision for each data source d that generates a data item d_i and for each service request. In other words, a policy is total if it defines valid permissions for all legitimate data-service combinations.

Assuming a well-formed policy graph $\mathrm{SDI}_{\mathcal{G}}$ as defined in Definition 4, the $\mathsf{AD}_{\mathcal{G}}$ and $\mathsf{ADS}_{\mathcal{G}}$ relations must include all tuples from A (e.g., collection set) $\times$ OD $\times$ DA and A (e.g., sharing) $\times$ AD $\times$ S, respectively. Each role $\mathsf{r} \in \mathsf{R}$ should have defined access rights to all data categories $\mathsf{c} \in \mathsf{C}$.

Definition 8 (Conflict Free): A policy graph $\mathrm{SDI}_{\mathcal{G}}$ is conflict-free if it does not authorise incompatible actions on the same data. To resolve conflicting actions a_1 and a_2:

- *Data conflicts:* No OD node should have paths that enable both a_1 and a_2.
- *Service Conflicts:* No DA node should be connected to conflicting service actions through its policy path.

Example 5 (Resolving Policy Conflicts): If a policy unintentionally permits both ShareRaw and Aggregate on BIM metadata:

- A Cypher query finds conflicting authorisation paths, such as $\mathsf{DA} \rightarrow \mathit{Public} \rightarrow \mathit{ShareRaw}$ and $\mathsf{DA} \rightarrow \mathit{Public} \rightarrow \mathit{Aggregate}$.
- A smart contract is then executed to revoke improper permissions.

Property 1 (Conflict Detection): Conflicts are identified by checking for overlapping paths that permit mutually exclusive actions. Future conflicts may necessitate reachability analysis in order to evolve policies, which is currently being investigated.

Example 6 (Verification through Graph Paths): Let $\mathrm{SDI}_{\mathcal{G}}$ be a well-formed policy graph. Assume that two ACD actions da_1 and da_2 (or ACS actions sa_1 and sa_2) are mutually exclusive. The policy graph $\mathrm{SDI}_{\mathcal{G}}$ ensures that da_1 and da_2 do not conflict if, for each $\mathsf{ud} \in \mathsf{UD}$ node, the set of authorisation paths beginning with ud does not include paths through both da_1 and da_2. Similarly, for sa_1 and sa_2, the set of authorisation paths beginning with a node $\mathsf{sd} \in \mathsf{SD}$ does not include paths through both sa_1 and sa_2.

Note: If the policy represented by the graph $\mathrm{SDI}_{\mathcal{G}}$ is dynamic, the absence of conflict at a given time does not ensure conflicts cannot arise in the future. For this, a reachability analysis is required, which we leave for future work.

C. PERFORMANCE EVALUATION

Let $G = (V, E)$ denote the SDI policy graph. Full structural validation includes reachability and consistency checks over G, with a worst-case complexity of $\mathcal{O}(|V| + |E|)$. In practise, policy updates are incremental and limited to bounded subgraphs, allowing for tractable verification even as data sources grow. Runtime authorisation is based on constrained traversal of predefined policy layers rather than unrestricted graph searches. A typical decision follows a bounded path with a maximum depth of d that is independent of the total system size. Let b represent the branching factor between categories and actions. The worst-case evaluation is $\mathcal{O}(b^d)$; however, since both d and b are structurally bound by design, the enforcement cost is effectively constant relative to the number of data sources. Contextual attribute checks add an additional $\mathcal{O}(k)$ factor, where k represents the bounded number of relevant attributes. Therefore, SDI achieves polynomial-time structural validation with bounded per-request authorisation overhead, demonstrating scalability in realistic smart building expansion scenarios.

VIII. CONCLUSION AND FUTURE WORK

We presented SmartData Interlock (SDI), a novel access control architecture for smart building data ecosystems, together with Personalised Data Hubs (PDHs), a decentralised data-sharing framework. By integrating PDHs with the SDI model, we address key challenges in IoT ecosystems for built environments, including data sovereignty, privacy preservation, and dynamic, fine-grained access control. SDI within PDHs ensures that sensitive information is anonymised or restricted before sharing, while SDI policy graphs allow stakeholders to define and enforce context-aware rules. These contributions lay the groundwork for secure and compliant data collaboration in smart buildings. Beyond architectural design, this work analytically evaluates SDI using a realistic smart building policy graph, demonstrating that data exchange can be achieved with bounded computational overhead and that the approach is operationally viable for smart building deployments. Future work will focus on prototyping PDHs and SDI on building management systems.

REFERENCES

[1] F. A. Ghansah, "Digital twins for smart building at the facility management stage: A systematic review of enablers, applications and challenges," *Smart Sustain. Built Environ.*, vol. 14, pp. 1194–1229, 2024.

[2] M. Fernández, J. Jaimunk, and B. Thuraisingham, "A privacy-preserving architecture and data-sharing model for cloud-IoT applications," *IEEE Trans. Dependable Secure Comput.*, vol. 20, no. 4, pp. 3495–3507, Jul./Aug. 2023.

[3] D. Guinard, V. Trifa, F. Mattern, and E. Wilde, "From the Internet of Things to the web of things: Resource-oriented architecture and best practices," in *Architecting the Internet of Things*. Berlin, Germany: Springer, 2011, pp. 97–129.

[4] G. Epiphaniou, P. Pillai, M. Bottarelli, H. Al-Khateeb, M. Hammoudesh, and C. Maple, "Electronic regulation of data sharing and processing using smart ledger technologies for supply-chain security," *IEEE Trans. Eng. Manag.*, vol. 67, no. 4, pp. 1059–1073, Nov. 2020.

[5] A. Ahmed, "Context-aware access control in ubiquitous computing (CRAAC)," Ph.D. dissertation, Univ. Manchester, Manchester, U.K., 2010.

[6] R. S. Sandhu, "Role-based access control," *Adv. Comput.*, vol. 46, pp. 237–286, 1998.

[7] A. Dorri, M. Steger, S. S. Kanhere, and R. Jurdak, "BlockChain: A distributed solution to automotive security and privacy," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 119–125, Dec. 2017.

[8] A. Kovach, L. Montalvillo, J. Lanza, P. Sotres, and A. Urbieta, "Understanding data spaces: A systematic mapping study of foundations, technical building blocks, and sectoral adoption," *Comput. Sci. Rev.*, vol. 59, 2026, Art. no. 100819.

[9] R. Sharma and B. Villanyi, "A spatiotemporal order-revealing query processing approach for industrial Internet of Things," *J. King Saud Univ.- Comput. Inf. Sci.*, vol. 34, no. 10, pp. 8985–8995, 2022.

[10] J. Singh, J. Powles, T. Pasquier, and J. Bacon, "Data flow management and compliance in cloud computing," *IEEE Cloud Comput.*, vol. 2, no. 4, pp. 24–32, Jul./Aug. 2015.

[11] P. Nguyen, H.-H. Nguyen, P. Phung, H.-L. Truong, and T. Cheung, "Advanced context-sensitive access management for edge-driven iot data sharing as a service," *ACM Trans. Internet Technol.*, vol. 25, no. 2, pp. 1–31, 2025.

[12] M. Fernández, M. Kantarcioglu, and B. Thuraisingham, "A framework for secure data collection and management for Internet of Things," in *Proc. 2nd Annu. Ind. Control System Secur. Workshop*, 2016, pp. 30–37.
[13] C. Dwork et al., "The algorithmic foundations of differential privacy," *Foundations Trends Theor. Comput. Sci.*, vol. 9, no. 3/4, pp. 211–407, 2014.
[14] S. Ravidas, A. Lekidis, F. Paci, and N. Zannone, "Access control in internet-of-things: A survey," *J. Netw. Comput. Appl.*, vol. 144, pp. 79–101, 2019.
[15] J. Qian, S. Hinrichs, and K. Nahrstedt, "ACLA: A framework for access control list (ACL) analysis and optimization," in *Proc. Commun. Multimedia Secur. Issues New Century, IFIP TC6/TC11 5th Joint Work. Conf. Commun. Multimedia Secur.*, 2001, pp. 197–211.
[16] G. Batra, "Attribute-based access control," in *Encyclopedia of Cryptography, Security and Privacy*. Cham, Swizerland: Springer, 2024, pp. 131–133.
[17] A. Chaudhry et al., "Personal data: Thinking inside the box," in *Proc. 5th Decennial Aarhus Conf. Crit. Alternatives*, 2015, pp. 29–32.
[18] N. Mishra and H. Levkowitz, "PDV: Permissioned blockchain based personal data vault using predictive prefetching," in *Proc. 3rd Blockchain Internet Things Conf.*, 2021, pp. 59–69.
[19] Y.-A. De Montjoye, E. Shmueli, S. S. Wang, and A. S. Pentland, "openPDS: Protecting the privacy of metadata through safeanswers," *PLoS One*, vol. 9, no. 7, 2014, Art. no. e98790.
[20] J. A. Hoagland, "Specifying and implementing security policies using LaSCO, the language for security constraints on objects," Ph.D. dissertation, Univ. Calif., Davis, Davis, CA, USA, 2000.
[21] A. Heydon, M. W. Maimone, J. Tygar, J. M. Wing, and A. M. Zaremski, "Miro: Visual specification of security," *IEEE Trans. Softw. Eng.*, vol. 16, no. 10, pp. 1185–1197, Oct. 1990.
[22] M. Kehlenbeck, T. Sandner, and M. H. Breitner, "Managing internal control in changing organizations through business process intelligence-a service oriented architecture for the XACML based monitoring of supporting systems," in *Proc. 43rd Hawaii Int. Conf. System Sci.*, 2010, pp. 1–10.
[23] J. H. Huh et al., "Next-generation access control for distributed control systems," *IEEE Internet Comput.*, vol. 20, no. 5, pp. 28–37, Sep./Oct. 2016.
[24] J. Singh and N. K. Chaudhary, "OAuth 2.0: Architectural design augmentation for mitigation of common security vulnerabilities," *J. Inf. Secur. Appl.*, vol. 65, 2022, Art. no. 103091.
[25] S. Barker, "The next 700 access control models or a unifying meta-model?," in *Proc. 14th ACM Symp. Access Control Models Technol.*, 2009, pp. 187–196.
[26] M. Fernández, A. Franch Tapia, J. Jaimunk, M. Martinez Chamorro, and B. Thuraisingham, "A data access model for privacy-preserving cloud-IoT architectures," in *Proc. 25th ACM Symp. Access Control Models Technol.*, 2020, pp. 191–202.
[27] N. Luo, G. Fierro, Y. Liu, B. Dong, and T. Hong, "Extending the brick schema to represent metadata of occupants," *Automat. Construction*, vol. 139, 2022, Art. no. 104307.
[28] A. Ramachandruni, R. Sharma, E. Petrova, and P. Pauwels, "Vocabulary hub for semantic data exchange and interoperability in an AECO domain-specific dataspace," in *Proc. Eur. Conf. Comput. Construction 42nd CIB W78 Conf. IT Construction*, 2025, pp. 1–8.
[29] R. Sharma and B. Villányi, "A sustainable ethereum merge-based big-data gathering and dissemination in IIoT system," *Alexandria Eng. J.*, vol. 69, pp. 109–119, 2023.
[30] J. Pampus, B.-F. Jahnke, and R. Quensel, "Evolving data space technologies: Lessons learned from an IDS connector reference implementation," in *Proc. Int. Symp. Leveraging Appl. Formal Methods*, 2022, pp. 366–381.
[31] S. Alves and M. Fernández, "A graph-based framework for the analysis of access control policies," *Theor. Comput. Sci.*, vol. 685, pp. 3–22, 2017.
[32] M. Fernández, J. Jaimunk, and B. Thuraisingham, "Graph-based data-collection policies for the Internet of Things," in *Proc. 4th Annu. Ind. Control System Secur. Workshop*, 2018, pp. 9–16.
[33] E. Bertino et al., "Analysis of privacy and security policies," *IBM J. Res. Develop.*, vol. 53, no. 2, pp. 3:1–3:18, Mar. 2009.
[34] A. Armando and S. Ranise, "Automated and efficient analysis of role-based access control with attributes," in *Proc. Annu. Conf. Data Appl. Secur. Privacy*, 2012, pp. 25–40.

RAVI SHARMA (Member, IEEE) received the PhD degree in computer science and engineering, focusing on the Industrial Internet of Things (IIoT) and industry 4.0 (I4) from the Department of Electronics Technology, Budapest University of Technology and Economics, Budapest, Hungary, in 2024, and the master's degree in computer science and engineering from the Indian Institute of Technology, Patna, India. He is currently a postdoctoral researcher with Information Systems in Built Environment, Eindhoven University of Technology, Eindhoven, The Netherlands. His research interests include building secure, intelligent, and sustainable IIoT systems that enable trustworthy data flow from sensors to enterprise decision-making, addressing critical gaps in I4, digital twins, and cyber-physical security.

AVINASH RAMACHANDRUNI (Graduate Student Member, IEEE) received the MTech degree in structural engineering from the Indian Institute of Technology Hyderabad, India, in 2019. He is currently working toward the PhD degree in construction informatics with the Eindhoven University of Technology, Eindhoven, The Netherlands. His research interests include decentralised information systems, data security, and interoperability, and smart buildings and cities.

EKATERINA PETROVA received the PhD degree in civil engineering from Aalborg University, Denmark, in 2019. She was a visiting researcher with the Department of Architecture and Urban Planning, Ghent University, in 2018. In 2021, she joined the Information Systems in the Built Environment Research Group, Eindhoven University of Technology, where she is currently an Assistant Professor of artificial intelligence in construction with the Department of the Built Environment. Her research interests include the integration of various symbolic and statistical artificial intelligence approaches for decision support in performance-oriented building design and engineering, artificial intelligence in the built environment, ambient intelligence, cognitive approaches, smart buildings, building information modelling and management, semantic web technologies, implementation of symbolic and statistical artificial intelligence approaches for decision support in the context of sustainable building design, circular buildings, digital twins, and data-driven smart buildings.

PIETER PAUWELS is currently an Associate Professor with the Department of the Built Environment, Eindhoven University of Technology. From 2008 to 2019, he was with the Department of Architecture and Urban Planning, Ghent University. With a lot of experience and knowledge in computer science and software development, he is involved in several industry-oriented research projects on topics affiliated with AI in construction, design thinking, building information modelling (BIM), linked building data (LBD), linked data in architecture and construction (LDAC), and semantic web technologies. His research interests include information system support for the building life cycle, such as architectural design, construction, and building operation.