Follow the Schema: Finding the Valve Before the Flood

Navian Francis
University of Virginia
Charlottesville, Virginia, USA
naf7nh@virginia.edu

Sriya Gandikota
University of Virginia
Charlottesville, Virginia, USA
tdm8qg@virginia.edu

Shaina Kumar
University of Virginia
Charlottesville, Virginia, USA
mzm2cj@virginia.edu

Nester Phiri
University of Virginia
Charlottesville, Virginia, USA
dzv2hf@virginia.edu

ACM Reference Format:
Navian Francis, Sriya Gandikota, Shaina Kumar, and Nester Phiri. 2025. Follow the Schema: Finding the Valve Before the Flood. In . ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Abstract

Buildings are increasingly integrating cyber-physical systems within buildings. This paper examines the use of the Brick schema, a standardized metadata framework, to identify and localize valves during leak scenarios. The Brick schema is based on a mapping of valves, pipe segments, fixtures, and flow relationships across a building, independent of existing asset systems. The schema can be extended with domain-specific metadata and paired with a reasoning engine to provide actionable guidance for emergency response and maintenance workflows. The results from multiple test cases validate the schema's alignment with Brick/UVA conventions and demonstrate its capacity to balance containment efficiency with operational continuity. The model can identify optimal valves for leak isolation and quantify room-level service outages, underscoring its potential to enhance emergency response, maintenance planning, and overall building resilience. The system transforms static building information into actionable insights, laying the foundation for more scalable, intelligent infrastructure management tools.

2 Introduction

The Link Lab in Olsson Hall at the University of Virginia is an ideal location for analyzing the effects of rapid infrastructure identification during an emergency. Events such as pipe bursts require efficient identification and localization to mitigate damage and ensure occupant safety. A general tool for navigating such situations has yet to be found. This paper examines Brick schema integration, a "standardized metadata framework" for valves and sensors, as the primary tool used in emergencies. The Brick Schema, with its mobile infrastructure, can provide situational awareness and accurate navigational insights during disasters.

Link Lab can represent a set of challenges that commercial and institutional buildings experience. Water and utility networks, with limited staff, further underscore the need for a rapid response. This study highlights the broader need for scalable solutions that can be translated to other building types.

Emergencies, leaks, and bursts rarely happen in isolation. A semantic framework like Brick unifies data from sensors, valves, and spatial descriptors into a model that supports decision-making during a crisis. This work demonstrates how the Brick schema can serve as an effective tool for building resilience by pinpointing problems and compiling functional responses. This can be a step in efficient infrastructure management.

3 Problem Statement

The Brick schema has enabled standardized metadata representation in buildings. The schema was designed to be complete, expressive, and usable to support the development of more efficient, centralized applications for operation, management, and fault diagnosis [1]. Thus, the uniform schema addresses interoperability issues arising from the increased integration of cyber-physical systems within buildings by capturing the devices in a building and their relationships. However, when equipment or sensors require manual, hands-on repair, the metadata for entity encoding locations does not ensure technicians can physically locate and access those devices. Although the Brick schema facilitates device identification, this project extends it by identifying the appropriate valves required when leaks occur.

Traditional facility management systems rely on documents, internal systems, and phone calls, which are not very effective at saving time. This is because existing digital systems only provide location data as text, which makes it difficult for the maintenance team to locate the actual broken utility component, especially in large buildings. For large buildings such as universities and hospitals, which are increasingly becoming smart buildings, there is a need to develop management systems that support efficient workflows.

Research has been conducted to integrate the use of specific-utility sensors; however, this approach has narrowed the location to the room with the broken utility rather than the actual wall. The use of Geographical Information Systems (GIS) and indoor maps has proven very useful and efficient, but still causes delays in locating the specific broken utility, as they do not use 3D visual data or real-time diagnostics to pinpoint the exact location of the burst pipe [9]. There has also been work on the joint use of Internet of Things

(IoT) devices and sensors, Indoor Positioning Systems (IPS), and Building Information Modeling (BIM) to address this issue [8]. This solution would have the system collect real-time data on people and tagged objects and store it in the BIM database for real-time monitoring. This, however, has not provided much detail on the use of this combined system for the maintenance and actual 3D location of the broken utility infrastructure in the building.

Furthermore, navigation in modern buildings is an area increasingly explored with the advent of cost-effective sensors and the development of new technologies, such as augmented reality (AR), artificial intelligence, and Wi-Fi, among others [3, 5]. As devices such as smartphones become more powerful alongside these developments, they are being integrated into solutions that enable users to navigate their spaces efficiently and with confidence. While current in-building navigation methods, such as WiFi, QR codes, AR, and others, exist, they do not provide the required accuracy to guide users to individual devices within a building [14, 11]. In an emergency, such as a burst pipe causing flooding, the current methods alone would not guide someone to the correct valve for that pipe.

4 Motivation

The complexity of building water networks (both domestic and emergency lines) has made responding to leaks challenging. When a pipe failure occurs, building facilities must respond quickly and efficiently to minimize damage. Conventional approaches to this management have focused on locating the leak but have failed to determine which valves to shut off and isolate. Shutting off a single valve may trigger a cascading effect, blocking critical water zones in areas such as hospitals, laboratories, and even heating/cooling systems. There is a need for a tool that extends the semantic building model to assist facility management in preventing critical infrastructure impacts.

The increase in sensors and equipment in commercial buildings also requires greater maintenance by facility managers. However, facility management teams are often small, especially given the number of emergency work orders related to equipment or control infrastructure that must be resolved each year. The improper maintenance of that infrastructure contributes to considerable energy waste [6]. Although efforts are underway to enhance preventive maintenance, including automated fault detection and diagnostics and adaptive occupant-driven controls, spatially useful tools for effective maintenance response are still needed when failures inevitably occur.

Therefore, it is essential to evaluate how well a Brick-based modeling approach represents complex water-network scenarios. These simulations will be fairly simple, but the goal is to test whether the Brick schema could provide evaluation cases. Semantic models should support the building's dynamic infrastructure to enable rapid diagnosis in the event of a pipe failure.

5 Methodology

To address the primary concern of facility management teams, the system must enable rapid, efficient identification and localization of valves, especially for on-site staff who lack expertise or familiarity with the building infrastructure. The goal is to develop

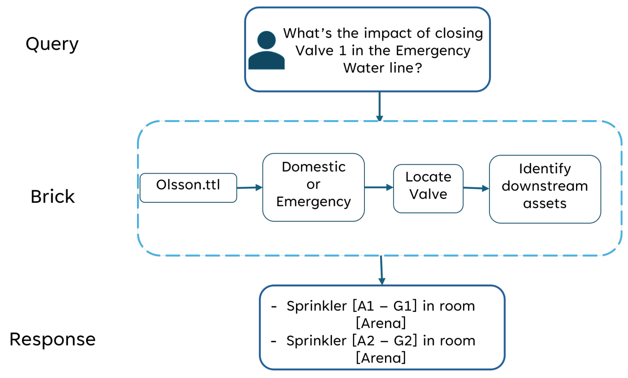

Query
What's the impact of closing Valve 1 in the Emergency Water line?
Brick
Olsson.ttl
Domestic or Emergency
Locate Valve
Identify downstream assets
Response
- Sprinkler [A1 – G1] in room [Arena]
- Sprinkler [A2 – G2] in room [Arena]

Figure 1: System Overview

a system that allows occupants and personnel to locate the necessary valves to be isolated and shut off without relying on costly automated flow valves. The valves of concern are not limited to those closest to the leaks, as it may be necessary to identify multiple upstream valves or strategically design the outage process around the results of the shut-off. Thus, the fundamental components of the system include the ability to identify the valves that source the leaks and to determine the implications of shutting off a particular valve, including consequential events and how water may need to be redirected to accommodate the building's needs.

5.1 System Overview

For the scope of this project, we focus on applying this system to one floor of the Link Lab, an Engineering building at UVA. To implement this system, we created a modified brick schema to capture the various water supplies (their properties, relationships, and locations) across the floor, independent of existing asset systems. The schema captures necessary attributes, such as location, flow direction, and connected equipment. It can also capture additional attributes, such as valve materials and sizes, to account for cases where standard metadata is insufficient to identify leaks that are not directly at pipe endpoints. The tool was designed generically so that it may be adapted to any context or building, as the underlying idea that valves are connected through a network, both expandable and reducible, will always hold. The semantic model of the water network encoded in the Brick schema must be complemented by a reasoning engine that implements graph-search logic. In other words, this schema will facilitate reasoning about water flow and, ultimately, which valves to shut off, as shown in Figure 1. At the same time, the application logic queries the constructed model to trace water-flow connections upstream and downstream via the standardized Brick relationships (*hasLocation, hasPart, isPartOf, and feeds*). The modified Brick schema directly draws from Brick's ontology, which is open source and publicly available on GitHub[2].

5.2 Customized Brick Schema

We build on the Brick schema to model our water/valve network, using a small UVA-specific schema based on the Resource Description Framework (RDF) to define the entities (valves, rooms, pipe segments) and their relationships. This framework represents

Table 1: Brick Ontology to Locate Valve

Relationship	Subject	Base	Object
Location	PipeSegment_(RoomX)	hasLocation	Room_X
Location	Leak_Alarm_(RoomX)	hasLocation	Room_X
Composition	Room_X	hasPart	Leak_Alarm_(RoomX)
Composition	Room_X	hasPart	PipeSegment_(RoomX)
Flow Control / Topology	Valve_Isolation_(X)	feeds	PipeSegment_(RoomX)
Flow Control / Topology	PipeSegment_(RoomX)	feeds	Fixture_(RoomX)
Status Sensing	Valve_Isolation_(X)	hasPoint	Valve_Position_Sensor
Actuation Logic	Valve_Isolation_(X)	hasPoint	Valve_Command
Domain Metadata	PipeSegment_(RoomX)	uva:hasDomain	uva:EmergencyWater / uva:DomesticWater
Valve Metadata	Valve_Isolation_(X)	uva:hasNominalDiameter	Diameter_Value
Valve Metadata	Valve_Isolation_(X)	uva:isNormallyOpen	"true/false"
Flow Topology	PipeSegment_(RoomX)	brick:feeds	PipeSegment_(AdjacentRoom)

the data as triples (subject, predicate, object), producing a connected graph that can be queried [12]. With the Web Ontology Language (OWL), we specify the classes, subclasses, individuals, and custom properties (*uva:hasDomain, uva:hasNominalDiameter, uva:isNormallyOpen, and uva:PipeSegment*), introducing reasoning and semantic metadata to the RDF data [7]. We write both the base Brick ontology and the UVA extension in Turtle (.ttl) format, which allows us to use the SPARQL Protocol and RDF Query Language (SPARQL) to select the correct set of upstream or downstream valves for a given component [10].

In particular, the UVA-specific schema introduces water-domain entities to distinguish domestic and emergency water lines, valve metadata, and pipe segments as first-class entities representing each meaningful section of pipe within a room, as shown in Tbale 1. For this project, we create one instance file to outline the emergency sprinkler network in the Link Lab Arena, augmented with additional instance files for synthetic evaluation scenarios. These instance files declare the rooms, valves, fixtures, pipe segments, and flow topology. Pipe segments are introduced as entities in this project for cases where leaks do not occur precisely at valves or fixtures. As we know, leaks can happen at any point along a pipe run. By modeling short pipe segments, distinct in each room, as potential leak points, the system can reason about which upstream valves affect that segment, rather than treating the entire room as a single undifferentiated node. We also account for such scenarios because relying solely on the closest upstream valve for the room may prove insufficient when multiple pipes pass through the room or when multiple fixtures or valves are in proximity to it.

Though it might seem unintuitive to model each branch of pipe as a sequence of pipe segments instead of a single pipe object associated with a list of rooms, we wanted the ability to correctly identify which upstream valve might control any leak point in a room, which can vary at different points for pipe branches that run across several rooms. In its current form, the system cannot provide sufficient information to identify the source of a leak in a room definitively. Still, the structure of the Brick schema allows us to provide information specific to each candidate source in a given room, and each component in a room is treated as equally likely.

Central to the system is the Python reasoning engine, which first loads the TTL layers (Brick ontology, the small extension, and the instance file) into an RDF graph that enables identification of the upstream/downstream, domain-specific subsets. We then convert this RDF graph into a NetworkX directed graph, which stores node attributes and can be efficiently traversed and used for path-based computations. In this graph, anything that participates in a brick:feeds relationship is a node, including fixtures, valves, and pipe segments.

Directed edges are added for each brick:feeds triple to represent the direction of water flow. We specify how to traverse the resulting graph to provide three key functions: locating the upstream isolation for a leak in a room, the upstream isolation for a specific pipe segment, and the downstream impacts of closing a valve. If a leak is reported in a room, SPARQL can be used on the RDF graph to find all the endpoints in that room, and then a reverse breadth-first search (BFS) is performed from those points to determine how many hops away each upstream valve is. The candidate valves are ranked by hop count, and the valve with the fewest hops from any endpoint in the room is provided as the area-wide isolation/shut-off recommendation. For more accurate recommendations when the actual leak point is known or can be inferred, the system also returns the valves that isolate each pipe segment/sprinkler/fixture head in the room.

For downstream impact analysis, when specific valves are proposed to be shut off, SPARQL is used to identify all downstream nodes of those valves, and BFS is used to determine the hop distances to all downstream valves. Here, the system also produces a room-level summary of which rooms and fixtures lose water flow and how many components in each room are affected. The system can interpret simple, plain-English queries made through the user interface, matching them to one of the two intents we account for (valve identification for shut-off and downstream impacts). This is through a basic rule-based parser that looks for leak or impact words and extracts the relevant information, such as the room or valve identifier. Therefore, the current approach does not support complex queries, but an LLM could replace it in the future.

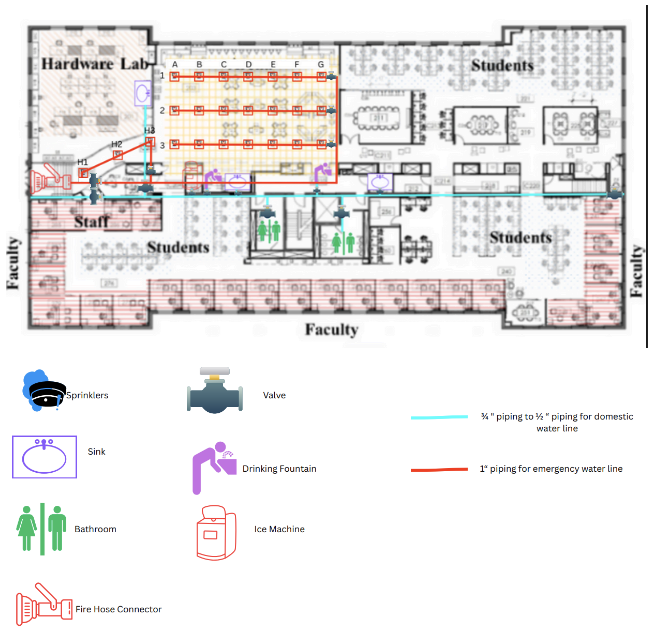


Figure 2: Layout of Valves/Fixtures of Link Lab

5.3 Link Lab Arena

The layout of the Link Lab is broken into two sections: Domestic Water Line and Emergency Water Line as shown in Figure 2. For the project, the team chose to use the emergency water line to create the Brick schema.

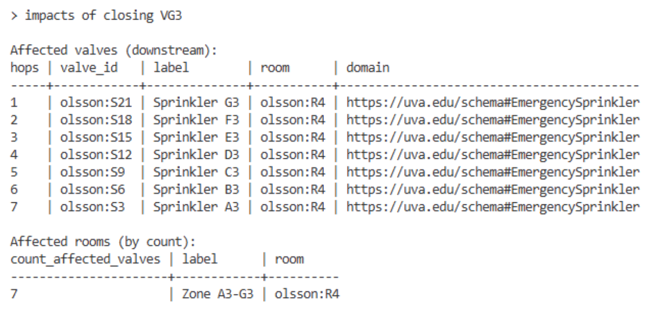
The emergency sprinkler system layout is in the Arena of the Link Lab. The system features 24 sprinklers, each equipped with a shutoff valve at the end of Sprinkler G1, G2, and G3 which are runs that branch off from the main feed and feed sprinklers 1A-1G, 2A-2G and 3A-3G. A separate shutoff valve is shown on the H1-H3 section where a single run branches off from the main feed to feed sprinklers H1-H3. Shutoff valves are placed at the ends of pipes to control the flow of water to specific sprinklers, in this case, three rows of sprinklers. This allows for localized leaks without shutting off the water to the entire building. The piping system is consistent throughout the system with a 1" diameter pipe.

Test cases for the Link Lab were executed through command line queries. For additional evaluation cases, separate .ttl files were created to encode each water-network scenario/topology. A dedicated Python script was included to loads those test graphs independently and execute our system's logic; this was so that we could easily test those cases without relying on manual input.

6 Results

Multiple test cases were executed successfully, validating the semantics of the Brick schema for the affected Arena and its leak scenarios.

To evaluate the semantic integrity and logic of the Brick-based sprinkler system, we simulated five fault scenarios. These tests confirmed the schema's ability to trace water flow paths, identify

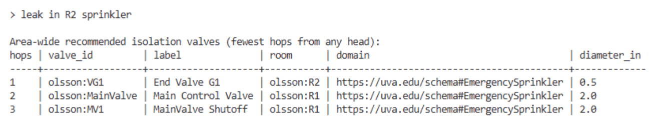


```
> impacts of closing VG3

Affected valves (downstream):
hops | valve_id   | label       | room      | domain
-----+------------+-------------+-----------+------------------------------------------
1    | olsson:S21 | Sprinkler G3 | olsson:R4 | https://uva.edu/schema#EmergencySprinkler
2    | olsson:S18 | Sprinkler F3 | olsson:R4 | https://uva.edu/schema#EmergencySprinkler
3    | olsson:S15 | Sprinkler E3 | olsson:R4 | https://uva.edu/schema#EmergencySprinkler
4    | olsson:S12 | Sprinkler D3 | olsson:R4 | https://uva.edu/schema#EmergencySprinkler
5    | olsson:S9  | Sprinkler C3 | olsson:R4 | https://uva.edu/schema#EmergencySprinkler
6    | olsson:S6  | Sprinkler B3 | olsson:R4 | https://uva.edu/schema#EmergencySprinkler
7    | olsson:S3  | Sprinkler A3 | olsson:R4 | https://uva.edu/schema#EmergencySprinkler

Affected rooms (by count):
count_affected_valves | label      | room
----------------------+------------+----------
7                     | Zone A3-G3 | olsson:R4
```

Figure 3: Room Impacted Results



```
> leak in R2 sprinkler

Area-wide recommended isolation valves (fewest hops from any head):
hops | valve_id         | label              | room      | domain                                    | diameter_in
-----+------------------+--------------------+-----------+-------------------------------------------+------------
1    | olsson:VG1       | End Valve G1       | olsson:R2 | https://uva.edu/schema#EmergencySprinkler | 0.5
2    | olsson:MainValve | Main Control Valve | olsson:R1 | https://uva.edu/schema#EmergencySprinkler | 2.0
3    | olsson:MV1       | MainValve Shutoff  | olsson:R1 | https://uva.edu/schema#EmergencySprinkler | 2.0
```

Figure 4: Leak Results

affected pipe segments, and identify the valves to close during a leak.

6.1 First Scenario

The first scenario, simulated closing of valve 'olsson:VG3' (the end valve for the third row of sprinklers), triggered a downstream impact analysis shown in Figure 3. The model identified seven affected valves, located in the third row of sprinklers, from A3-G3. It was also classified under the 'EmergencySprinkler' domain. Each valve in the simulations was assigned a "hop count" from the source, thereby confirming the logic across the system. The accumulation of affected components demonstrated that Zone A3-G3 in R4 would be inoperable. This further demonstrated the efficiency of valve isolation in preventing flooding.

6.2 Second Scenario

The second scenario focused on leak containment for the R2 (first row) sprinkler zone. The model generated a prioritized list of recommended isolation valves based on proximity (fewest hops) and the operational state, as shown in Figure 4. Valves such as 'VG1' at the end of the first row of sprinklers, the 'MainValve', and 'MV1' were identified as optimal first-line shutoffs due to their minimal hop distance and normally open status. The Higher-level parent valves, like 'MainValve' and 'MV1', were also included, offering broader isolation options with system-wide control. This output demonstrated the model's capacity to balance containment efficiency with operational continuity, providing actionable guidance for emergency response or maintenance workflows.

6.3 Third Scenario

The third scenario shows two rooms with two pipes running through the ceiling in parallel, as seen in Figure 5. The pipes share a common feed that branches just before entering the first room, and one pipe passes through the space over rooms 1 and 2 before turning right to feed Fixture A. The other pipe runs parallel to the

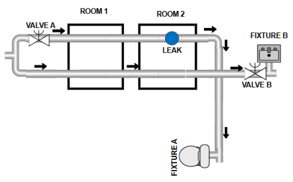


Figure 5: Test Case 1

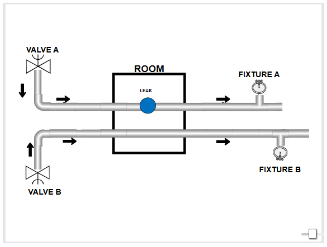


Figure 6: Test Case 2

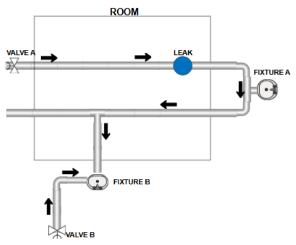


Figure 7: Test Case 3

pipe feeding Fixture A, passes through the space over Rooms 1 and 2, and continues further to feed Fixture B. Both pipes have their own isolation valves. In this configuration, the Brick model would represent two ceiling pipe segments passing over the rooms: one belonging to the one feeding Fixture A to the right and one belonging to the one feeding Fixture B further down from the rooms. The engine's job is to show that a leak on the pipe segment feeding Fixture A should be isolated by closing the isolation valve. A room-level leak query would show both pipes passing through the space, but the branch-specific queries seeded from each pipe segment demonstrate that the system can distinguish between "same-looking" ceiling pipes that have very different upstream consequences.

6.4 Fourth Scenario

The fourth scenario shows two straight, parallel pipelines within a single room, as seen in Figure 6. The pipes enter the room from different directions but pass through the same space over the room. One pipe leaves the room and feeds Fixture A, while the other also leaves the room and feeds Fixture B. Each pipe has its own isolation valve. In this configuration, the Brick model would represent two ceiling pipe segments passing over the rooms: one belonging to the one feeding Fixture A to the right and one belonging to the one feeding Fixture B further down from the rooms. The engine's job is to show that a leak on the pipe segment feeding Fixture A should be isolated by closing the isolation valve.

6.5 Fifth Scenario

In the fifth scenario, there is a single room, and a looped pipe passes through the ceiling twice: the first ceiling run enters the room, passes over the space, and feeds Fixture A before continuing out of the room, as seen in Figure 7. Somewhere further along the building, that same loop returns and re-enters the room at the ceiling a second time, now feeding Fixture B. Importantly, Fixture B is not only supplied by this returning loop; it is also fed by a second, independent pipe that comes into the room from another direction with its own isolation valve. From the operator's perspective, both visible pipes near Fixture B are on the ceiling and both appear to "touch" the same fixture. In this configuration, the Brick model would represent two ceiling pipe segments in the room: one in the main loop (which has already passed through once and returns), and one in the secondary feed. The engine's job is to show that a leak on the loop segment feeding Fixture A and then continuing to Fixture B should be isolated by closing the main loop valve, while a leak on the independent feed into Fixture B should instead be isolated by the local secondary valve. A room-level leak query would still list both options for Fixture B, but the branch-specific queries seeded from each pipe segment demonstrate that the system can distinguish between "same-looking" ceiling pipes that have very different upstream consequences.

7 Discussion

The test cases demonstrate the feasibility and effectiveness of the proposed system using the Brick schema for water distribution systems within buildings. The first scenario, simulating the closure of a specific valve, demonstrated that the system accurately identified all downstream assets (sprinklers) affected. The second scenario, simulating a leak in a room, demonstrated the system's ability to identify all upstream valves, starting with the valve closest to the leak and propagating to the others if the first failed. The third scenario, with Valve A closed, correctly isolates only the upper straight run within the room. The fourth scenario shows that Valve A serves as the upstream shutoff for the upper supply line feeding Fixture A. The leak is located downstream of Valve A on the same branch, as illustrated in Figure 7. When Valve A is closed, the Brick-modeled hydraulic graph correctly isolates all downstream components along the upper pipeline while leaving the lower branch and Fixture B unaffected. Closing Valve A isolates the entire top branch in both rooms while leaving the bottom branch unaffected. In the fifth scenario, the impact analysis indicates that Fixture A and all intermediate pipe segments between Valve A and the leak point experience a complete flow loss. Because the leak resides on this same branch, the closure effectively removes pressure at the failure site, confirming the model's ability to trace a linear single-branch spread. The lower supply line independent of the flow through Valve B remains fully operational, and Fixture B continues to function without interruption. This targeted isolation highlights another core feature of the system design, where independent branch closures mitigate leaks without triggering building-wide equipment shutdowns. The result demonstrates that Brick's graph representation can distinguish between parallel supply paths, enabling fault containment and operational continuity during maintenance or emergency response. Therefore, these results validate the logical and semantic relationships defined in the model, and that the knowledge graph and query logic that we implemented are both correct and useful starting points for leak response.

These results and capabilities have significant implications for building managers and water infrastructure management. This system provides a foundation for informed decision-making during maintenance and emergencies. During planned maintenance or a water line leak, building managers can use this system to minimize service disruptions and identify critical components affected by the intervention.

7.1 Limitations

While the test cases confirm the system's functionality, the proposed approach has several limitations. The test cases were limited to single, small-scale examples for each capability and, therefore, do not demonstrate scalability for large or complex networks. Furthermore, the current system is limited, which restricts its usefulness. Primarily, the user interface is relatively basic, making it challenging to navigate to significant insights. In addition, the Brick schema does not integrate the ever-changing data sources of a building, such as water meters, which could significantly enhance the system's ability to detect and diagnose issues. This paper's evaluation also did not extend to scenarios involving domestic water lines, leaving important cases untested. Domestic fixtures far outweigh emergency fixtures in overall usage. The likelihood of a domestic water leak occurring on a building floor is high. Most importantly, the system lacks functionality to identify the actual causes of leaks, limiting its practical value for operators.

7.2 Future Work

Future work will focus on expanding the system to handle larger datasets, integrating real sensor data to update the model, and improving the query efficiency for real-time applications. Additionally, improvements should include enhancing the Brick model with additional spatial and semantic details. Instead of assigning pipes only to brick:Room spaces, specific descriptors could be used, such as "Room 2:north ceiling" or "Room 2:floor chase." Moreover, pipe-specific details, such as brick:hasLocation, can be attached to each pipe segment. Custom-created properties, like uva:hasOrientation (e.g., ceiling, floor, north wall), uva:hasElevation, or uva:branchId (e.g., TopBranch, BottomBranch)—would further distinguish overlapping pipes that share the same room

8 Conclusion

This work demonstrates that the Brick schema, when extended with domain-specific metadata and paired with a reasoning engine, can serve as a practical foundation for modeling, analyzing, and managing water infrastructure within buildings. By applying this approach to the emergency sprinkler network in the UVA Link Lab, we show that a semantic representation of valves, pipe segments, fixtures, and flow relationships enables accurate tracing of both upstream isolation paths and downstream impacts during leak or valve-closure events. The successful execution of the two test scenarios confirms the viability of using Brick to support real-time decision-making during emergencies, reducing dependence on manual searches, incomplete documentation, or specialized staff knowledge.

Beyond validating Brick's representational accuracy, this project highlights its practical value for facility management workflows. The model's ability to identify optimal valves for leak isolation and quantify room-level service outages underscores its potential to enhance emergency response, maintenance planning, and overall building resilience. By integrating semantic metadata with automated reasoning, the system transforms static building information into actionable insights, laying the foundation for more scalable, intelligent infrastructure management tools.

References

[1] Bharathan Balaji et al. 2016. Brick: towards a unified metadata schema for buildings. In *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, 41–50.

[2] [n. d.] BrickSchema/brick: uniform metadata schema for buildings. Retrieved Oct. 29, 2025 from https://github.com/BrickSchema/Brick.

[3] Ruijia Chen, Junru Jiang, Pragati Maheshwary, Brianna R Cochran, and Yuhang Zhao. 2025. Visimark: characterizing and augmenting landmarks for people with low vision in augmented reality to support indoor navigation. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, 1–20.

[4] Tor Åsmund Evjen, Seyed Reza Hosseini Raviz, Sobah Abbas Petersen, and John Krogstie. 2020. Smart facility management: future healthcare organization through indoor positioning systems in the light of enterprise bim. *Smart Cities*, 3, 3, 793–805.

[5] M Gheisari, G Williams, BN Walker, and J Irizarry. 2014. Locating building components in a facility using augmented reality vs. paper-based methods: a user-centered experimental comparison. In *Computing in Civil and Building Engineering (2014)*, 850–857.

[6] Burak Gunay and Weiming Shen. 2017. Connected and distributed sensing in buildings: improving operation and maintenance. *IEEE Systems, Man, and Cybernetics Magazine*, 3, 4, 27–34.

[7] [n. d.] Introducing RDFS & OWL – LinkedDataTools.com. Retrieved Oct. 29, 2025 from https://linkeddatatools.com/introducing-rdfs-owl/.

[8] Hussein Marza. 2024. A review of indoor positioning techniques. *Al-Farabi for Engineering Sciences*, 1, 2, 7.

[9] Supattra Puttinaovarat, Suwat Jutapruet, Aekarat Saeliw, Siwipa Pruitikanee, Jinda Kongcharoen, Watchara Jiamsawat, and Suchakree Limpasamanon. 2019. Facility maintenance management system based on gis and indoor map. *International Journal of Electrical and Computer Engineering*, 9, 4, 3323.

[10] [n. d.] Querying semantic data – LinkedDataTools.com. Retrieved Oct. 29, 2025 from https://linkeddatatools.com/querying-semantic-data/.

[11] CP Rahul Raj, SeshuBabu Tolety, and Catherine Immaculate. 2013. Qr code based navigation system for closed building using smart phones. In *2013 International Mutli-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s)*. IEEE, 641–644.

[12] [n. d.] Resource description framework (RDF): concepts and abstract syntax. Retrieved Oct. 29, 2025 from https://www.w3.org/TR/2004/REC-rdf-concepts-20040210/#section-Concepts.

[13] Zeev Volkovich, Elena V Ravve, and Renata Avros. 2024. Indoor navigation in facilities with repetitive structures. *Sensors*, 24, 9, 2876.

[14] Zhenyong Zhang, Shibo He, Yuanchao Shu, and Zhiguo Shi. 2019. A self-evolving wifi-based indoor navigation system using smartphones. *IEEE Transactions on Mobile Computing*, 19, 8, 1760–1774.