

DATA SCHEMAS

Building Blocks of System Automation

OpSys Solutions & Simfeny Consulting

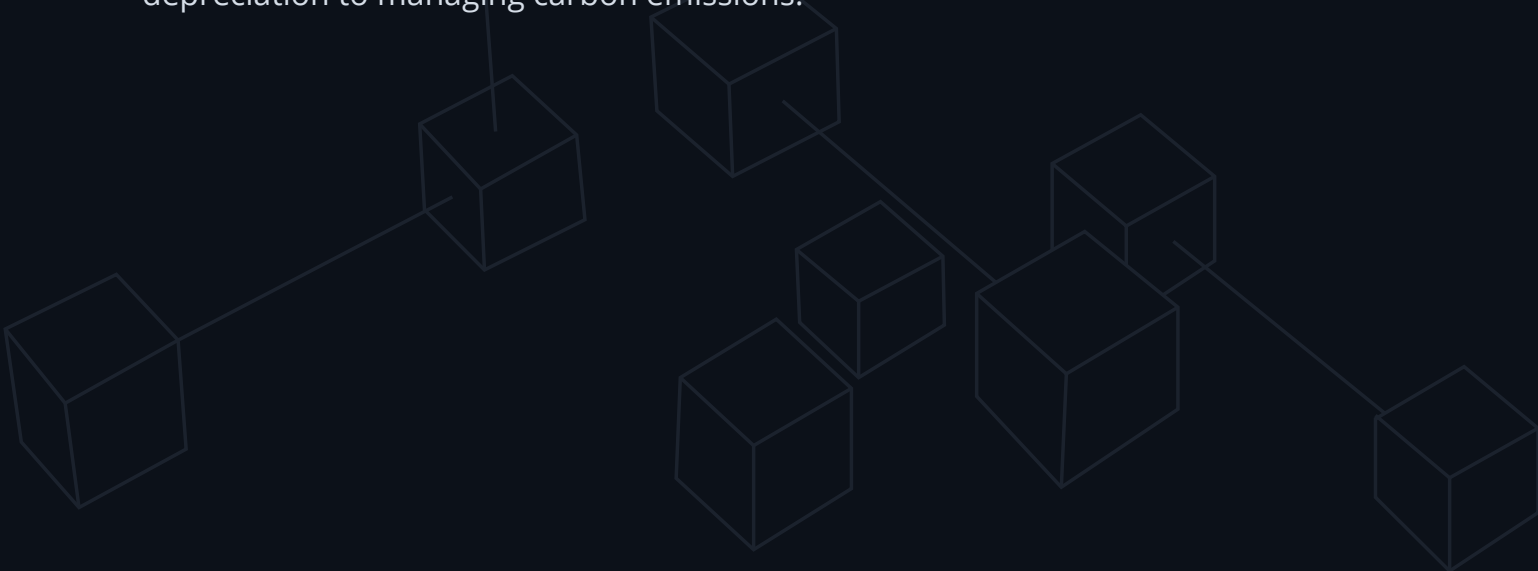


Introduction

It's difficult to make big data work for you without first putting it into a standard structure. Schemas are that structure: the foundational building blocks of your digital architecture.

By building your asset database with standard schemas, you're ensuring your building, tenants and occupants benefit from future innovations such as advanced analytics, AI, machine learning, and cloud computing. These are the future of building operations and facilities management.

Once all buildings graduate to smart status, they'll be connected to everything, and proptech will help managers do everything from calculating asset depreciation to managing carbon emissions.



Data is a beautiful thing, until you're drowning in it. FMs and building owners will likely soon experience that sentiment firsthand, as many flounder in a sea of building data from IoT devices, AI and machine learning, cloud computing, and new “proptech” ventures. All of these collect, store, and analyse petabytes of information, much of which may be wasted on an industry too ill-prepared to receive it.

Over the next five to ten years, we're likely to hear a chorus of desperate voices sounding the cry: “Data, data everywhere and not a byte to use!”

Some will have the capital, but not the will. Others will have the will but not the capital. For too many, the inescapable “solution” to this data inundation will be to simply slow down adoption of tech. However, this only kicks the can down the road and forecloses on the benefits analytics brings to building performance and value.

These growing pains can be avoided altogether if property owners see preparation as the investment, and not tech itself. As a wise person once said, “Give me six hours to chop down a tree, and I'll spend the first four sharpening the axe.”



But how do you prepare for building tech? What is the “axe” in this scenario? For most, it will be adopting an interoperability schema standard. Schemas make recording and managing your asset database easier by ensuring your asset library is mapped, tagged and organised in a way that's easily understood by machines and software. So, these standards are intended for both building owners and developers, ensuring both parties are speaking the same language.



Today's most popular standard schemas differ in their approach, but all attempt to standardise the way assets are described and stored to aid interoperability and software deployment. Project Haystack is a tag-based schema focused on streamlining operation between smart devices within buildings, homes, factories, and cities. The Brick Ontology standardises both asset labels and connections, allowing the user to create a relational database, and has been harmonised with RealEstateCore, a schema widely used by property owners. A third development to watch is ASHRAE's proposed Standard 223P, which will bring semantic building data under a formal industry standard and is designed to work alongside Haystack and Brick rather than replace them. But to fully understand the benefits schemas bring to building systems, it's critical to know exactly how they work.

what's a database?

Let's use a book as an analogy to answer this question. Books must have structure, order, logic, words, sentences, paragraphs, punctuation and the like. These structures and rules are what make writing and reading possible. Imagine an entire book with no paragraphs, chapters, or punctuation. Could readers "get" the story? Yes, but it would be inefficient and hard-going work. In the absence of structure and standards, information becomes disparate chunks of semi-familiar things with little context.

A database works in a similar fashion, in that it also must have structure and rules for reading and writing data. Your building database holds the information your building assets need to communicate, and that data must follow rules. Interoperability standards are those rules for data management. One critical type of data is metadata.

metadata

Continuing our book analogy: metadata is any information about a book, such as its title, author, and publisher. In other words, metadata is “data about data”. It says something about the data, something that's often just as important to readers as the book's contents. Without metadata, it becomes difficult to locate, store, discuss, or appreciate a book's full meaning. For example, without the call number, how is a library patron supposed to locate a specific book?

Asset metadata is similar in that it says something about the asset, such as its location, size, type, class, and relationship to other assets. All this metadata is critical to efficient storage and retrieval of all those billions of bits. Therefore, how you go about handling metadata has a huge impact on what you can do with it.



What's a Schema?

A schema is a format for creating, recording, storing and retrieving metadata. Schemas can be a way of saying “x is a type of thing” or “x is related to y”. Some schemas are basic, recording only a few pieces of metadata (e.g., the call number and title of a book). Others are complex, recording multiple pieces (e.g., call number, title, author, publisher, publication date, genre).

schema

The more complex your schema, the more descriptive it is, just as a long sentence is more descriptive than a short one. Consider the following two sentences:

1. The dog fetched.
2. The black Labrador fetched the yellow tennis ball from its toy box.

What can we do with the second sentence that we can't do with the first? Sentence 2 gives us more descriptive words ("black Labrador", "yellow", "toy box"), so we have a better understanding of the context. It also supplies the object: we know not just that the dog fetched, but what it fetched.

Most importantly, Sentence 2 creates relationships. It describes how one thing (the dog) is related to another (the ball), which is related to a third (the toy box). We can therefore say the longer sentence is "relational".



These same differences exist between informal and standardised schemas. Longer, more descriptive schemas provide more context and meaning around things such as a building asset. They're also relational, in that they describe how one asset (e.g., a temperature sensor) is related to another (e.g., an AHU). Consider these two naming schemas for a temperature sensor housed on Level 9 of a hospital:

#1 LV09_TempS

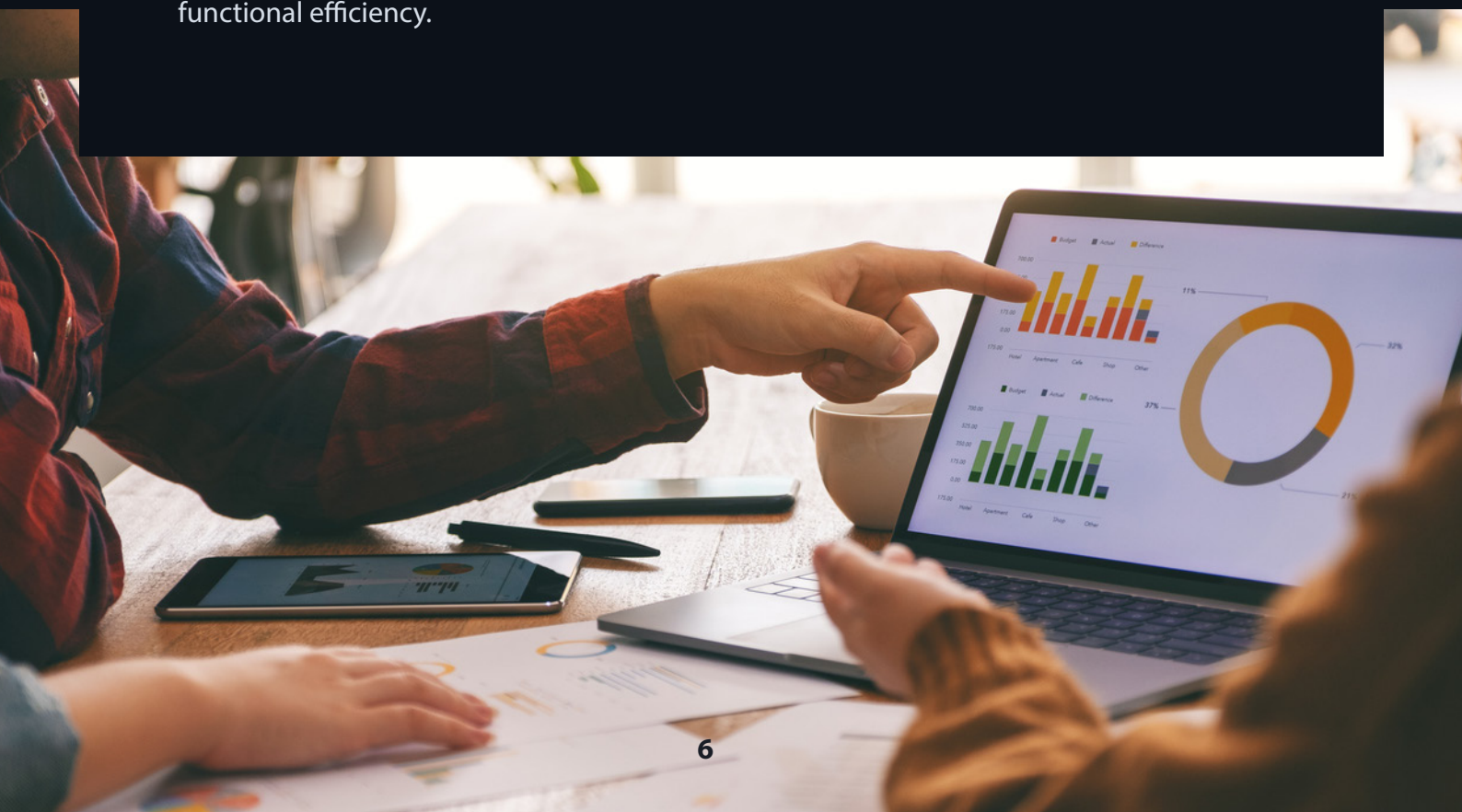
#2 A32_M_AHU_L10L82_RMT_L9OR2_TS

Building System Asset Type Asset Name Point Type Supplies Location /Asset Device

While Schema 1 lists only the location (LV09) and device name (TempS), Schema 2 extends the description to include the building, system, asset type, point type, specific location, and the device class.

With these added details, we now have a relational description of the sensor. For example, we know it is part of the mechanical (M) system and part of an AHU. Therefore, we can say Schema 2 is part of a relational database, and that it gives us a greater understanding of the asset and its place in the system.

Overall, Schema 2 gives us more context and meaning than Schema 1, and we can use this information to learn more about how our buildings operate. Once we extend this schema strategy to our entire building, we have a powerful way to analyse its contents and functional efficiency.



standard schemas & relational databases

As we've seen, standard schemas provide more detailed descriptions of building equipment, software, and processes than simple ones. The upside of this enhanced description is system interoperability. Within relational databases, an "asset" is no longer just a single piece of equipment. It is a well-defined instance of a component within a much larger and more complex system. An asset logically includes everything it contains and everything that contains it. Integration creates positive benefits for your building management.

Software Deployment

Standard schemas create a common lexicon and database structure for software developers to use. Again, think of the book analogy. Reading and writing a book both require adherence to protocols and structures. Writers understand how to use those rules to encode meaning, and readers understand how to use them to decode meaning. When an author writes "dog", the reader understands that to mean a four-legged domesticated mammal that barks. The same rules apply to asset naming standards.

With respect to metadata, it matters what you call an asset. Too often, database naming conventions are inconsistent from one property to the next or even within an existing building. One system may refer to a temperature sensor as "TempSensor" while another uses "TempS".

Such inconsistencies create a database which humans can make sense of, but machines can't. Standardised schemas remove these inconsistencies.



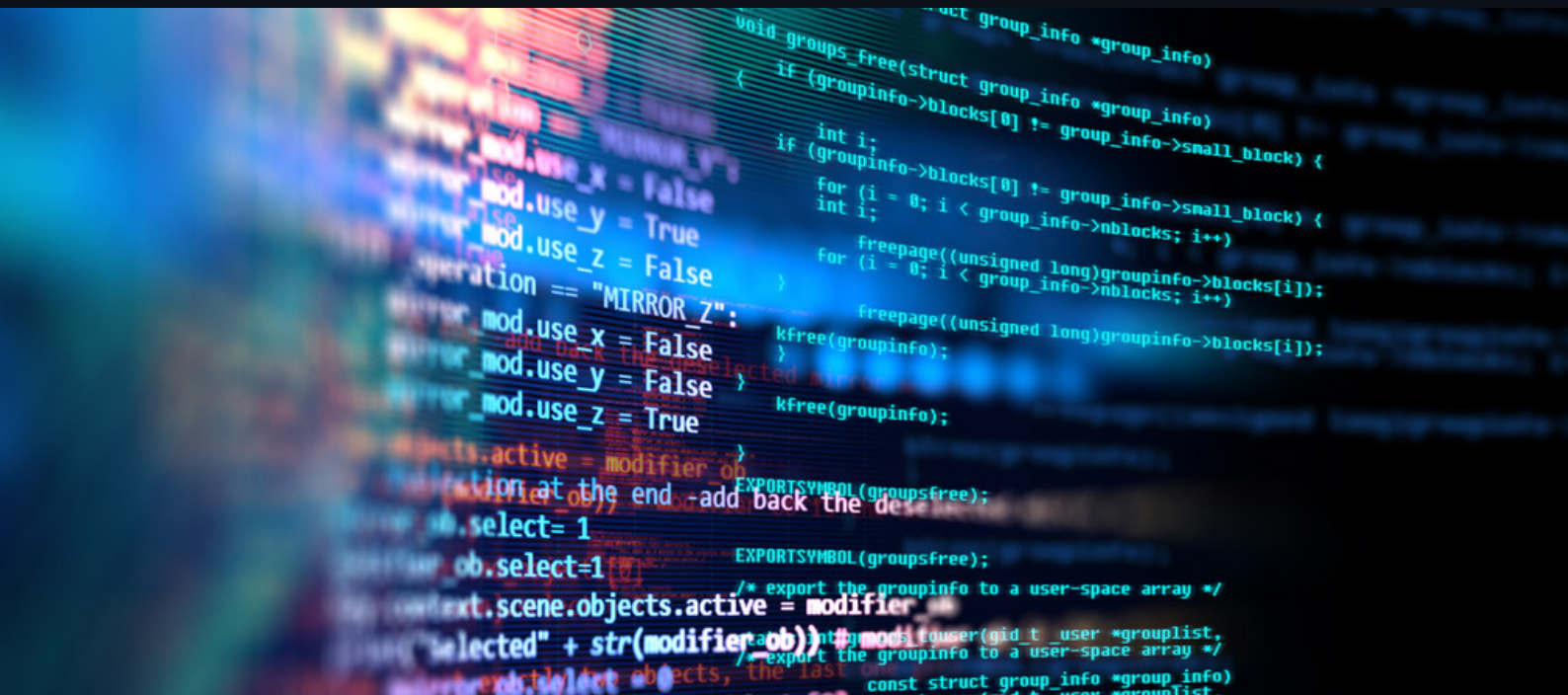
Adopting a standard naming schema makes software deployment and management much simpler. Developers and building systems benefit from a common, predictable set of rules and naming conventions. Such standards make software development and deployment easier and cheaper because both stakeholders are working from a shared data structure. The developer can simply bolt their software package to your system, and everything works out-of-the-box.

Schemas also aid software deployment by codifying asset relationships. Schemas define relationships like: "X" AHU contains "X" VAV, which contains "X" damper, which contains "X" thermostat, which has "X" setpoint. Without this relational data, the software can't "read" these connections or understand what an asset is, because it can't relate it to its constituent components. (The word "dog" has little practical meaning by itself, but in relationship to other words gains more signifying potential.)

Naming conventions and asset relationships are often created and discovered manually for building systems; however, it's a tedious and costly process. Relational schemas create a predictable, standardised database that allows software to easily read the entire system architecture at a glance.

“

Relational schemas create a predictable, standardised database that allows software to easily read the entire system architecture at-a-glance.

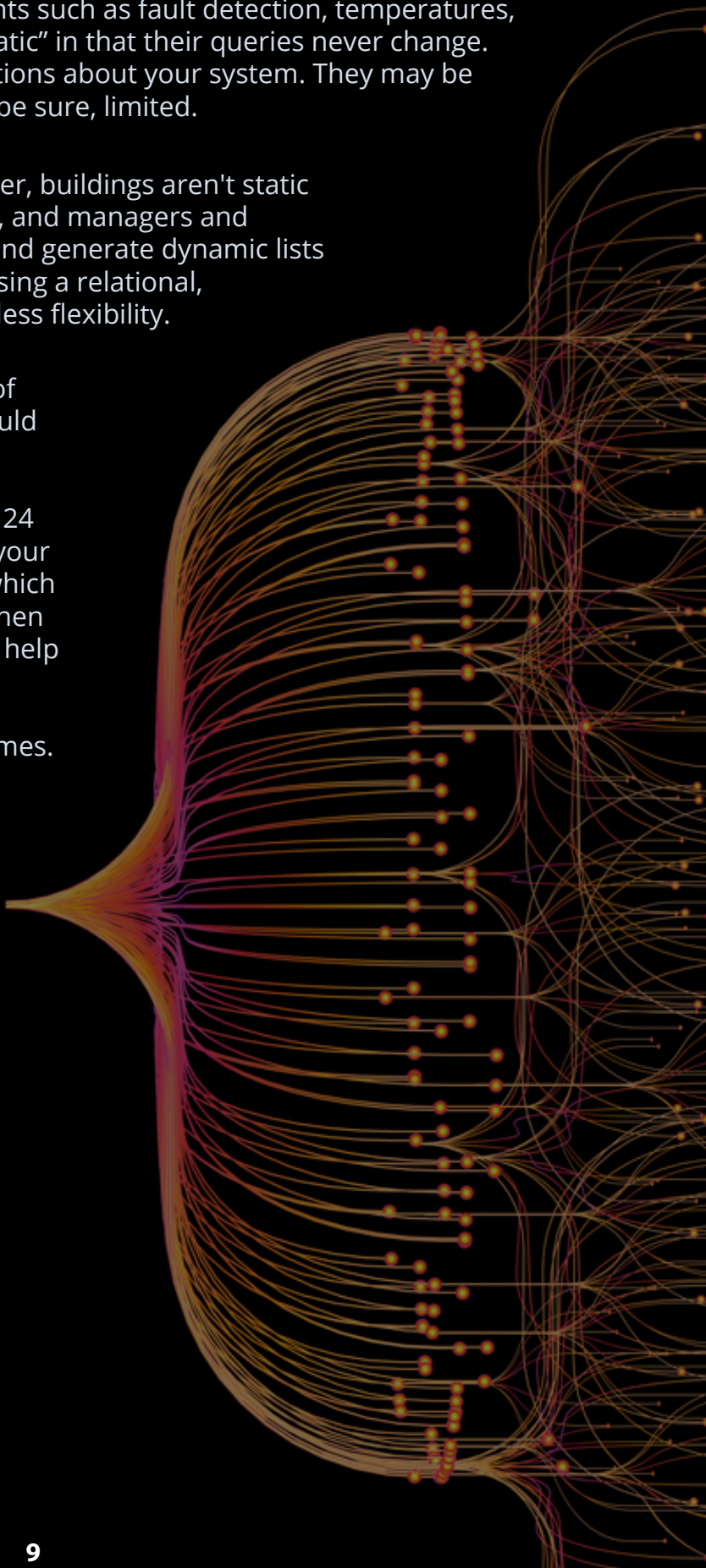


Advanced Queries & Dynamic Lists

Conventional BMS pages are static. Their queries are hard-baked, with pre-built graphics that deliver data around points such as fault detection, temperatures, run speeds and statuses. They are “static” in that their queries never change. Your BMS will only “ask” specific questions about your system. They may be important questions, but they are, to be sure, limited.

Contrary to their appearances, however, buildings aren't static with respect to the data they produce, and managers and engineers often need to run queries and generate dynamic lists that exist outside the BMS purview. Using a relational, standardised schema allows this limitless flexibility.

For example, say you suspected one of your AHUs was starting to fail. You could run a query that identified all room temperature sensors that have been reading above 21 degrees for the last 24 hours for that specific AHU. Because your schema is relational, it understands which specific sensors to target. You could then upload the data to a dynamic page to help troubleshoot performance issues. Dynamic lists like these can improve predictive failure and shorten downtimes.



Asset Replacement

With a standard relational schema, you can identify an asset's effect on the wider system and its impact on service. For example, a standard schema can show you how other systems are affected when you plan to replace a failed actuator. Before work begins, you can ask questions like: “Will replacing the actuator stop chilled water to the whole building, or just the data centre?” or “How will the replacement affect Tenants X, Y and Z?” Such insights give you and your service engineers the right information for estimating costs, cutting downtime, and ensuring better tenant outcomes.



Updating Building Data

Buildings go through many evolutions in their life cycle, and these changes affect your asset database. Some changes involve assets. For example, most fit-outs involve installing or relocating HVAC and electrical equipment. Other changes are conceptual, such as renaming room numbers or floor levels. Because of cost and time commitment, these changes are seldom updated in the database. Service providers and managers often neglect revising metadata for their BMS, floor plans or switchboards. Instead, remembering these changes is left to individual team members, who inevitably move on to other properties or on to retirement.

Eventually, chaos creeps into your building systems and databases. Suddenly, historical data like equipment names and room numbers no longer reflect mechanical drawings. Simple replacements of VAVs or actuators become complex and time-consuming ventures, with engineers forced to track down which sensors are connected to what assets. Switchboards with mismatched labels require manual shutdowns of areas to re-map electrical circuits.

Standard relational schemas make updating metadata much easier and more accurate. Recording changes only requires updating one specific piece of data, like a room number or new part. After that, your system automatically adjusts names and relationships, both upstream and downstream. Standard schemas cut the time and costs of updating asset databases.



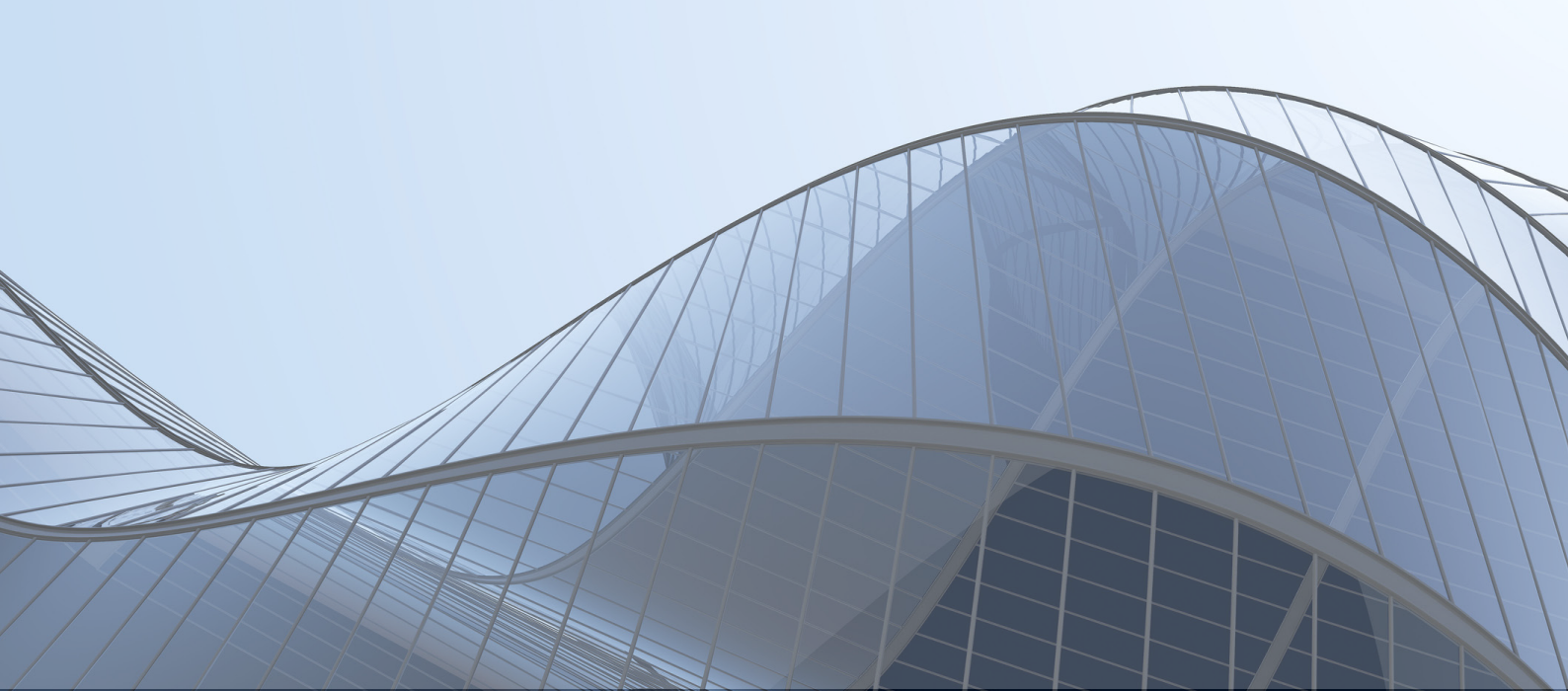
Conclusion

“Sharpen the Axe”

Standard schemas aren't glamorous, but they're the groundwork everything else stands on. A building with a clean, relational asset database can adopt analytics, AI and new proptech services quickly and cheaply. A building without one pays for that gap on every project, every integration, and every service call.

The good news is that preparation is a choice. Whether you're planning a new development or bringing order to an existing portfolio, adopting a standard schema today sharpens the axe for every technology decision that follows. Take after-hours HVAC scheduling as one example. 7NOX layers over your existing BMS to schedule, meter and report after-hours air conditioning, and it deploys fastest in buildings whose assets are already mapped and consistently named. The better your schema, the faster services like this deliver value. Well-organised data pays for itself the first time you plug something new into it.

Ready to put your building data to work? Visit 7nox.com to see what a well-connected building can do.



7NOX is an after-hours HVAC scheduling and billing platform from Opsys Solutions. It gives commercial buildings named-user accountability, smart scheduling and exportable energy reports, layered over the existing BMS. Learn more at 7nox.com.

info@opsyssolutions.com



Simfeny is a building services and technology solutions consultancy. We specialise in creating, distributing and maintaining building system schemas that give our client's the solid foundations for developing their property's full potential.

+64 21 137 4264
dominic.lauten@simfeny.com

