2019 EC³
2019 European Conference on Computing in Construction
Chania, Crete, Greece
July 10-12, 2019

DYNAMIC DISTRICT INFORMATION SERVER: ON THE USE OF W3C LINKED DATA STANDARDS TO UNIFY CONSTRUCTION DATA

Cathal Hoare, Usman Ali, and James O'Donnell
School of Mechanical and Materials Engineering and UCD Energy Institute
UCD, Dublin, Ireland

Abstract

The evolution of ICT and BIM systems in the construction domain yield detailed views of buildings and their use throughout their lifespan. These systems also provide a structure around which information about buildings and their effect on surrounding infrastructure can be described in space and time. Thus, when aggregated, information provided by these systems can serve as a semantic structure through which other information can be stored and contextualized. While bespoke systems have explored these approaches in particular contexts, few if any systems have been constructed to provide a flexible, semantically rich structure that can be used to structure information about any urban landscape at district and regional scales. This paper describes such a system. The Dynamic District Information Server (DDIS) provides a core information structure which can be extended to store as yet undefined information structures and allow these to be reasoned about in the contexts that range from neighbourhoods to regions. In addition, the paper describes how the DDIS can serve as a coordinating process in a tool chain by providing a semantically rich and flexible notification system that allows tools in the chain to notify one another when steps in some information process have been completed.

Introduction

Technological advances are improving how we build, operate and renovate buildings. Advances in data measurement, collection and management are providing a rich picture of how buildings operate and perform. This information can be used by stakeholders such as owners and managers to improve a building's performance, and by planners and decision makers to plan future buildings and urban spaces. Such advances have never been more important as society begins to address the twin challenges of climate change and access to quality accommodation for ever denser population centres; both goals must be met within the strictures of emissions standards - both during construction and operation - while providing comfortable, affordable accommodation within an attractive, efficient urban environment. Other advances, such as cloud computing, and access to reliable, robust software packages are making the use of such data for the development of decision support systems possible for all stakeholders.

While the use of building and environmental data is replete with benefits, challenges remain (Azhar et al. 2019). Cost of development, regulatory issues such as data protection, trust worthiness of results and technical challenges pose significant hurdles[1]. While custom information management systems are accessible to large firms and operators of buildings, the cost of development, operation and maintenance of such systems are prohibitive and unjustifiable to many stakeholders. In light of public reaction to social media companies apparent misuse of individuals' data, awareness around privacy and data access is growing, and may cause individual building owners and organizations to be reluctant to share their information (Department of Communications & Resource 2014). Furthermore, government and EU regulation has defined significant punishments for misuse of users' data under General Data Protection Regulation (GDPR)[2] and other regulations; this has placed a burden on operators of software systems that may cause reticence about storage and use of data. In addition, this information often has commercial value.

Stakeholders are also more discerning and cautious about the promises of both organizations and technologies; for instance, it has been found that uptake of grant support for building renovation in Ireland has not been forthcoming unless users can see and trust the promised outcomes of these efforts[3].

Finally, technological challenges exist. For example, over the course of time, development of building and urban data systems has diversified into task specific silos. Software systems and standards developed around these and so data exchange from within a specific silo to some other domain remains challenging; yet, such exchanges are key to proving decision support systems that are wholly informed Yoshida & Hirashita (2003).

[1] https://constructionprocurement.gov.ie/wp-content/uploads/BIM-Adoption-Strategy-Statement-of-Intent.pdf
[2] https://eugdpr.org
[3] Introduction Part iii https://www.dccae.gov.ie/documents/2017041220LTRS.pdf

This paper presents and details a novel data management solution called the Dynamic District Information Server (DDIS) that uses W3C Linked Data standards and conventions to:

- Bridge various complimentary, yet diverse data standards (for example, CityGML and BIM IFC);
- Provide both data and process management for relevant tool chains;
- Preserve data ownership and privacy, and provide a mechanism to explain results in order to build users' trust in decision support outputs;
- Be affordable, both in terms of operation and expertise required to run the system. Furthermore the system seeks to be flexible, accommodating new data sources and tool chains with minimal refactoring.

It is anticipated that this system could serve as the nucleus to a data management and exchange system that informs decision support in building and urban planning environments.

Throughout this paper, a scenario will be used to illustrate how the server meets the goals described above. In this, a semantic bridge will be built between district data expressed CityGML and building data in IFC format. This information resource will then be augmented with data from other sources. The scenario will be described in detail later, and should be considered as just one application of the server. While this paper focuses on the use of these formats because of their use in the NewTrend Project[4], preliminary work has been done to assess the feasibility of supporting additional formats such as W3C's LBD[5].

The paper continues by briefly describing relevant systems and standards and introducing W3C Linked Data and related standards used in this work. The systems architecture will then be described, before concluding with an explanation of how the server was applied to manage information related to the scenario.

Background

(Boddy et al. 2007) presented an innovative visualisation of the Computer Integrated Construction (CIC) landscape that helped researchers place their systems in the context of research and industrial systems, as well as assist in the identification of axis of advance for research outputs. Their work divided the CIC landscape into four quadrants by proposing two key axis, shown in Figure 1:

- *semantic focus (from Data Structures through Semantic Description)* - which provides an axis to express the semantic expressivity of a system, standard or protocol, from generic data structures which express little context to semantic standards that permit richer opportunities for synthesis of information;
- *application focus* - which provides an axis to express the scope of the work, from low-level APIs and monolithic applications to entire tool chains and complex processes.

These axes define four quadrants (from Software Application Interworking through Process Integration) - *Integration at data-application level*, *Integration at application-semantic level*, *integration at data-process level* and *integration at process-semantic level*.

In our discussion, we merge the *data-process* and *process-semantic* levels into a single data-process-semantic level. In the authors' opinion, these quadrants are about process integration either as a functional goal of the software, and to provide process-oriented software architectures; the goal of the DDIS is to admit an event driven architecture that advances both of these goals, as well as address related issues of trust and cost of running such a system.

Information Interoperability remains a challenging topic in CIC. While individual tools have evolved to provide complex functionality, the ability to pass information artefacts between these has relied on techniques as simple as manually passing files, often in intermediary formats that involve human intervention and loss of detail. This approach also has implications for data control, and in particular, handling information that is of a sensitive nature. Files that are manually passed may be subject to unauthorised access if the transfer medium is compromised - for example, a lost memory stick, or an email account compromised through use on an unsecured network (Belanger & Crossler 2011).

Automated information interchange was previously achieved through the use of shared databases, where a common schema was used by multiple software tools, and then through the use of Application Programming Interfaces (APIs). Both of these approaches enable information sharing. However, both approaches have issues; the most significant is inflexibility - with significant refactoring (redesign and re-implementation) required to extend in order to cater for new information requirements of new applications; while standards exchanges would require changes for any approach, these changes are less common.

The CIC landscape diagram described by Boddy et al is extended in this work to include semantic contributions in each quadrant. The authors contend that this is the natural extension of progress as semantic standards provide mature mechanisms to facilitate information exchange while systems interoperability offering information organisation options that facilitate data control. Semantic standards,

[4]http://newtrend-project.eu/
[5]https://www.w3.org/community/lbd/

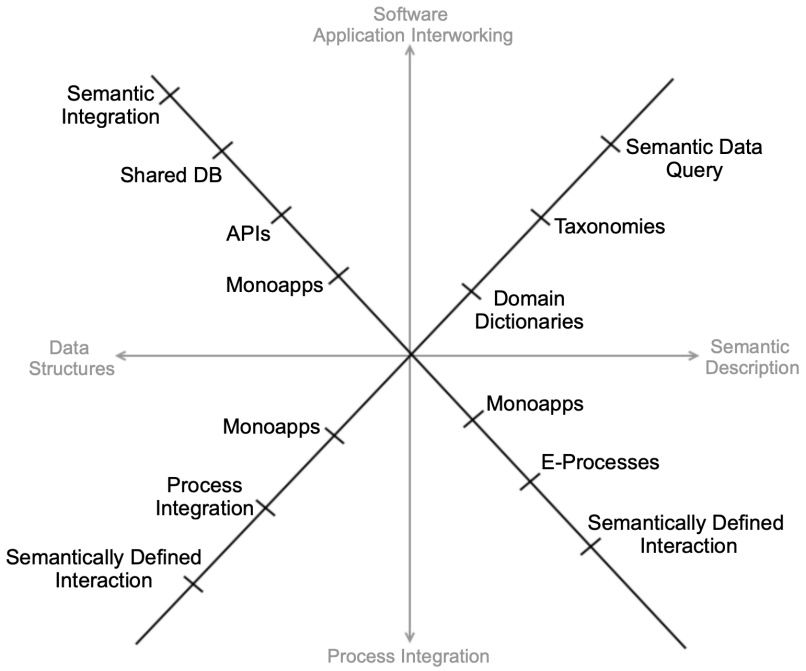

Figure 1: Boddy et al's representation of CIC landscape

for example, Linked Data, are proposed and developed by the World Wide Web Consortium (W3C). These standards seek to enrich the ability to communicate information in machine readable formats, and have these standards widely adapted to ensure compatibility access across domains.

Linked data is increasingly being exploited in CIC (as promulgated by the broad range of research presented at the LDAC[6] conference series). These standards offer both expressivity and connectivity, which are key requirements for interoperability. Several projects have been involved in using the semantic web and linked data in the building domain. The SWIMing project has examined over 100 EeB projects, analysing them in detail to identify where linked data and semantic technologies can be used to make information more accessible, exchangeable and easier to exploit (Mcglinn et al. 2016).

Linked Data approaches have also been used to address privacy issues. It was recognized that linked data would pose challenges to privacy and data control. However, the standards developed to support information discoverability have addressed many of these concerns. In particular, efforts to make information discoverable has lead to a distributed architecture where it is possible to host information on servers that are managed by the data's controller, and exposing only the required subset. Information science has also developed best practices and methodologies that strive to guide organizations in striking a balance between information openness and control (*Big and Open Linked Data (BOLD) in government: A challenge to transparency and privacy?* 2015).

W3C Linked Data Protocols

This work describes the implementation of the DDIS that manages building related information at multiple scales. To do this, it leverages implementations of several protocols designed by W3C to manage linked data. This section describes the relevant protocols and how they are used by the DDIS.

Linked Data

Linked data is a set of design principles for sharing machine-readable data on the Web (Mihindukulasooriya & Menday 2012). It is made up of named data that can be exchanged and reasoned about. Data can be expressed statically as RDF or dynamically embedded in the format. This data is named using a unique Uniform Resource Identity (URI) that allows it to be found and shared across the web. The data can contain links to other named data sets and meta data about the relationship between these linked datasets. Given a rich enough network of data, a computer algorithm can reason about entities based on the contexts provided by their related data.

While the ability to reason about distributed data is in itself sufficient motivation to embrace the concept of linked

[6]http://linkedbuildingdata.net/ldac2018/

data, the schema has further advantages. Because the data is distributed, maintenance is reduced and only expertise about data related to an organisation's expertise need be maintained. Furthermore, the data exposed need only be the subset that is relevant and permissible for an organisation's remit. Maintenance of data is distributed, while the linked data concept is infinitely extensible. While there is an overhead in re-expressing data in RDF format, the process can be largely automated and supported by mature software tools.

In this work, the server both exposes and describes related linked data to facilitate data exchange between attached software clients.

Data Cubes

W3C's RDF Data Cube Vocabulary makes it possible to publish a multi-dimensional dataset as part of a linked data schema (Cyganiak & Reynolds 2014). In this context, a dataset consist of the following data types:

- Observations or the raw data;
- Organisational Structure positions the observation in each dimension it is related to;
- Structured metadata provides metadata about he observation, such as its unit, or whether the value is estimated or measured;
- Reference metadata is used to describe the dataset as a whole, including its publisher, information about how measurements were made and a SPARQL endpoint through which the data can be accessed.

The data cube vocabulary defines dimensions, attributes and measures. Dimensions provide organizational structure or context for an observation, for example, time of observation and location. Measures represent the value of an observation. Attributes describe structured meta data or information that allows us to interpret and reason about observations, for example units. Together this vocabulary can be used to reason about data contained in a data cube and also to facilitate its exchange with other services.

In this work, data cubes are used to associate external data with core entities, such as buildings. The data could be any related data including measurements from sensors or simulation results.

Data Catalogue

Data Catalogue (DCAT) is an RDF vocabulary that allows reasoning about online data catalogues (Maali & Erickson 2014). Its use promotes discoverability of data sources and admits consumption of catalogues' metadata. The vocabulary also provides metadata that can be used to reason about a dataset's provenance. The vocabulary provides a rich set of attributes that can provide context about a dataset and its versions. A catalogue entry can be maintained on a server that is remote from the dataset's SPARQL endpoint.

For the purposes of this paper, data catalogues are used to document remote data source such as datasets from sensor sets and results from simulations. This arrangement means that the server manages a reference to some dataset and does not maintain a copy of the data (reducing data duplication and management overheads) as well as ensuring that the dataset itself remains under the control of its owner.

Notifications

Software applications manipulate information or produce new information as a result of their execution. If several applications are used together to achieve some task, then they define a tool chain and implement a process where certain tasks must be completed by particular applications in a particular order. When a user is using the applications, they are in effect the coordinator of the process, and are responsible for the correct execution of the process. To automate such a process, software applications must be capable of producing notifications to indicate that they have completed some task, and also be capable of receiving and acting on relevant notifications from other applications.

A common design for a notification system uses a central coordinating server to which attached clients can publish notifications. Other clients can register interest in these by subscribing for notifications from the server. W3C's Linked Data Notifications defines a protocol for exchange of notifications.

In this work, Linked Data Notifications (Capadisli & Guy 2017) are used to develop tool chains that can act on data exposed by the server; note this data may be accessible through a SPARQL endpoint exposed by the server itself, or one referenced by a data catalogue entry. Clients can send notification to notify the server (and its connected consumers) of a change to a data set (when a new set of sensor readings are available) or when some new artifact has been produced (when results of a simulation have been produced). Clients can trigger actions based on receipt of some notification and so a collection of clients (each an element of a tool-chain) can co-ordinate their actions.

Implementation

This section describes how the DDIS was implemented, with particular focus on how the server was developed to meet its goals. The server seeks to act as a data store which client software can submit to and retrieve information from. This information, while submitted in disparate formats, is represented on the server as a single, unified source which can be queried by other software clients.

This is achieved by creating semantic bridges between formats. The store is extensible, and can include data that is remotely located from the server, and so, allows data owners to maintain control over their data. The server is implemented using mature, cost effective techniques to reduce cost, ensure maintainability and availability.

The Scenario

The DDIS grew out of a previous project[7]. While the project had several nascent versions of features described here-in, functionality was centralized, clients and server were tightly bound, meaning that repurposing would have required a significant refactoring effort. For example, in the project, modification of attributes provided by a data collection tool required reworking of both the database and code used to expose the attributes to client systems. The goals outlined for the DDIS seek to address these limitations and make the server readily applicable to a wider range of scenarios.

The scenario used to illustrate the DDIS is derived from this earlier system, and consists of three client components in addition to the server; these are a standalone data collection tool, a simulation engine and a reporting client through which end users can view results. It is assumed that each of these clients are provided by separate vendors, that their implementation is commercially sensitive and that each must act as data controller for information artifacts generated by their tools.

The data collection tool collects attributes required by the simulation tool. It is able to determine when it has collected sufficient information to allow the simulation engine run. When end users have completed data entry, the tool exposes this information and creates a data catalogue for that information on the DDIS. It also posts a notification to indicate that data collection is complete and provide details of the data catalogue entry. As an interested client, can then either alert its operator (or automatically) retrieve the collected information run a simulation on it. The simulation engine then adds a data catalogue entry for the simulation results, and posts a notification. The reporting interface, having expressed interest in such events, then retrieves the result set and displays it for the end user.

Guiding Principles

The design and development of the Dynamic District Information Server was guided by two principles; the first, ensured that the server was flexible enough to be customized for specific project requirements and, secondly, ensuring that operation of the server was affordable. Flexibility was achieved by keeping core data representations - such as bridges between various information formats - simple, and devoid of extraneous detail; this detail can be added as bespoke functionality, or accessing external data stores.

The second goal of reducing the cost of operating the server is achieved by mitigating costs around setting up and maintaining the server. Costs arise through the acquisition of hardware, which must be maintained by trained personnel, and often requires bespoke supporting infrastructure. The DDIS is deployable as a turn-key machine image that is deployed to cloud based servers such as those provided by Amazon Web Services and Microsoft Azure. This approach is advantageous as the effort of maintaining hardware and software services, such as security, are delegated to specialists in these companies. Furthermore, these virtual servers will not suffer failures resulting in extra expenses and reduced availability. In addition, hardware can be scaled to demand dynamically, ensuring that cost of operation is optimized.

The DDIS is then initialized through the use of a web based wizard; users from a civil engineering background did not have to develop specific technical skill outside of their domain to use the server. The work described in this paper was run on a medium EC2 AWS instance, and supported data collection and simulation clients.

Architecture Overview

The DDIS consists of four key functional areas (shown in Figure 2):

- A series of interfaces to allow client software to query (through RESTful or SPARQL based interfaces) and to interact with the notifications system;
- A DCAT compliant data catalogue;
- A Core Spine or RDF schema that can be used to place other information in the space and time covered by the project. This spine allows information to be associated with entities in either the urban or building information spaces, and creates a bridge to allow seamless querying across these spaces;
- Subsystems to manage the native formats of both the urban and building spaces;

The DDIS server provides a semantic context for building related data, placing it in time and space; this information can be local or remote. The resulting context can be used by clients to query the associated information. The server can be queried through either RESTful APIs or SPARQL query interfaces. The server also provides a notification service to allow attached clients to inform each other of events. In order to define a common context, a definition must be provided in terms of information available in the

[7]http://newtrend-project.eu

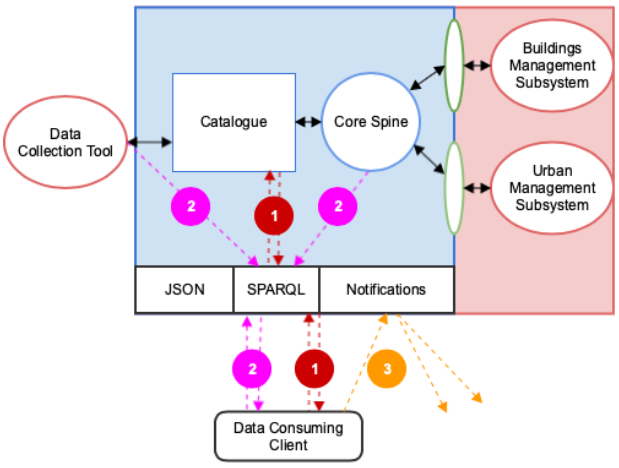

Figure 2: DDIS Architecture Overview

project. Ideally, the context is agreed between organizations participating in the project.

The server would be set up thus; both the owner of the server and any other members of the project that will manage data would meet at the project's beginning. The structure of the Core Spine would be agreed, including supported data formats (in the case of the NewTrend project, these formats included CityGML and IFC). Once this structure was agreed, individual data stakeholders can proceed to publish their data in a way that links to this spine schema. As each contribution was made available, its location would be published in the DDIS' data catalogue. There after, when a client tried to access this data, they would first query the catalogue for the available RDF schema (Step 1, Figure 2), and using these, formulate SPARQL queries across both the Core Spine and the information sources described in the catalogue (Step 2, Figure 2). A separate process management schema based on notifications (described later) is represented by Step 3, Figure 2. The structure of the core spine, shown in Figure 3, is intentionally minimalist. Its purpose is to provide a generic context in place and time to other information artefacts. Entities - buildings, building components or groups of buildings are contained by or contain other entities. Each version of an entity is versioned using a change event which has a time and changing agent. This change event can be used both to version and inform the provenance of entities. Finally, an entity can act as a binding element between formats represented on the server. For example, in the scenario, where an entity, such as a building, is represented in both CityGML and IFC formats[8], the identities of the entity in both formats can be associated as values in the bridging entity. The spine is managed using the Apache Jena project[9], with particular use being made of the RDF API, ARQ, TDB and the Ontology API. This infrastructure is made available through the Fuseki project[10].

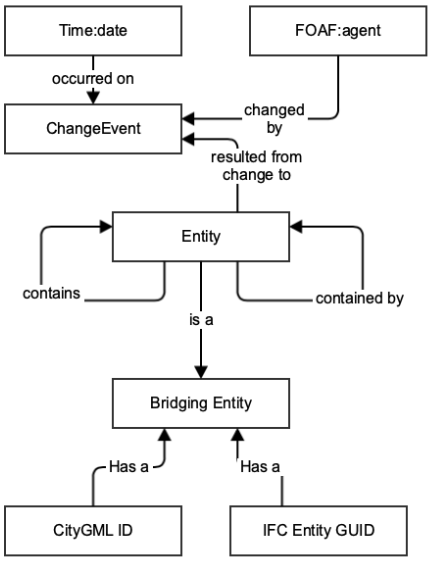

Figure 3: Core RDF Schema

The scenario called for management of data in two formats, CityGML (Gerhard Gröger 2015) and IFC. In this case the DDIS managed these formats using mature, widely adopted software systems, and interfaced with these to produce RDF on the fly. CityGML was managed on a POSTGIS database[11] using Virtual City Systems 3d City Database Importer/Exporter tool[12]. IFC is managed using the open source BIMServer[13]; its APIs and plugins are used to interact with the buildings' representations. These are sources are accessed using Python libraries - that query POSTGIS in the case of CityGML or through the BIMServer's API for IFC (represented as green elipses in Figure 2).

Adding Other Data

Organisations can publish data and add it to the server. This information must be expressed in the context of the core RDF schema for the project. For example, in the case of the scenario, information would be associated with some version of an entity; for example, sensor data might be expressed as being measured in a room of a particular building. Information is expressed in RDF. Any vocabu-

[8]We focus on these formats, other formats can be supported

[9]https://jena.apache.org

[10]https://jena.apache.org/documentation/fuseki2/index.html

[11]https://postgis.net

[12]https://www.3dcitydb.org/3dcitydb/3dimpexp/

[13]http://bimserver.org

lary can be used, though, the Data Cube has been found to be particularly suitable, given its support for expressing information in several dimensions.

Once the information has been acquired, it must be made linkable. This can be achieved using varying degrees of sophistication. Simple publishing the file on a web server will expose information statically. Information can also be exposed dynamically, where a web server accesses some information source (such as a database) and transforms an information subset to a data cube. Both processes ensure that the data is controlled by its owner. This approach helps maintain compliance with data privacy best practice.

Finally, information must be made discoverable. This is done by adding a data entry to the DDIS data catalogue. This entry is in turn exposed using the DCAT vocabulary. The vocabulary is expressive, allowing details including the title, version, data of publication and modification, data use licence URL and many more attributes to be expressed. This entry is discovered by connecting clients. The structure of the data is provided, and can be used to formulate a SPARQL query.

Data access is further controlled through the DDIS by adding access control for artefacts listed in the catalogue. DDIS user accounts are grouped, and access privileges are determined by these.

Process Support

While each client of the server can interact independently with the DDIS, these tools are often used as part of defined processes, creating ordering dependencies in their use. For instance, an energy simulation engine cannot run until it can retrieve building geometry, district data and other parameters from the server. The DDIS provides a notification service that allows other tools to register custom notifications on DDIS, provides an interface to allow a tool to publish these notifications, and allows other clients to subscribe for notifications that are relevant to their operation. Available notification definitions can be retrieved from the server. These definitions describe both notification types and the cargo element of each. Both a data pull and push methodology are provided. The pull methodology is implemented by the client polling for notifications of interest; this technique was initially provided, but was found to be inefficient because of the amount of network traffic generated and the extra computational effort imposed on the DDIS. A pull service was implemented where a client provided a callback server call. In this case, on receipt of a notification, the server posts a notification to interested clients' callback. This call then invokes the clients response to the notification. This process is described in Figure 4. The schema used builds on W3C's Linked Data Notifications standard[14]. It was built using a server add-on called Django Notifications, though it could also be built using any cloud based notification service.

In the case of the scenario, two components needed to work with the server in sequence. The data collection tool must first complete data collection. Once this is complete, the tool posts a notification to the server. When the project was set up, the client running the simulation engine posted interest in notifications from the data collection tool; when it did this, it provided a URL that would invoke the simulation clients response. On receipt of the notification from the data collection tool, the server makes a call to the URL and the simulation client retrieves the data from the server.

Conclusions

This paper introduced the DDIS and a scenario to which it was applied. The server contributes the following:

- Provides a semantic context to place information in time and space
- A data catalogue to make project information discoverable
- A notification service to allow clients attached to the server interact with one another

Together these features provide the tools to help stakeholders and toolchains in a project to interact with one another in a way that is relatively cheap to run and maintain, while addressing real world commercial interests, such as allowing participants to maintain control over their data.

Future Work

The development of the DDIS server has highlighted several future avenues of investigation, including dynamic toolchain definition and data provenance.

Other formats, such as W3C's LBD standard will be integrated. In addition, the notification system will be extended. While the server provides a mechanism through which clients can notify the server of some event. This notification can then be polled by other clients, and used to trigger events on these. Future work will include providing a semantic infrastructure through which complex processes can be defined.

Data Provenance is concerned with providing traceability for information. While information artefacts can be versioned using many current information management system, little meta-data about the production of information is captured and maintained. Furthermore, once that information has been used in a simulation, few if any systems record details of datasets input to these. The

[14]https://www.w3.org/TR/ldn/

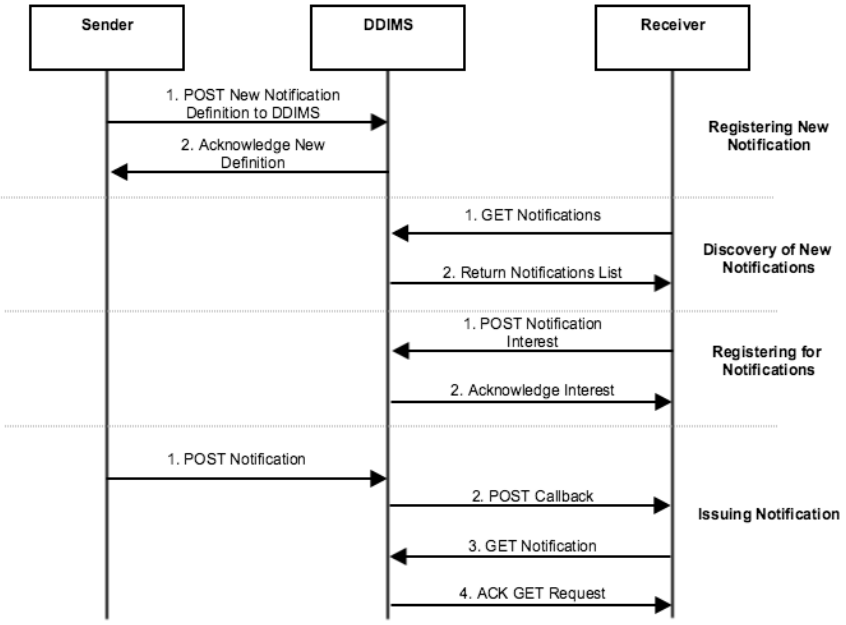

Figure 4: Notification Flow

authors consider that being able to query for, for example, which simulation results depend on a mis-calibrated measuring device, is a key extension to the current DDIS server. While several approach to this functionality exist, it remains to be determined which of these will provide the desired accuracy, while optimising computational and storage costs.

Acknowledgements

This publication has emanated from research supported by a research grant from Science Foundation Ireland (SFI) under the SFI Strategic Partnership Programme Grant number SFI/15/SPP/E3125. The opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Science Foundation Ireland.

References

Azhar, S., Hein, M. & Sketo, B. (2019), 'Building information modeling (bim): Benefits, risks and challenges'.

Belanger, F. & Crossler, R. (2011), 'Privacy in the digital age: A review of information privacy research in information systems', *MIS Quarterly* **35**, 1017–1041.

Big and Open Linked Data (BOLD) in government: A challenge to transparency and privacy? (2015), *Government Information Quarterly* **32**(4), 363 – 368.

Boddy, S., Rezgui, Y., Cooper, G. & Wetherill, M. (2007), 'Computer integrated construction: A review and proposals for future direction', *Advances in Engineering Software* **38**, 677–687.

Capadisli, S. & Guy, A. (2017), Linked Data Notifications), Technical report.

Cyganiak, R. & Reynolds, D. (2014), The RDF Data Cube Vocabulary, Technical report.

Department of Communications, E. & Resource, N. (2014), 'A national renovation strategy for ireland'.

Gerhard Gröger, Thomas H. Kolbe, C. N. K.-H. H. (2015), OGC City Geography Markup Language (CityGML) En- coding Standard, Technical report.

Maali, F. & Erickson, J. (2014), Data Catalog Vocabulary (DCAT), Technical report.

Mcglinn, D. K., Wicaksono, H., lawton, w., Weise, M., Kaklanis, N., petri, i. & Tzovaras, D. (2016), Identifying use cases and data requirements for bim based energy management processes.

Mihindukulasooriya, N. & Menday, R. (2012), Linked Data Platform 1.0 Primer, Technical report.

Yoshida, T. & Hirashita, H. (2003), 'Study of data exchange for use by construction information systems'.