2023 European Conference on Computing in Construction
40th International CIB W78 Conference
Heraklion, Crete, Greece
July 10-12, 2023

A REFERENCE FRAMEWORK ENABLING TEMPORAL SCALABILITY OF OBJECT-BASED SYNCHRONIZATION IN BIM LEVEL 3 SYSTEMS

Sebastian Esser[1], Simon Vilgertshofer[1], and André Borrmann[1]
[1]Technical University of Munich, Munich, Germany

Abstract

Model-based collaboration has been identified as one of the key concepts in Building Information Modeling (BIM) to enhance productivity in construction projects. Today's best practice and international standards demand "BIM level 2" which is based on transferring entire (domain) models through file-based exchanges. However, this coarse granularity causes severe limitations regarding concurrency, conflict resolution, and traceability. To overcome these limitations, an increasing number of researchers are investigating incremental model updates. In this paper, we critically assess current applications of BIM level 2 collaboration and summarize strategies for BIM level 3 with regard to interdisciplinary application scenarios, ownership management in federated environments, and their integration into existing collaboration workflows. Furthermore, we present a reference architecture that reflects the aspects of user interaction, application logic, and incremental update exchange that are deemed necessary for future BIM Level 3 collaboration systems.

Introduction

To date, model-based information exchange is used in numerous projects carried out in the AEC industry. The interaction between various stakeholders in a construction project using BIM models with geometric and semantic information is denoted as BIM level 2 and reflects an advanced maturity stage according to the definitions given in PAS 1192-2 published by the British Standards Institution (2013). Prior to this stage, BIM levels 0 and 1 addressed project management with less digitization. The main communication means were still 2D plans and documents but the first 3D models have been produced and shared as additional resources.

Today's broadly adopted best practice is denoted as BIM level 2 where collaboration among various project teams and stakeholders is realized through *Common Data Environments* implementing the concept of model federation. A CDE platform enables interdisciplinary collaboration and offers procedures for structuring, merging, distributing, managing, and archiving digital information as part of a holistic project management approach. Today, information to be shared over a CDE appears mostly in the form of BIM models, which contain geometric and semantic information representing the design and engineering in a computer-readable manner.

From a technical point of view, CDE systems follow the specifications provided by a series of standards including ISO 19650 CEN (2018) and implement model federation by means of file-based data exchange. At its heart, the principle of model federation relies on discipline-specific partial models (such as architectural, structural, or HVAC models) that are created and maintained independently, but coordinated in well-defined intervals to achieve a coherent overall design solution. The main motivation for following the federated model approach lies in the legal demand that the responsibility for a discipline model must remain with the respective author. In implementing this approach, the current best practice relies on transferring complete domain models stored in files each time a model version is shared. Since BIM models can easily reach the extent of several Gigabytes, this process creates substantial network traffic and consumes a significant amount of storage space. More severely, model changes are not tracked by the system, and thus have to be identified manually in the coordination session. This poses substantial limitations to the collaboration procedure.

Looking ahead to the upcoming BIM level 3, collaboration systems are expected to obtain a more complete integration of design information authored by each discipline and to implement tighter connections between the federated BIM models. Specifically, the next maturity level should provide a more agile and flexible means to react to model changes and alterations during the design process. To fulfill these requirements and to overcome the aforementioned limitations, the authors propose to transfer only model changes instead of entire BIM models. As discussed in detail in the following sections, this approach will reduce network traffic and storage space and allow the effects of changes to be evaluated much more efficiently by the collaborating stakeholders.

Motivation and Research Gap

The paper at hand introduces a technical approach that overcomes the current limitations of BIM level 2 CDEs with regard to their version management of BIM models. To this end, it is necessary to identify given peculiarities and historically grown constraints that are characteristics of the AEC industry. Furthermore, current limitations in BIM level 2 implementations are analyzed and compared to other technical systems that are already broadly used in other industry branches. Finally, a distributed system architecture is introduced that acknowledges industry-specific limitations and provides a novel method for bet-

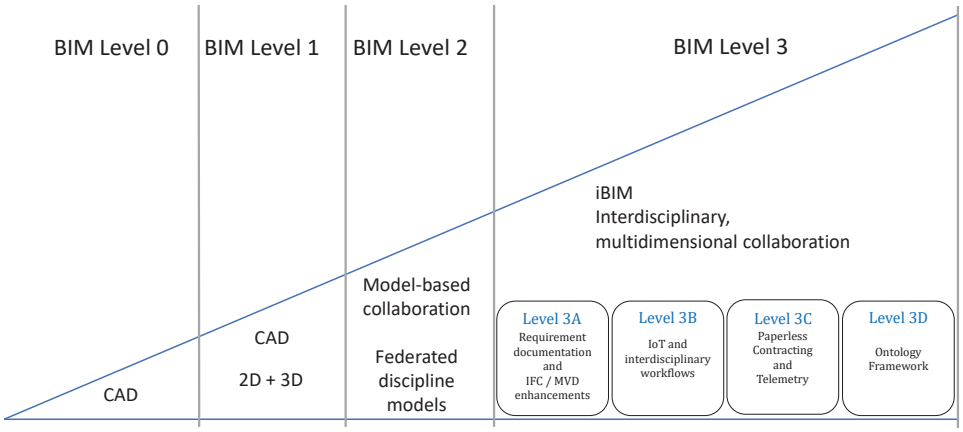

Figure 1: BIM Maturity Levels specified in PAS 1192 and combined with the expected technical enhancements specified in the strategy paper of Digital Great Britain (inspired by British Standards Institution (2013); Digital Built Britain (2015))

ter version control in iterative, interdisciplinary design projects.

Background and related work

The following paragraphs summarize relevant publications in the context of BIM maturity levels from a technical and management point of view. Furthermore, an overview of existing approaches for version and revision control of BIM models is provided.

Model-based collaboration and BIM-maturity levels

Prior to the BIM maturity model in PAS 1192-2 published in February 2013, Succar (2010) described multiple BIM stages that can be understood as a predecessor version of the levels defined in PAS 1192-2. He distinguishes in total five BIM capability stages, where the first four are in essence similar to the maturity levels of PAS 1192-2. For BIM stage 3, Succar expected interdisciplinary nD models that allow complex analyses at early stages and a synchronous interchange of information among project partners. Accordingly, the strict borders between specific design phases should become less important as detailed information can be produced on a better-informed basis already in earlier project phases. As the ultimate and subsequent stage, Succar expected the "Integrated Project Delivery" phase which is characterized by interdependent discipline models and real-time collaboration in interdisciplinary settings.

Even though Succar's expectations for BIM stages 1 and 2 have been met by evolving technology in the past decade, there are still major deficiencies when applying BIM level 2 in practice. Awwad et al. (2022) have analyzed the application of BIM levels 1 and 2 in small and medium-sized enterprises (SMEs) in the UK using literature reviews and expert interviews. They concluded that the true potential of BIM-related techniques has not been fully exploited by SMEs in the UK construction sector yet. Attrill and Mickovski (2020) and Alankarage et al. (2022) report various concerns and shortcomings that hinder the adoption of BIM Level 3 from practitioners' perspectives and let them stick in incomplete applications of BIM level 2. Their outcomes are based on interviews and reveal that concepts commonly understood as BIM level 2 are still not fully adopted in all parts of the AEC industry. In particular, both have identified that BIM practitioners and BIM managers still lack sufficient skills to perform BIM principles to the full extent. Furthermore, they highlight missing consistency control between BIM models and 2D drawings that have been derived from these BIM models. If changes are applied to a model, the plans are often not regenerated as well. Related observations have been reported by Kurul et al. (2016). According to their literature research and case studies, the adoption of model-based collaboration has not been established in the UK construction market even though public administrations and large contractors have extended their demands. Nevertheless, it appears relevant to consider the existing proposals of BIM level 3 approaches to resolve the highlighted limitations of level 2 platforms.

Digital Built Britain (2015) has published a strategic plan that refines the rather coarse definitions of BIM level 3 given in PAS 1192-2. According to this document, BIM level 3 should leverage how data is generated and analyzed during a built asset's design, construction, and operation phases. To accomplish this vision, a delivery program defines four sub-levels of BIM Level 3. BIM levels 3A and 3B aim to address the shortcomings of BIM level 2, especially including model federation and integration and overall data management in cloud-based systems. Later, levels 3C and 3D should mainly address economic and enterprise-related topics like tendering and contracting and therefore aim to introduce enhanced techniques. Figure 1 illustrates the maturity levels according to PAS 1192 from a technical point of view and extends level 3 by the visions specified in the strategic plan.

Besides the challenges caused by the production and interpretation of exchanged data, it is also relevant to consider how additional data captured by various acquisition meth-

ods can be integrated. Bucher and Hall (2020) have proposed a terminology to distinguish different dimensions of interoperability in the context of CDEs. According to their observations, current CDE systems can be seen at a one-dimensional interoperability level because systems enable data exchange by means of BIM models but typically have limited options to connect additional information to them. Two-dimensional interoperability is characterized by the advent of exchange interfaces between CDE systems whereas three-dimensional interoperability extends this idea by connecting enterprise-related platforms to various CDE systems. Whereas the latter case is not been further documented or investigated so far, the former vision of unified interfaces between CDE platforms has been tackled in DIN SPEC 91391-2 (DIN Deutsches Institut für Normung e. V, 2019) and related publications (Senthilvel et al., 2020). Hence, further developments are expected throughout the next years.

Version control and dependency modeling in CDEs

Despite the advances in technology for sharing and managing BIM data, the design of buildings and infrastructure facilities remains a highly iterative task due to the fact that it is a multi-objective problem that often does not have one optimal solution. Instead, design requires the balancing of a multitude of objectives and fulfilling various constraints. In consequence, tasks that involve multiple disciplines in a project are subject to several iterations, which results in numerous versions of the federated discipline models.

To observe consistency aspects across multiple discipline models coordinated in a CDE, Preidel et al. (2017) have presented an integration framework that enables a seamless connection between multiple authoring systems and a CDE platform. However, despite the technical interaction between various applications, their approach does not provide comprehensive techniques for the case that multiple versions of a model are created and populated over the project platform.

So far, the version and revision control of BIM models has been subject to only a few investigations in academia and practice. Nour and Beucke (2010) are the first to discuss object versioning as a basis for design change management within a BIM context. Schapke et al. (2018) have proposed naming conventions or recommended assigning metadata for the unique identification of each model version. However, these approaches do not enable users to access the changes made but demands manual identification and checking for possible inconsistencies caused by foreign model updates. Simeone et al. (2018) have summarized current CDE systems as mainly document-oriented. According to their explanations, level 2 CDEs focus merely on sparse meta-information for each data set and offer limited capabilities to link data sets for comprehensive management and analysis tasks. To overcome these limitations, they have proposed the application of graph-based databases, which enable high flexibility in instantiating relationships.

The application of version control for BIM models and digital twins was recently addressed in a number of publications (Esser et al., 2022b; Jones et al., 2021). Shafiq et al. (2018) suggest managing iterative changes in BIM models as a database problem, exacerbated by the long transaction times needed to support collaborative design progression. Other research groups propose to implement update mechanisms between different BIM software tools using dedicated exchange languages (Poinet et al., 2021). These approaches are promising techniques for future BIM collaboration systems but they only solve a small fragment needed to define a solid base for future BIM level 3 collaboration. Hence, the next paragraphs aim to connect the outlined proposals with the propagated goal of "integrated BIM".

The BIM Level 3 Reference Framework

To achieve the goal of improved support for collaboration on the basis of federated models, it is essential to discuss (i) how relationships and links between heterogeneous design information can be maintained and (ii) how the iterative nature of engineering and design is considered. Methods to establish links between specific information provided by diverse domain models have been addressed by several researchers lately (Beck et al., 2021; Curry et al., 2013). The presented framework discussed in the paper at hand, however, focuses on the version control problem and approaches the topic of dependency modeling.

Principles, premises and assumptions

The proposed BIM collaboration method is based on a number of principles, premises, and assumptions that are discussed in the following and depicted in Figure 2.

Given lacking support for the incremental exchange of modified BIM models, the authors envision BIM level 3 systems to combine established means of version management systems with established principles of model-based collaboration. Therefore, the paradigm of federated discipline models is preserved to maintain the clear handling of authorships and responsibilities. Collaboration in one monolithic model that is jointly edited by multiple actors has been evaluated positively for closed collaboration within a single design team but is seen as critical in interdisciplinary cross-company settings (Preidel et al., 2017). Instead, we assume a central BIM collaboration hub for managing all project-wide data. This server provides similar services as existing CDE systems, including persistent data storage and access rights management, but does not rely on file-based technology anymore.

While it is assumed that each discipline continues to produce BIM models in an authoring tool that best suits the specific design task, models should be curated in open, vendor-neutral data representations. Existing data models like the Industry Foundation Classes (IFC) for BIM models or CityGML as an important exchange standard for city models have been continuously extended towards the support of new use cases and data exchange scenarios (Jaud

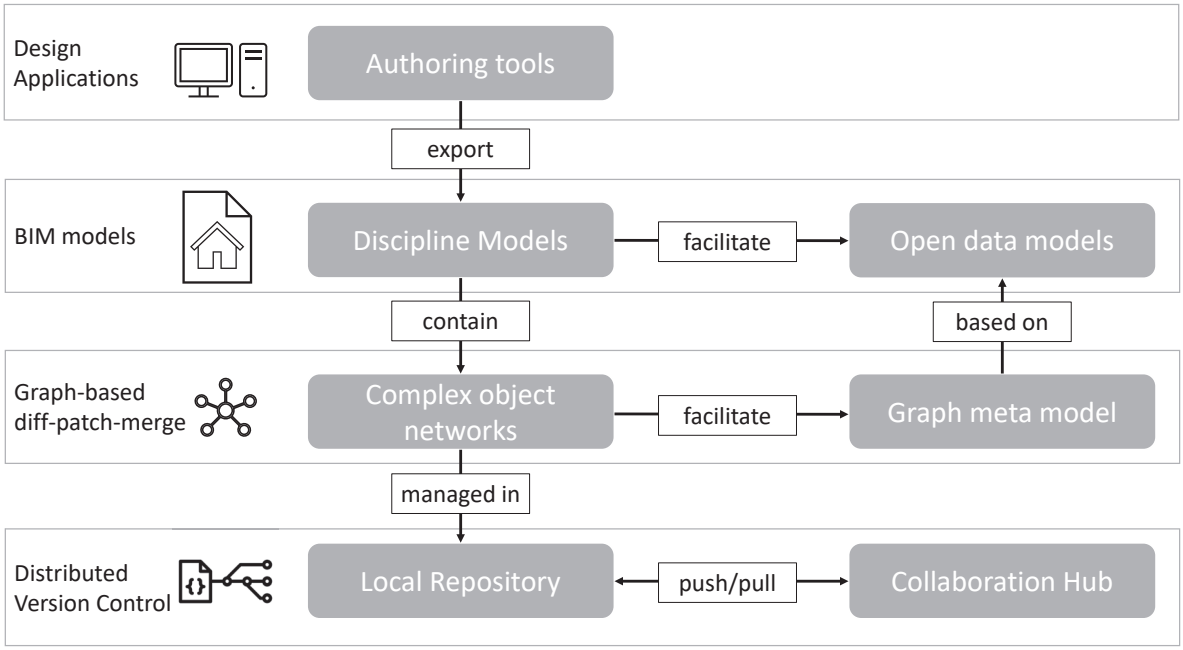

Figure 2: General overview about the premises and assumptions made for the reference model

et al., 2020; Kutzner and Kolbe, 2018). Therefore, it can be expected that the market, especially public authorities, will further enforce the demand to support these data models. Hence we assume open, standardized data models to form the basis for any data exchange across the distributed system.

As extensively discussed in Esser et al. (2022b), there is a clear need for improved change tracking in BIM level 3 to better support the coordination across the various disciplines and stakeholders and cope with the iterative nature of design processes. To this end, we propose to make the progression to incremental updates, also denoted as delta transmissions. The advantage of communicating changes instead of complete models lies not only in reduced network traffic, but also in transparent messaging and, most importantly, the possibility to install triggers for automating reactions to others' modifications on the receiving side. This may range from performing a collision check, over the generation of an entry in a to-do list up to the automated update of the receiving user's own BIM model.

There are multiple approaches to realize the principle of incremental updates, including text-based diff-and-patch as commonly applied in concurrent software development. However, due to the non-linear graph-like structure of BIM data, we argue that representing a model as a graph and transmitting the corresponding changes as graph updates (transformations) is a much more versatile approach with high descriptive power (Esser et al., 2022b). The usage of graphs as fundamental representation results in graph databases being the optimal technology for persistent storage, providing fine-grained object-level access to BIM model information at the central repository. Respecting the principle of model federation, however, each domain model is represented by a separate graph.

The implementation of the aforementioned principles forms the basis of a powerful version control approach that provides the flexibility of asynchronous collaboration while enabling overall consistency preservation. In analogy to code versioning systems such as Subversion or Git, the point of time of providing updates to the other stakeholders is freely chosen by the model author, as well as the time for integrating changes of others. We will discuss below, however, that by using fully automated updates and integration, our collaboration approach can be stretched to eventually become synchronous collaboration.

It is anticipated that implementing comprehensive version control methods will enable project partners to share and consume updated information more proactively. At the same time, project participants will greatly benefit from the communication of model updates as they gain direct insights into the changes made to foreign models and immediately evaluate the impact on their design tasks.

Version control: Workflow and commands

Only a few additions to existing concepts are required to introduce version control into the established principles of model-based collaboration. Figure 3 provides an overview of the proposed workflow and the corresponding commands. The overall system architecture follows the principles of distributed version control (Ruparelia, 2010) where the server repositories are mirrored to each machine, thus creating a local version of the repository. Each user tracks changes in the local repository which is subsequently synchronized with the remote servers (which is located on a

hub).

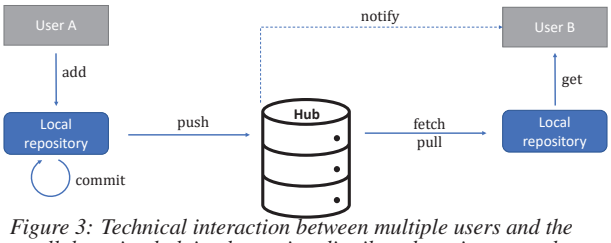

Figure 3: Technical interaction between multiple users and the collaboration hub implementing distributed version control

Every user requires a set of basic commands to participate in the incremental version control system. The commands are closely aligned with the notions and semantics used in the version control system Git (Chacon, 2009; Blischak et al., 2016) and are illustrated in Figure 4. All commands are part of a command-line application called `conman` that has been prototypically implemented to demonstrate the overall system.

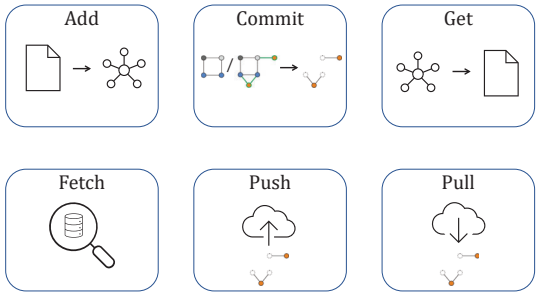

Figure 4: Set of commands exposed to an end-user performing version control over BIM data in the local repository

The *add* command adds a model to the local repository. In the graph storage of the repository, multiple versions of a model can exist, reflecting different design variants or, more likely, evolved versions of a model. By performing the *commit* command, the latest committed version of a model is compared to the most recent version and a subsequent patch is generated. The update patch is then stored in the patch storage and can be synchronized with the hub. In case a model has been added to the graph storage in only one version, the entire graph reflecting this version is committed, which reflects the initial upload of a model.

Given the repository storage being a graph storage, the *get* command enables the user to translate a graph from the graph storage back into a file-based (i.e., serialized) representation. The parallel use of files and graphs on the client's machine enables the support of exchange interfaces in existing (legacy) software applications. From a long-term perspective, however, it is anticipated that software tools will directly connect to the graph storage to either add a new version or retrieve version-controlled models from the graph storage.

Figure 5 illustrates an exemplary situation. The user has various discipline models stored as files. By performing the *add* command, a model is inserted into the graph storage. For further identification of each model, a stable universally unique identifier (UUID) is attached to each node, forming one version of a model (indicated as `ts` in the figure). Furthermore, the timestamp at the model created is attached to the nodes of the respective graph as a unixtime value. Combining these two labels enables the system to identify which graphs reflect a particular model and its associated versions without requiring naming conventions to be applied. Furthermore, the user can specify a commit message, which gets attached to each patch authored during the commit execution.

To keep track of versions already committed, an additional attribute *COMMITTED* is used as a flag indicating that a version in question has already been committed in a previous commit. In the example illustrated, three versions of the discipline model with *MODEL_ID* 1 have been added to the graph storage. In an earlier commit, a patch was created that reflects the changes between timestamp 0 and timestamp 1. Hence, the commit flag has been set to *true* for timestamp 1. Executing the *commit* command in the illustrated situation results in a patch that upgrades model 1 from timestamp 1 to 2 and exchanges the model indicated by id 2 in its initial state.

Three additional commands are exposed to the end-user to exchange commits with other project stakeholders. By performing the *push* command, the committed changes (i.e., the patches) are synchronized with the central collaboration hub. On the collaboration hub, an event log of received commits and their corresponding initiators is produced. The *fetch* command enables a user to oversee commits of foreign users that have been synchronized with the central repository but not yet applied to the local repository. Finally, the *pull* function downloads the pending commits from the server and applies them to the data stored locally in the repository.

Implementation of the repository storage and diff-patch mechanism

Unlike Git, the proposed version control system does not make use of text files but relies on graph representations. The key motivation for applying graphs as a basis for the discussed version control problem is the intrinsic nature of BIM data. Most data models, including the widely established IFC data model, represent BIM information in a network of interconnected objects. For example, any built element in a BIM model has multiple relationships to other built or spatial elements and geometric representations. Accordingly, graph databases are the optimal solution for hosting and managing BIM instance models and providing the technical basis for the local and central repositories (Jeon and Lee, 2022; Angles, 2012). In consequence, diffs are determined on the basis of graph comparison, and patches are represented by graph transformations. This has the positive side effect that the representation is immune to changes in the serialization order. Details are provided in (Esser et al., 2022b).

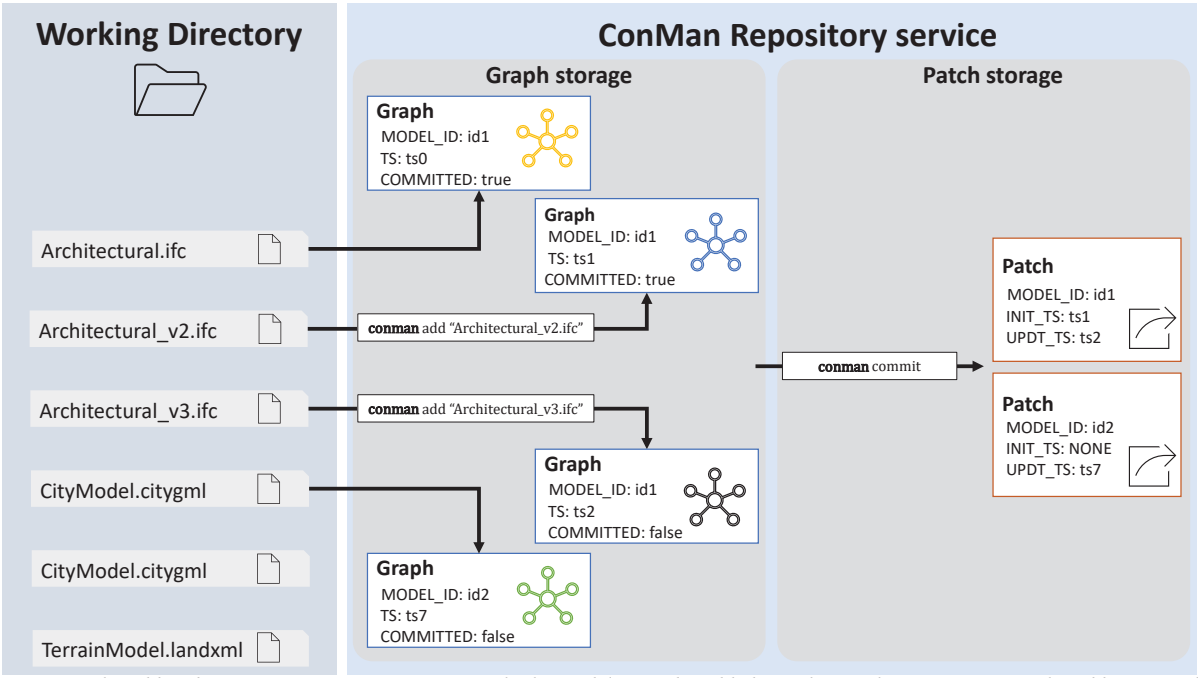

Figure 5: The add and commit process: To create a patch, the models must be added into the graph storage running the add command. Then, the commit command detects the most recent version and the last committed version and produces the corresponding patches

Eventual consistency and synchronization scalability

The outlined system enables stakeholders to achieve eventual consistency after performing the corresponding commands. Therefore, it adopts the BASE (basically available, soft state, eventually consistent) transaction model, which is generally known from NoSQL database systems. Each discipline continues to work independently and thus creates versions of BIM models that are not immediately synchronized with all other actors. Instead, the end-user must actively perform the commit and push command to release the updated information to all other domains. In consequence, there is the risk of concurrent changes applied by other team members to a model, which leads to two further considerations.

On a project coordination level, however, the eventual synchronization of locally stored patches with the collaboration server can still lead to situations where design tasks are performed on outdated information provided by foreign disciplines. This deficiency is complex to overcome in a general manner as it requires sensitive consideration of the individual management policies in each project. The phenomenon is inherent to long transactions (Aish, 2000; Weise et al., 2004) and is well-known also in the field of concurrent code development. Here, best-practice policies and workflows are applied in an informal, i.e. not technically enforced manner, such as ”commit early and often” or ”short branches, quick merge” to avoid the divergences growing too large. A similar approach should be taken in collaborative BIM projects.

As a dedicated enhancement over existing CDE platforms, however, the collaboration server provides an additional notification interface implementing a publish-subscribe pattern as it has been discussed in Esser et al. (2022a). In addition to a manual *pull* operation applying pending patches to the local repository, each end-user can subscribe to events (i.e., recently pushed patches) of a specific domain or change event and receives notifications accordingly. The notification interface enables stakeholders to precisely control, if and how often local copies of other discipline models should be updated. This has particular relevance when they are used as references for their own model. The time interval of pulling pending patches into the local repository can be chosen as either rather long (e.g., pulling daily or weekly) or fairly short (e.g., every minute or hour).

Furthermore, two types of integration modes are envisioned to handle updates. For incoming patches of low relevance for the user, a silent pull can be triggered when a notification has been received. This means that the user does not need to explicitly invoke the pull command but automatically stays up-to-date at any time. If also the push on the sender’s side is performed automatically after any model modification, the asynchronous mode of collaboration is turned into the synchronous one, completely avoiding any inconsistencies. The stakeholders in a project can flexibly decide on the synchronization mode that best fits their needs and workflows; a concept that we denote ”synchronization scalability”.

On the contrary, updates to foreign discipline models that could affect the receiver’s engineering domain should be indicated by a clear warning and applied only after the design impact has been evaluated.

By providing the two modes of notification handling and synchronization, the envisioned BIM level 3 system provides a sufficient foundation to (i) improve the overall information flow by means of update patches and (ii) respect the established paradigm of federated, disparate discipline models.

Conclusions

We have presented a reference model for BIM Level 3 systems enabling fine-granular version control and temporal flexibility. The proposed approach is agnostic to naming conventions, storage locations, and different serialization approaches at the client's machine. Instead, a graph storage technology has been applied, which handles different versions of a BIM model and is capable of both hierarchical data and connected object networks. If models are modified, the user explicitly commits the changes and pushes them to the server as a set of update patches.

Every party involved in the design project can specify if updates made in foreign discipline models should be picked manually by performing the pull command or silently integrated by continuously applying any new update patch. Furthermore, the user can set up notifications informing about updates emitted by other project stakeholders. The overall architecture overcomes existing issues in BIM level 2 platforms and paves the way toward incremental information exchange. As the concept of disparate (federated) discipline models remains intact, it is anticipated that the presented means for version control can be implemented into existing model-based workflows with little effort.

The authors identify the need for future research in extended reasoning about the actual content exchanged in each patch. By not only applying the patches to foreign models used for reference purposes but by proactively interpreting the impact of incoming changes, possible inconsistencies between designs from different disciplines can be detected much more efficiently and resolved accordingly.

Acknowledgments

This research has been partially funded by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) - 271444440.

References

Aish, R. (2000). Collaborative design using long transactions and "change merge". In Proceedings of the 18th eCAADe Conference, pages 107 – 111.

Alankarage, S., Chileshe, N., Samaraweera, A., Rameezdeen, R., and Edwards, D. J. (2022). Organisational BIM maturity models and their applications: a systematic literature review. Architectural Engineering and Design Management, pages 1–19.

Angles, R. (2012). A comparison of current graph database models. In 28th International Conference on Data Engineering Workshops, pages 171–177. IEEE.

Attrill, R. and Mickovski, S. B. (2020). Issues to be addressed with current BIM adoption prior to the implementation of BIM level 3. In Proceedings of the 36th Annual ARCOM Conference, pages 336–345.

Awwad, K. A., Shibani, A., and Ghostin, M. (2022). Exploring the critical success factors influencing BIM level 2 implementation in the UK construction industry: the case of SMEs. International Journal of Construction Management, 22:1894–1901.

Beck, S. F., Abualdenien, J., Hijazi, I. H., Borrmann, A., and Kolbe, T. H. (2021). Analyzing contextual linking of heterogeneous information models from the domains bim and uim. ISPRS International Journal of Geo-Information, 10:807.

Blischak, J. D., Davenport, E. R., and Wilson, G. (2016). A quick introduction to version control with git and github. PLoS Computational Biology, 12:1–18.

British Standards Institution (2013). Pas 1192-2:2013: specification for information management for the capital/delivery phase of construction projects using building information modelling.

Bucher, D. and Hall, D. (2020). Common data environment within the aec ecosystem: moving collaborative platforms beyond the open versus closed dichotomy. In EG-ICE 2020 Proceedings: Workshop on Intelligent Computing in Engineering, pages 491–500. Universitätsverlag der TU Berlin.

CEN (2018). DIN EN ISO 19650-1:2018 Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) — Information management using building information modelling — Part 1: Concepts and principles.

Chacon, S. (2009). Pro Git. Apress.

Curry, E., O'Donnell, J., Corry, E., Hasan, S., Keane, M., and O'Riain, S. (2013). Linking building data in the cloud: Integrating cross-domain building data using linked data. Advanced Engineering Informatics, 27:206–219.

Digital Built Britain (2015). Level 3 Building Information Modelling - Strategic Plan.

DIN Deutsches Institut für Normung e. V (2019). DIN SPEC 91391-2: Gemeinsame Datenumgebungen (CDE) für BIM-Projekte - Funktionen und offener Datenaustausch zwischen Plattformen unterschiedlicher Hersteller - Teil 2: Offener Datenaustausch mit Gemeinsamen Datenumgebungen.

Esser, S., Abualdenien, J., Vilgertshofer, S., and Borrmann, A. (2022a). Requirements for event-driven architectures in open BIM collaboration. In Proceedings of the 29th EG-ICE International Workshop on Intelligent Computing in Engineering, pages 45–53.

Esser, S., Vilgertshofer, S., and Borrmann, A. (2022b). Graph-based version control for asynchronous BIM collaboration. Advanced Engineering Informatics, 53:101664.

Jaud, Š., Esser, S., Muhič, S., and Borrmann, A. (2020). Development of IFC schema for infrastructure. In Proceedings of 6th International Conference siBIM: Structured data is the new gold.

Jeon, K. and Lee, G. (2022). The status quo of graph databases in construction research. In The 9th International Conference on Construction Engineering and Project Management.

Jones, D., Nassehi, A., Snider, C., Gopsill, J., Rosso, P., Real, R., Goudswaard, M., and Hicks, B. (2021). Towards integrated version control of virtual and physical artefacts in new product development: inspirations from software engineering and the digital twin paradigm. Procedia CIRP, 100:283–288.

Kurul, E., Oti, A. H., and Cheung, F. K. T. (2016). Level 3 BIM for Standardised Design Delivery, Refinement and Optimisation: Is it a real option in the UK? In CIB World Building Congress.

Kutzner, T. and Kolbe, T. H. (2018). CityGML 3.0: Sneak preview. In Kersten, T. P., Gülch, E., Schiewe, J., Kolbe, T. H., and Stilla, U., editors, PFGK18 - Photogrammetrie - Fernerkundung - Geoinformatik - Kartographie, 37. Jahrestagung in München 2018, pages 835–839.

Nour, M. and Beucke, K. E. (2010). Object versioning as a basis for design change management within a BIM context. In Tizani, W., editor, Proceedings of the International Conference on Computing in Civil and Building Engineering. Nottingham University Press.

Poinet, P., Stefanescu, D., and Papadonikolaki, E. (2021). Collaborative Workflows and Version Control Through Open-Source and Distributed Common Data Environment, volume 98. Springer International Publishing.

Preidel, C., Borrmann, A., Oberender, C., and Tretheway, M. (2017). Seamless integration of common data environment access into BIM authoring applications: The BIM integration framework. In eWork and eBusiness in Architecture, Engineering and Construction, pages 119–128. CRC Press.

Ruparelia, N. B. (2010). The history of version control. ACM SIGSOFT Software Engineering Notes, 35(1):5–9.

Schapke, S.-E., Beetz, J., König, M., Koch, C., and Borrmann, A. (2018). Collaborative data management. In Building Information Modeling - Technology Foundations and Industry Practice, pages 251–277. Springer International Publishing.

Senthilvel, M., Oraskari, J., and Beetz, J. (2020). Common data environments for the information container for linked document delivery. In Proceedings of the 8th Linked Data in Architecture and Construction Workshop - LDAC, pages 132–145.

Shafiq, M. T., Matthews, J., Lockley, S. R., and Love, P. E. D. (2018). Model server enabled management of collaborative changes in building information models. Frontiers of Engineering Management, 5:298–306.

Simeone, D., Cursi, S., and Coraglia, U. M. (2018). Reasoning in Common Data Environments Re-thinking CDEs to enhance collaboration in BIM processes. In eCAADe 2020 - Architecture and Fabrication in the Cognitive Age, pages 499–506.

Succar, B. (2010). Building Information Modelling Maturity Matrix. In Handbook of research on building information modeling and construction informatics: Concepts and technologies, pages 65–103. IGI Global.

Weise, M., Katranuschkov, P., and Scherer, R. (2004). Managing long transactions in model server based collaboration. In Proc. of the 5th European Conf. on Product and Process Modelling in the Building and Construction Industry - ECPPM, page 311. Taylor & Francis.