# Tracking model checks in information delivery controlling processes

Noemi Kremer[1]
[1]Design Computation, RWTH Aachen University, Schinkelstraße 1, 52062 Aachen
E-mail(s): kremer@dc.rwth-aachen.de

**Abstract:** While the development of checks and rule sets has been widely considered, and many contributions have been made, the process-oriented traceability of model checks has received less attention. As part of the Bundesinstitut für Bau-, Stadt- und Raumforschung (BBSR) ZukunftBau project BIM-based Information Delivery Controlling, an information delivery controlling system was developed. Part of the system is the process-oriented checking of IFC model contents with mvdXML. The model checking results are stored as BIM Collaboration Format (BCF) ZIP file. Model checking thus involves three distinct files, which have no tracked dependence on each other either before or after a model check. There is a gap in how model checks are recorded and processed in construction processes in order to reconstruct checks. This paper presents an approach on how to link model checks using open standards to enable evaluation and reconstruction capabilities in information delivery processes. Tracking files is done by storing hashes of the three files involved in the check in a database. In addition, process-related meta information of the model check is recorded. The model check tracking method is implemented and tested based on a BIM use case utilized in the Information Delivery Controlling project. Saving the hash values of the involved files enables their unique identification. Each model check can thus be individually distinguished, and duplicate checks can be recognized and avoided at an early stage. Finally, the possibilities and limitations of linked model checking in BIM use cases are discussed. Further application possibilities and challenges for linked model checking are presented.

*Keywords:* BIM Model Checking, Check Tracking, Information Delivery Controlling, Process Modeling

## 1 Introduction

In planning processes, BIM based model checks improve planning success due to the constant possibility of checking delivered information according to requirements. Process related exchange requirements at specific points of data drops in processes can be mapped with IFC elements. Developments, such as the Information Delivery Manual (IDM), support the definition of requirements on a functional and technical level. Therefore, models can be checked for exchange requirements at

defined points of data drops in processes. To perform a model check, various types of software are available, such as Solibri Model Checker [1] or Desite BIM [2].

Independently of software, model checks influence processes in the life cycle of a building, especially negative checking results can have an impact on the entire process flow, requiring a revision of the model content. For significant data drops in processes, individual model checks may have multiple iterations. Due to the importance of model checking in processes, a structured record of performed checks is essential. Recording and tracking BIM-based model checks as part of process-oriented information delivery offers a possibility to use model checking more efficiently. Model check records enable tracking and accountability of model checks within process sequences at a later date. Thus, they support the reconstruction of information delivery processes and support analyzing and assessing transactions retrospectively. Therefore, an approach is shown how BIM-based model checks can be tracked to enable evaluation and reconstruction possibilities in information delivery processes. Since model checks are highly individual, this paper presents an approach for recording model checks using the Information Delivery Controlling (ILC) project as an example.

# 2 Existing solutions and previous work

## 2.1 Model checking and data management software

BIM-based model checking is the basis for a variety of decision-making processes. Checking software such as Solibri and Desite perform BIM model checks based on IFC models. Products in the open source category mainly use open standards for checking models. For instance, the Model View Definition (MVD) Model Checker [3] from RWTH Aachen using mvdXML. Checking software stores the components of a model check within a defined project or data package: the loaded model, the defined and used rule sets, the issues found, as well as metadata. However, the storage of this information only enables the management of the files involved in the examination within the software. For example, the Solibri native format SMC is composed of geometry, sources and relationships, rule sets, results and user-defined information and can be shared within Solibri products only [4]. Open source software such as the MVD Model Checker does not offer an integrated data management option. Model checking software in general is not designed to contextualize executed model checks in information management. The focus of model checking in software products lies on rule definition and checking results saved as BCF file. Therefore, software for managing and editing BCF files, is offered by BIMcollab workflow manager [5]. Software specializing in the creation and management of exchange information requirements and quality assurance of BIM models is provided by BIMQ [6].

A Common Data Environment (CDE) defined according to ISO 19650 supports Information management and exchange processes in a common digital collaboration platform. In a CDE, processes can be created as workflows, tasks, and files can be assigned to project participants and metadata such as timestamps and progress can be recorded. However, the automated, rule-based model check is not part of the CDE. CDE software providers such as Oracle Aconex [7] or Thinkproject [8] cooperate with checking software such as Solibri and Desite BIM to enable automated model checks, but only

exchange check results so far. Reconstructing model checks is difficult since automated model checks and information management is carried out in different software.

## 2.2 Hashing

A hash function is the mapping of a string of characters or bytes to a numeric value or key of fixed length. The input set can be of any size and/or contain elements of different lengths. All files relevant in a model check are hashed with the standardized secure hash algorithm 256 (SHA-256) [9]. SHA-256 is also used as a cryptographic hash function and has a strong collision resistance. Hash values are composed of the content of the file, but are not influenced by metadata.

## 2.3 Information Delivery Manual

The information delivery manual has been introduced by buildingSMART, in order to describe process related information about who delivered what to whom at what specific point in a project life cycle [10]. According to ISO 29481-1, the basic elements of an IDM are an interaction plan/transaction diagram and/or a process diagram, and one or more information exchange requirements. In addition, there is the use case and the technical implementation as mvdXML. Since the components of an IDM are only loosely structured and hardly reusable, the third part of ISO 29481-3 defines a data schema and an XML representation, the idmXML. Each idmXML contains a fixed set of metadata, a use case, a possible process map and exchange requests [10].

## 2.4 ILC - Information delivery controlling project

The information delivery project supported by ZukunftBau and worked on by Bergische Universität Wuppertal(BUW), RWTH Aachen University and numerous practical partners involves the development of a tool for process-supported information control. The demonstration tool should improve digital information management in BIM processes on the client and contractor side. The developed tool, the MVD generator, is part of the ILC system (see figure 1), it enables an interoperable flow between the BUW process database and the MVD Model Checker [3] based on open standards. The process-modeling database at the University of Wuppertal has been continuously enriched with working processes from different perspectives since 2016 [11]. For each use case provided in the BUW process database, a workflow with the associated exchange requests was developed. The stored information can be retrieved as a table for further processing. The requirements for the tool were collected in a workshop with 11 practitioners from AEC industry and afterwards classified [10]. The developed tool in the ILC system consists of two subsystems, the MVD generator and the MVD checker. The MVD generator allows the generation of mvdXMLs from the corresponding tables in the process database. These can then be used to check properties of an IFC model. Results are saved as BCF file.

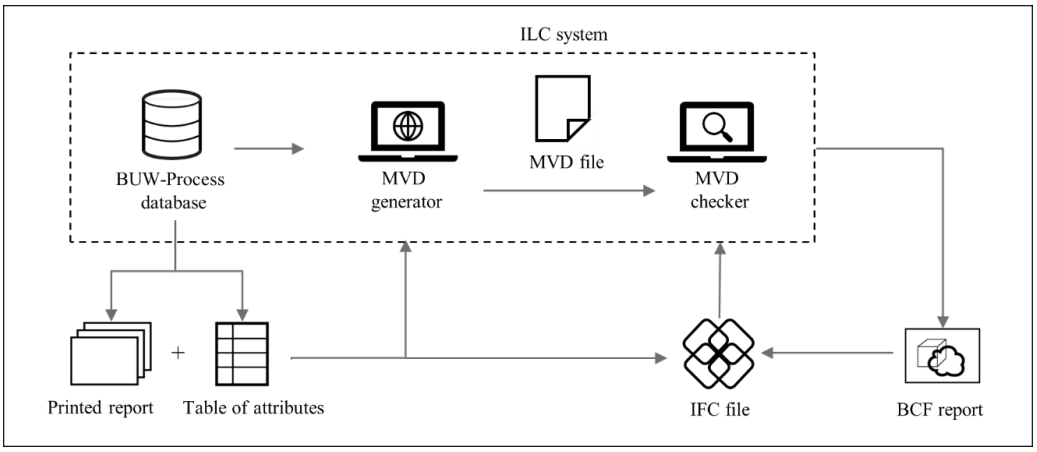


Figure 1: ILC-Project developed System [11]

# 3 Database - development and structure

## 3.1 Database development

Since model checks can be dynamic, e.g. due to iterations, the construction of a relational database is proposed. All information such as entities, attributes, and relations are represented as a collection of tuple relations, which can be thought of as tables.

To answer the question of how model checking tracking can be efficiently carried out using open standards, a combination of storing process related data and process file hashing is introduced. The development of the database is linked to the development and evaluation of the ILC system (see figure 1). Therefore, the open standards used in the ILC project and developed functional units form the basis for the structure of the database. During the development and validation of the ILC system, frequent iterations of model checks due to unspecific or incorrect rule definitions for exchange requirements and incorrect model content were necessary. A first version of the database was developed to manage these model check iterations. Since the initial situation in the ILC System contains three files: model (IFC), check file (mvdXML) and result file (BCF), the first version only stored their hash values. In order to time model checks and to identify and assign checking iterations to the model checks, further columns were added to the database. This leads to a first process-oriented sorting of model checks. Since individual model checks are assigned to process tasks, process-relevant information from the BUW-process database such as name of process owner and name of process task were added as an additional table in the database. This allows model checks to be managed by locating them in the process. The direct link to IDM was originally managed by the developed processes and exchange requirements. The database is thus closely influenced by the experiences gained in the development of the ILC system.

### 3.2 Database structure

The structure of the database was developed closely to content and requirements of data and tools in the ILC system. As shown in figure 2 the database consists of three tables, which are IDM, process and checking. A process defined within the IDM is broken down into individual transfer points that are assigned model checks. The central information is the hash values of the files. Furthermore, there is the process-oriented classification of the model check. In addition, there is metadata that provides additional information on data management.

The checking table consists of all information about each carried out model check. This information includes the hash values of all files required for a model check, as well as metadata about the checking process, such as iteration number and timestamp. The process table contains all information about the process. The information about the process name and the name of the person responsible for the process are part of the BUW process database. Sending and receiving person are information added from the Business Process Modeling Notation (BPMN). The additional information *has_subprocess* has been added for administrative purposes. The IDM reference table contains the associated identifiers of the IDM elements, including the unique identification name of the process. The database maps a tree structure, with the checking iterations representing individual branches.

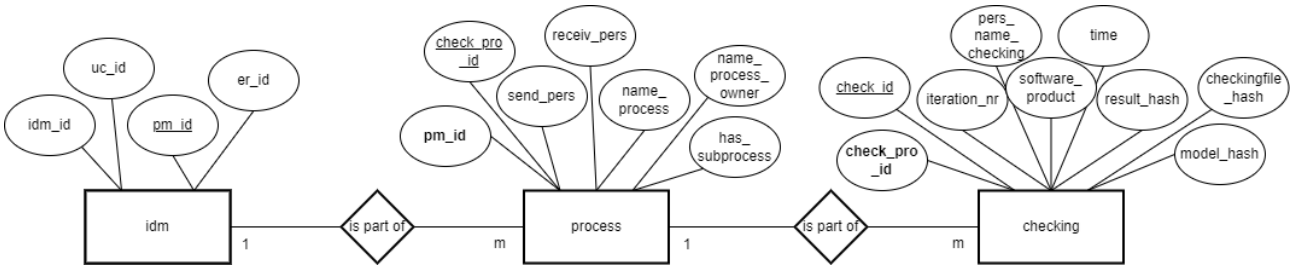


Figure 2: Structure database

## 4 Application of the database in the ILC project

In order to validate the advantages and disadvantages of the database, the processes of the use case *maintenance management lift system* of the ILC project was used. The predefined data provided by the BUW process database represents a process containing information transfer points and their specific exchange requirements. This information is stored in tabular form and is used to generate mvdXML. The structure of this table is subdivided into three sections, which are process information, object reference and property information. The individual sections are further subdivided. The process information contains the process name which describes the task, in the example used here such a task is: *create asset directory maintenance*. In addition, the process information also includes the process owner, in this example there are a total of three owners: *Client/owner, executing company, facility manager*. The object reference section, includes the IFC element name. The property information includes the name of the property set, the property, and the value definition.

A corresponding model is checked against the requirements, which are stored in the property information section of the table, using mvdXML. For each individual checking process, the required information is recorded (see figure 3). Part of the process-relevant information is taken from the BUW process database table, this includes the process name and person responsible for the process. For the IFC file and the mvdXML file, the corresponding hash values are generated during file upload and compared with the tuples already stored in the database. If it is determined that the check with these two files has already taken place in this exact way, the check can be cancelled or continued. If it is determined that this model checking process does not yet exist, the check is saved as a new or alternatively as an iteration of an already stored check. For the different states, different display formats can be realized for a visualized representation in the user interface. Whether a checking process is to be declared as a new check or as an iteration is decided on the basis of the known process data. Additionally, the comparison between the hash value of the uploaded mvdXML with the already stored hash values from the database. The process information extracted from the table is not stored in the database until the comparison of the process information and the hash values is complete. Since in this use case example only the data drop is relevant, the process name may be composed of several task names. This happens because more than one task can be assigned to a process owner before a data drop, and a new mvdXML is only generated after each change of a process owner.

Further information about the check such as check ID, number of iterations and timestamp are automatically generated and added to the database. Information such as the name of the person performing the check and the software product are recorded manually and stored in the database. A process ID is automatically generated and stored. The information whether there are sub-processes is recorded manually in the ILC example. An IDM assignment plays a less important role in this example, as only one use case was tested. Therefore, IDM-relevant IDs were assumed and entered manually. In general, all information to be entered manually was recorded through a user interface.

When a model check is performed for the first time, all information is recorded, automatically and manually. In iterative model checks, only the information concerning the check is recreated, the information about the process remains the same.

## 5 Discussion

The developed database supports the delivery processes of the ILC-system by recording data and workflow management relevant information of performed model checks in a process-oriented manner. Therefore, it presents an approach to record checks in the context of open systems over the whole process and to reconstruct checks performed when needed. It enables model checks to be reconstructed at a later point in time. Furthermore, the database supports avoiding redundancies by repeating duplicate model checks. In the context of IDMs, model checks can be assigned to individual process steps with specific meta-information. In addition, it can be traced whether, with what and by whom a model was tested at a certain point in time. However, the implementation of the database does not

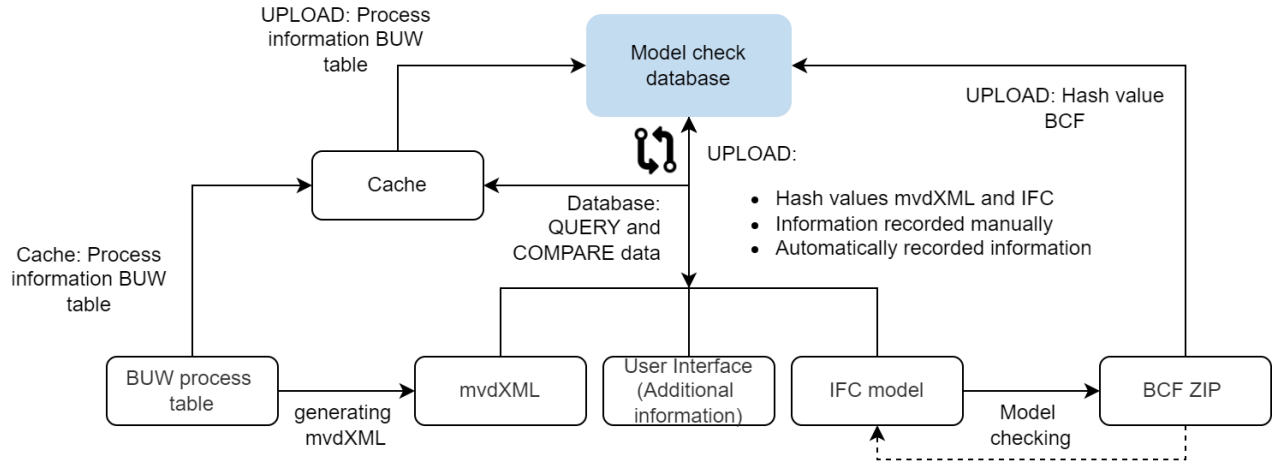


Figure 3: ILC example use case storing check information for every data drop and checking iteration in process diagram

distinguish between priorities and importance of a model check for the process flow. Model checks can also be carried out manually or semi-automatically. Within the presented use case, only automated property model checks have been recorded. In order to fully track model checks, semi-automated and manual checks must also be tracked. The database structure presented here does not necessarily have to store an IFC, mvdXML and BCF file, but has only been validated using these data files.

Since a hash value is derived from the content of the file, a new hash value will be generated if any of the file content changes. In addition to the content-dependent identification of files, a hash value requires significantly less memory than the check files used.

Potentially, a gap between data management and checking tool can be closed by recording automatic model checks. All components involved in rule-based checking are stored in addition to process-related information such as persons involved and time, as well as the associated use case. Data management and workflow management in the CDE can be enriched from this information pool and model checks can be assigned, reconstructed and retrieved.

Within the framework of the idmXML development, it will be possible to assign the definition of information delivery(s) to their delivery verification. The database developed here links to an IDM by assigning identifiers to individual components — IDM, use case, process map and exchange requirements.

## 6 Conclusion

This paper has presented one way to track automated, rule-based model checks in an information delivery process. The use case of tracking model checks is the ZukunftBau information delivery controlling project. It was shown how check-relevant files can be stored with hash values and enriched with process-relevant information. Particularly in the area of data and project management, this data

can help to reconstruct procedures in processes and to identify faulty files such as faulty control data records.

## References

[1] S. A. N. Company. “Führend bei der Qualitätskontrolle in der Bauwirtschaft”. (2022), [Online]. Available: https://www.solibri.com/de/ (visited on 05/12/2022).

[2] Thinkproject. “Desite bim”. (2022), [Online]. Available: https://thinkproject.com/de/produkte/desite-bim/ (visited on 05/23/2022).

[3] J. Oraskari. “Onlinemvdxmlchecker”. (2022), [Online]. Available: https://github.com/jyrkioraskari/OnlineMvdXMLChecker (visited on 05/12/2022).

[4] S. A. N. Company. “Unterstützte Dateiformate”. (2020), [Online]. Available: https://www.solibri.com/de/lernen/02-modelle-unterstuetzte-dateiformate (visited on 05/23/2022).

[5] BIMcollab. “Unterstützung aller openbim-workflows”. (2022), [Online]. Available: https://www.bimcollab.com/de/products/bcf-managers/workflow (visited on 05/12/2022).

[6] BIMQ. “Bimq intelligentes Informationsmanagement von bim-Projekten: Einfach, intuitiv & cloud-basiert”. (2022), [Online]. Available: https://www.bimq.de/ (visited on 05/12/2022).

[7] J. Oraskari. “Bewährte Projektdurchführung und -kontrollen.” (2022), [Online]. Available: https://www.oracle.com/de/industries/construction-engineering/aconex-project-controls/ (visited on 05/23/2022).

[8] Thinkproject. “Common data environment”. (2022), [Online]. Available: https://thinkproject.com/de/produkt-familie/common-data-environment-cde/ (visited on 05/23/2022).

[9] N. N. I. of Standards and Technology. “Hash functions”. (2022), [Online]. Available: https://csrc.nist.gov/projects/Hash-Functions (visited on 05/23/2022).

[10] A. Borrmann, M. König, C. Koch, and J. Beetz, *Building Information Modeling: Technologische Grundlagen und industrielle Praxis* (Springer-Lehrbuch). Wiesbaden: Springer, 2021, ISBN: 9783658333607.

[11] Z. Meng, N. Kremer, B. Klusmann, and J. Beetz, “Development of information delivery controlling tool based on process modeling”, in *Proc. of the Conference CIB W78*, vol. 2021, 2021, pp. 11–15.