Available online at www.sciencedirect.com

ScienceDirect

Procedia Environmental Sciences 00 (2014) 000–000

Procedia Environmental Sciences

www.elsevier.com/locate/procedia

12th International Conference on Design and Decision Support Systems in Architecture and Urban Planning, DDSS 2014

Using the BIM Collaboration Format in a server based workflow

L.A.H.M. van Berlo[a], T.F. Krijnen[b]

[a] *Netherlands organisation for applied scientific research TNO, leon.vanberlo@tno.nl, Delft, the Netherlands*
[b] *Eindhoven University of Technology, t.f.krijnen@tue.nl, Eindhoven, The Netherlands*

Abstract

This paper describes the research and development of a server based BCF workflow and open source BCF server software. BCF is short for "BIM Collaboration Format" and is an open standard to communicate about the 'issues' of a BIM model during its design cycle. Basically, a BCF issue holds a description of the issue, a status, links to a BIM model and objects, a picture of the issue and a camera orientation. The BCF standard is based on file exchange. BCF issues are packed into a ZIP file (.bcfzip) and sent to project partners. This paper, however, describes the use of the same BCF in a centralized online setting. An open source BCF server was created, and integrated with an online 3D viewer and BIMserver. Through the BCF server it has been shown that project partners have been able to create issues, manage them online and evaluate them in context of the actual BIM model. Important research questions investigated in this project included whether the BCF format is capable of describing the status of building objects in addition to merely their issues and to what extent the XML based BCF schema proved to be suitable for a more centralized online storage paradigm. A case study was conducted in order to investigate a suitable workflow for a collaborative design project that involved the online sharing of BIM data, issues, and status. This paper describes the project, the creation of the BCF server, the case study and reports on the results.

Peer-review under responsibility of the Eindhoven University of Technology, Faculty of the Built Environment, Urban Planning Group.

Keywords: BIM, BCF,BIM Collaboration Format, collaboration, issue management, open source

1. Introduction

The Architectural, Engineering and Construction (AEC) industry has a fragmented nature. Many different experts produce data (Building Information Models; BIM) that they exchange among each other. Every expert has a focus on their own work, but also on the integration with other experts involved in the project. Previous research has shown (Berlo et al, 2012) that working with different aspect models (or 'discipline models') can be more effective

Peer-review under responsibility of the Eindhoven University of Technology, Faculty of the Built Environment, Urban Planning Group.

than trying to get everybody to work on the same software platform, the latter being advocated in some of the early BIM literature. Regardless, in both situations, whether considering the model as a collection of aspect models or as being fully integrated and centralized, experts feel an increasing need to exchange additional information about the objects in a BIM. Specifically, a desire that resonates within the community, also expressed by the commercial partners that initiated this research project, is the ability to exchange information about the status of individual objects or parts of the model. For example, some scenarios highlighted by these commercial partners: suppose a building model is sent from the architect to the structural engineer. In such a scenario, the architect may want to state that everything in the model is to be considered as 'definitive', except for the part where the parking garage meets the rest of the building. Or alternatively, the architect wants to express that parts of a building being modelled should have the status 'proposal' for the structural engineer to evaluate.

Additionally, due to the growing complexity of building projects, the project client and the project manager feel an increasing need to obtain a more detailed, more accurate and real-time overview of the status of the overall project. Therefore, in addition to the status of individual objects, the project manager and client want to obtain an aggregated overview of how many objects are considered definitive, how many are still to be approved. Furthermore, they want an overview of the priority of outstanding issues and which project partners are responsible for many of the unresolved issues, etc.

These two aspects, which are currently unaddressed in the BIM practice, were the reason for the commercial partners of this project, the Dutch architectural firm ZEEP and the MEP engineering firm Quinta, to start a research project on how to address this situation.

2. Problem statement

There are two aspects to the project described in this paper. Firstly, several experts feel the need to exchange explicit information about element status along with the model delivery, in order to make their collaboration more effective. On the other hand, project managers and clients want to have more detailed insight into the status information in order to manage and steer the project more effectively. Therefore the following two research questions have been formulated:

2.1. How to facilitate the exchange of information regarding the status of BIM objects?

Exchanging BIM data between different project partners is getting more common in daily practice. However, exchanging status information about specific objects in the model is less common. The assumption being made here, is that project partners only share information when it has reached a (more or less) definitive status. When parts of the model are not finalized yet, this keeps the entire model from being handed over, because engineers will want to finish that last parts before exchanging the data. The hypothesis is that project partners will exchange their BIM data earlier on in their internal design process when it would be possible to easily add additional status information to objects in the BIM.

2.2. How to present a real-time and comprehensive overview of the project to project managers and clients?

From a project management perspective, it will be investigated whether an aggregation of all statuses of objects will give a quick overview of the overall project status. Providing this same level of insight to clients might lower the threshold for clients to actively participate in the project without the need to actually fiddle with BIM data (with all the software licenses and training involved).

In the following chapters, the search for a unified technical approach is outlined that seeks to implement a solution for both these research questions.

3. Technical options

Already several technical approaches exist that allow to associate additional information to BIM objects. In this chapter we expatiate about some of these possibilities.

3.1. Status and issue tracking by means of the IFC schema

The ambition of this pilot project is to create a central hub through which issues can be raised and comments can be made, queried and analyzed. As part of this effort, it has been investigated whether it would be possible to simply rely on the IFC model file to store its comments. After all, various tools already exist that centralize the storage of IFC files and allow them to be queried. An example of this is the BIMserver.org platform (Beetz, et al, 2011), which is used in this pilot to store the model revisions. With the comments directly embedded within the model file, this central storage and querying functionality could simply be re-used and slightly adapted to enable to it to list comments. Several ways, by which it is possible to annotate building products in the model file with their issues or status, are presented below. Nevertheless, note that the project zoomed in on BCF pretty early as it is an open standard that is currently being ratified (Aram, et al 2013).

The IFC file schema contains a central construct called the *IfcOwnerHistory*. It enables to relate, on a per-object basis, information about the *OwningUser*, or assignee of the object, and a state, a flag that pertains mostly to access rights. This in itself would be a crude way to model the status of objects, as the stakeholders in this project request a more fine-grained and extensible approach. On top of that, in the vast majority of IFC files, the owner history is not functioning as a central device to record revision history at all. In fact, in all publically available IFC models in the IfcOpenShell repository of test models [REF], which aggregates several sources of publicly available models, only 2 of the 122 contain multiple owner histories. Furthermore, within these two files, it does not associate information on responsible users to the products in the model.

Another alternative would be to model issues or comments as *IfcApproval* instances, which is the designated entity to model approval for certain parts of the design. However, the semantics of this entity are partly opposite to what the stakeholder wants to model, which would be a list of issues and associated responsible persons within the organization. Furthermore within the set of investigated test models no model at all features an *IfcApproval* instance, which would indicate that support for this construct is lacking in authoring applications.

A third approach would be to abuse the semantics of the *IfcPropertySet* construct, which is designed to associate arbitrary groups of key-value pairs with objects. These key-value pairs could be used to model the attributes of a comment, say a subject, assignee and date field. A nice consequence of this approach would be that many tools for viewing IFC files [REFS] will already be able to show these user-defined property sets in their interface. However, the discovery of elements with comments would still be very hard: perhaps the only way would be to individually click all elements and manually inspect whether they have property sets that in fact describe an issue. In addition, easily adding comments is not something that is supported by this approach, since the viewing applications, mentioned above, do not support to append entity instances to the file. Therefore this approach is also not suitable to meet the requirements from the commercial partners.

Despite of the objections mentioned above, several initiatives exist to model issues or a status description directly in the IFC file. For example the approach by (Jakob et al) [REF] to incorporate linked data from external sources into the model file. This approach can be used to model the status of BIM products using exactly the vocabularies and concepts that are already part of the universe of discourse of the project partners. On the other side of the spectrum, attempts exist to extract status information of the file as a whole or areas of it, based on quantifiable numerical properties, such as the deviation in geometric detail or amount of relations with other elements or other semantic data (Tamke et al) [REF]. Such an approach could be used to deduce which parts of the model are not fully developed yet and need additional work. This latter form of automated discovery is not considered to be part of this research project, neither are more conventional variants of this approach such as clash detection [REF], a technique that tries to find geometrically interfering objects and marks them as issues.

3.2. File based BIM Collaboration Format exchange

The BIM Collaboration Format defines an XML based schema, developed by several commercial software vendors [REF], to exchange written commentary pertaining to certain IFC file. Such a BCF file, conventionally, would be transferred by regular file sharing means.

Technically, using the BCF schema, issues are related to a set of building elements from the associated model file. This relationship is of a textual nature. The IFC *GlobalId* (a globally unique identifier as a Base64 encoded string) attribute of the building elements is used to identify an element from the IFC model. As part of the *VisualisationInfo* of the BCF structure, this *GlobalId* is referenced as a *Component* to which the issue is related. The BCF schema validity is checked by means of regular XSD string validation. Hence, an issue that refers to a non-existent component can still constitute a valid BCF file. On the other hand, in a scenario where both the building and the issue would be modelled in the same IFC file, similar as outlined above, the validation of the comments would be part of the natural semantic validation of the IFC file. Such validation checks whether all entity references can be resolved to actual instances. Likewise, for example, in a relational database approach the validity of references could be enforced by conventional integrity checking using foreign keys in a relational database management system (RDBMS). In the BCF workflow this strict validation is lacking. This becomes an increasing problem once multiple revisions of the model are to be managed and, due to the file based nature of BCF, the association between the two would not necessarily always be clear.

A conventional BCF workflow describes a file based transmission of issues, say e-mail based, and does not propose means to centrally organize the communication that pertains to a model. This makes it difficult for stakeholders to analyze the entire body of issues and get a quick overview of the whole project. Note that these comments and issues contain valuable insights into how the design team functions and about the performance of families of products and suppliers. By not relying on a central agent to process the flow of issues, this valuable information cannot be easily uncovered. Furthermore, the amount of open issues gives an indication of the progress of the design project, but without a central registration of issues, this number is not accessible. In addition, central storage allows for easy notification to project members when a new issue is reported and can it enables it to be easily delegated to the appropriate person. Therefore, such a file based approach contradicts the desire of the commercial partners to offer a centralized project management interface that communicates the state of the project as a whole to project managers and clients.

Lastly, updating the attributes of an issue is not straightforward when issues only exist in bilateral conversations, as it would imply contacting all project members to which the issue has been sent and inform them about the changes manually. Furthermore, in general, this approach of simply storing issues and comments in files and sending them is prone to error as issues are not recorded and might slip through the cracks.

4. Solution approach

Taken the considerations outlined in previous paragraphs into account, the chosen solution approach has been to develop (1) a BCF server that stores and allows to retrieve BCF issues and (2) a BCF dashboard in order to present an up-to-date issue listing to the end-user in context of the 3D BIM model and a revision timeline.

An overview of this dashboard is presented in Figure 1. This dashboard is built by re-using many components of the open source BIM ecosystem [REF]. This has enabled to build a coherent application framework with limited effort. The following existing initiatives have been re-used to build the dashboard: The BIMserver is used for project and revision management of IFC data [REF]. It exposes a standardized API that allows to retrieve project and revision information as well as a geometrical representation of the model. The representation of the model is constructed by IfcOpenShell, another constituent of the open source BIM collective, which is able to convert the many ways of describing geometry information in IFC into a unified tessellated format that is suitable for display using conventional computer graphics methods. Lastly, the web based 3D viewing component, called BIM Surfer is used, that presents to the user an interactive view of their 3D BIM model. By re-using modular open source components, the total development time for the BCF dashboard has been reduced significantly.

5. Development prototypes

An important part that constituted this development effort, has been to develop a client-side BCF implementation that communicates with the server in order to create and list issues and their comments. Because of the JavaScript Object Notation (JSON) API which is offered by the server, this has been a minimal effort. Therefore the main accomplishment, development wise, has been the integration of the various components in order to provide a cohesive communication platform to the stakeholders. The following features are examples of how the various components are integrated: The 3D view is synchronized with the issue listing so that the camera orientation is updated according to the currently selected issue. Also, the issue listing is integrated into the revision timeline. It shows a graph of the amount of issues being created using the same temporal axis as the display of revisions. This way, the user can visually correlate how the creation of revisions induces the creation of new issues.

6. Issue management processes

In order to guide the research and development of the BCF server and forum, the existing workflow for managing issues, as it is manifested within the commercial partners, has been analyzed. Furthermore, based on the initial developments, the usability of a server based approach has been validated by means of an experiment. The intention behind the project, as outlined in this paper, has been to find a solution that is appealing enough for practitioners to actually use it. Theoretical possibilities are numerous, but many ‘perfect world’ or academic solutions bring a high threshold for the current practice. Therefore the intention has been to sketch a process that adds value for the project partners in an unobtrusive manner, without disturbing existing processes or resulting in extra unnecessary tasks for project members.

6.1. Conventional BCF based issue exchange

As described, the conventional practice of ‘issue management’, when using the BCF methodology, is to create a BCF file that contains a list of issues. In the case of the project partners, this is done by a delegation from the project team during a session on Friday. During these sessions the project manager has the lead. BIM analysis software like Solibri Model Viewer [REF] of Tekla BIMSight [REF] is used to find and analyze issues of BIM data, either being a single model or a set of models. The software generates a list of issues in the BCF format. Every issue is now a ‘BCF Topic’ and the combination of several topics is put into a BCF file. The topics are discussed, filtered and the result is sent to all relevant project partners. These topics describe issues that need to be resolved by the project team and individual project partners and can be seen as a ‘to do’-list for the project partners. Sometimes the topics are transformed into a spreadsheet to facilitate project partners that do not have access to software that imports BCF files.

In subsequent sessions, a new BCF file is created with new topics. In the experiment, it turned out to be more efficient to create a new list of current issues, rather than evaluating the topics from the previous session and build from that list. Consequently, the topics from the previous session are deprecated. The drawback of this workflow is that no revision management on issues is possible. It is not possible to track which issue take a long time to solve and, more importantly, in this way, previously detected issues might slip through the cracks.

Furthermore, noteworthy is that in this situation the use of comments, as described by the BCF schema, to annotate and discuss topics, is not used at all. Rather, BCF is seen as a ‘read only’ ‘to do’-list of issues. Any discussions, related to the topics in the BCF file, were to be held during the Friday sessions.

Occasionally, the project manager would notice a repeated emergence of the same (low priority) issues. In such an event, he would initiate a ‘cleanup session’ with the modelers, in which small issues, like marginal geometry errors were to be fixed. This would stop lower priority issues from interfering with the general overview of issues of higher importance or urgency.

6.2. New process within the pilot project

The current practice of the project team has been taken as a vantage point for prototyping the server based method of issue management that has been hinted on in this paper. The regular practice of a meeting on every Friday is unaltered, as is the objective to compile a list of BCF topics. However, instead of e-mailing it to the relevant project partners, the topics were uploaded to a central BCF server, as developed for this pilot project.

One shortcoming, which stems from the BCF data model, is that, at the current state of the development, it is not possible to determine what to do with older issues associated with a revision, after a new revision is created on the model server. This stems from the fact that a BCF issue is paired with an IFC file, which is the off-line equivalent of a revision. However, in an on-line centralized environment the end-user expects a more intelligent system where issues are not automatically discarded upon each new revision. Therefore, during the experiments the team has been presented with two options: (1) automatically deprecate all the previous BCF topics or (2) leave all the previous topics open. In the second case there would have to be 'cleanup sessions' for BCF topics just like the cleanup sessions in the normal process.

The new process therefore has been fairly unaltered, had included advantages not seen in the traditional approach. Most notably, because of the server based architecture underpinning the new workflow, external parties, such as the client could get an immediate overview of the project, simply by logging into the online system. All project members, including the project manager, had more options to filter issues based on priority, date, assignment, and of course 'status', because an important contribution from the centrally accessible listing of issues has been the ability for Issues to be updated or even closed by team members, giving an up-to-date overview to the client and project manager. This last observation is exactly one of the key desires of the project partners.

Additionally, in this new process, the possibility exists for the client to create new issues from a web browser without the need to install complex software. This lowers the threshold for clients to get involved in the project in every stage of the design.

Lastly, another major accomplishment, which emanates from the new server based environment, is the opportunity to add iteratively keep adding comments to topics. This opens new doors for discussion between project partners in a collaborative setup. Creating such forum capability with optional involvement from the project manager and client is a valued addition to the current process.

6.3. Types of BCF Topic statuses

The BCF schema allows for some enumeration fields to be extended with user defined values. To adequately manage issues within a BCF environment, it is important to setup an effective set of values for these fields, tailored to the processes of the project partners. These fields include *TopicTypes*, *TopicStatus*, *TopicLabel* and *Priority*. The schema by itself already allows to extend these by editing a supplement file that comes with the standard. By means of the centralized nature of the BCF server these enumeration values can be decided upon on a per-organization or per-project level. Specifically, these values can be setup within a dedicated page on the BCF server.

The BCF standard proposes *Comment*, *Issue*, *Request*, *Solution* as the enumeration values for *TopicType*. However, in the pilot project the values *Question*, *Issue*, *Proposal*, *Correction* and – most notably – *Status* have been used. *Question* turned out to be mainly used by the client to create new topics to clarify aspects of the design. *Status* was used to facilitate the use case that is described in the problem statement of this paper, i.e. to streamline the communication between project partners and urge partners to hand over their models earlier by allowing them to state which parts of the design are not fully finalized yet. Used enumerations for *TopicStatus* were *Open*, *Closed*, *Under Consideration*, *Concept*, and *Definitive*. For *TopicLabels* it was *Architecture*, *MEP*, *Construction*, *Specs*. The BCF standard proposes the use of integers to define the *Priority* enumeration. However, the project members found it confusing whether these where in ascending or descending order in terms of their associated priority, so the set of possible values for Priority was adapted to consist of *Low*, *Moderate*, *High* and *Very high*.

7. Experiment / Pilot

Image 1 shows the BCF forum, as developed, with on the right-hand side the listing of issues and comments and functionality for their creation, on the left-hand side an interactive view of the 3D model with underneath a listing of the revisions combined with a visual indication of the creation of issues over time.

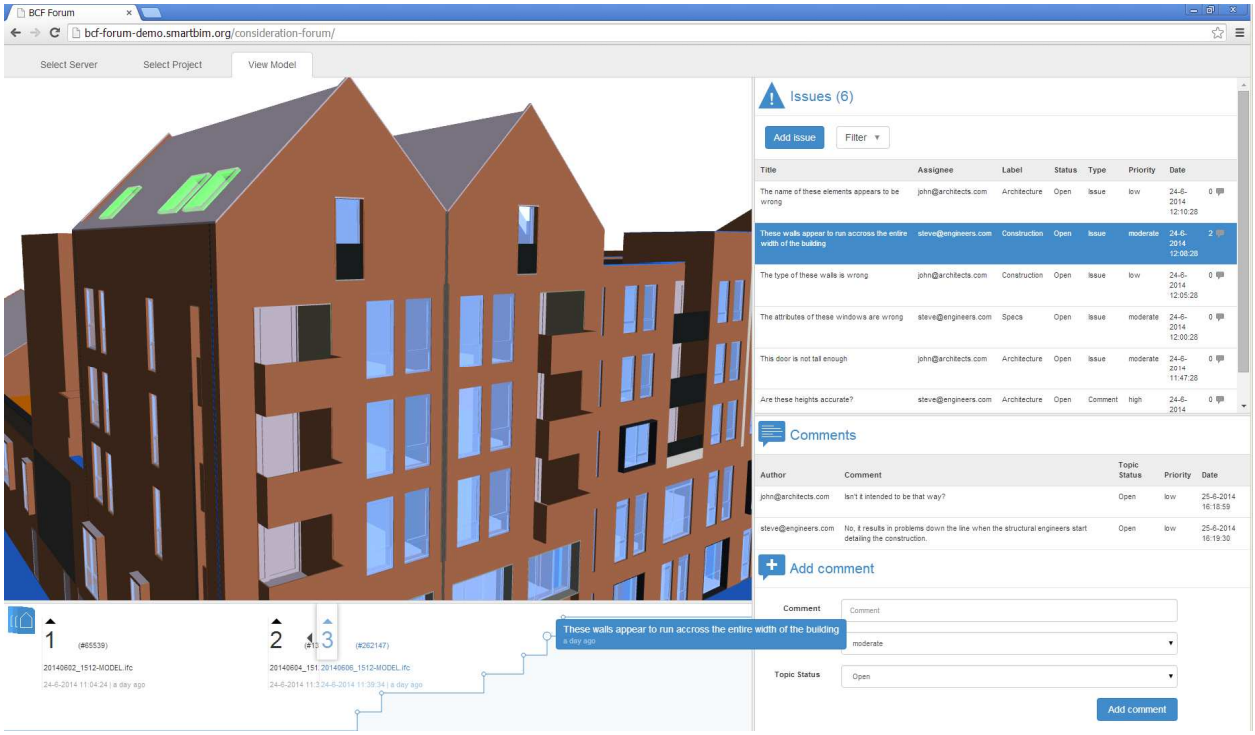

Fig. 1. (a) screenshot of the developed BCF Forum in use.

8. Observations and Conclusions

The pilot project showed more interaction between the project partners in the time between the actual project meetings. Even though the added possibility to make comments was not heavily used, as project partners preferred to talk over the phone to discuss topics and thus receive an immediate answer. The writers think this might be due to the fact that the possibility to get notified of a new comments currently does not exist. This made users unaware of new comments and thus project partners had to find other ways to indicate their remarks. This is something that can be easily addressed in a new version of the software.

The concept central server to do bookkeeping of the issues proved to be a viable solution to implement in the AEC domain. Of course, the dependence of a central location for all communication is a risk, for example in terms of availability considering unforeseen events. Since all project partners understood the experimental nature of this project most of them accepted some bugs and worked around it. The centralized nature of the system made it possible for the project manager and the client to get a quick overview of the status of the project and individual objects, as they desired. The democratic and egalitarian accessibility of the list issues (everybody could see all topics) was much appreciated by the project partners. This in contrast to individuals keeping track of a personal ‘to do’ lists in the original process. The overview of all issues was highly valued.

The experiment showed that some *TopicStatuses* were only used in combination with certain particular *TopicTypes*. For example, the ‘Definitive’ and ‘Concept’ status where only used in conjunction with the *TopicType*

‘Status’. Similarly, ‘Under consideration’ was typically paired with ‘Issue’ and ‘Proposal’. This observation makes sense and raised additional questions whether this correlation between certain *TopicStatuses* and *TopicTypes* could also be enforced or encouraged by the server, or that in fact, the combination of status and type should be some hierarchical tree of values, rather than two unrelated flat lists. This observation is something that could be submitted back as feedback to the BCF consortium members.

Overall the pilot project participants found the use of a centralized BCF server and management dashboard to be of added value to the design project.

9. Discussion and future work

The software prototypes and design experiments within this pilot project were based on the concept of a central server for storing data related to issue and status management for BIM models and elements. Given the fragmented nature of the industry, it is very hard to push project members to all work in a unified software ecosystem. Future research will have to show if a more distributed BCF topic management system with occasional synchronization could be more effective for this industry. After all, the many forms of communication around the design process can be considered sensitive, which individual firms might not want to leak to respective collaborators by default. A more decentralized approach would allow firms to isolate the internal communication and occasionally push fragments of this to external parties where needed.

As discussed, the validation of BCF issues in conjunction with the IFC file is something that has not been solved in detail. The authors propose, as a future extension to the BIMSie ecosystem [REF], a push based approach, in which a new revision on one model server could trigger events on other related servers. This way, the BCF file could be revalidated for products that potentially went missing or altered in the new revision and prompt the user, who committed the new model revision, for more information in order to update the associated issues.

Acknowledgements

This research was funded by Quinta Engineering from Utrecht, and the Netherlands organization for applied scientific research TNO, the Netherlands. The original research question came from the people at Root BV and ZEEP architects, both from Amersfoort, the Netherlands. Thanks to their efforts the developed software is published under the GNU Affero GPL open source license on http://github.com/opensourcebim/.

References

1. Berlo LAHM van, Beetz J, Bos P, Hendriks H, Tongeren RCJ van. Collaborative engineering with IFC: new insights and technology. *Proceedings of the 9th European Conference on Product and Process Modelling*; 2012

2. Beetz J, Berlo LAHM van, Laat R de, Bonsma P. Advances in the development and application of an open source model server for building information. *Proceedings of the 28th International Conference of CIB W078*, 2011

3. Aram S, Eastman C, Sacks R. A study of BIM collaboration requirements and available features in existing model collaboration systems. *Automation in Construction.* Volume 35, November 2013, Pages 1–17