A state of the art of collaborative CAD solutions

Hugo Locquet[1,2], Louis Rivest[1] and Matthieu Bricogne[2]

[1] École de technologie supérieure, Montréal, Canada
[2] Université de Technologie de Compiègne, Compiègne 60200, France

Abstract. Despite the ever-growing need for speed in conception, computer-aided design (CAD) has stayed confined to the PLM paradigm, which lets only a single user at a time access and modify a CAD document. A new type of CAD, called collaborative CAD, has emerged to overcome this conundrum. Tasks can be parallelized with much more ease when multiple users can access a document simultaneously. Two main currents exist in collaborative CAD. The first one is synchronous work, which involves multiple users working in the same document simultaneously. The second one is asynchronous work, which involves each user working in a copy of the document and then all users' work being merged at the end. In this paper, the workflows of PLM-based CAD and collaborative CAD are compared with a concrete example, the capabilities and limitations of current collaborative CAD solutions are identified and analyzed, and future perspectives regarding this work are presented.

Keywords: Computer-Aided Design, Collaborative Computer-Aided Design, Synchronous Computer-Aided Design, Asynchronous Computer-Aided Design.

1 Introduction

A new approach to product design, called collaborative computer-aided design (CAD) is emerging. It has the advantage of letting multiple designers work on a CAD document simultaneously. This speeds up the design process and makes it possible for multiple users with different competencies to contribute to the design at the same time. To make an analogy with word processing and spreadsheet software, collaborative CAD is akin to Microsoft 365 Word[1], Google Docs[2], Collabora Office[3] or Nextcloud Office[4], which allow multiple users to modify a document together in real time. To continue with the same comparison, the classic version of Microsoft Word, or Open Office, both of which allow only one user at a time to edit a document shared through SharePoint or a cloud service, would correspond to the classic way of designing a product, using a product lifecycle management (PLM) system. PLM aims to streamline the flow of

1 https://support.microsoft.com/en-us/office/collaborate-on-word-documents-with-real-time-co-authoring-7dd3040c-3f30-4fdd-bab0-8586492a1f1d, last consulted 20/02/2023.
2 https://support.google.com/docs/answer/2494822?hl=en&co=GENIE.Platform%3DAndroid, "Share & collaborate on a file with many people", last consulted 20/02/2023.
3 https://www.collaboraoffice.com/, last consulted 20/02/2023.
4 https://nextcloud.com/office/, last consulted 20/02/2023.

information about a product and its related processes throughout the product's lifecycle [1].

One of PLM's numerous tools and functionalities is product data management (PDM), whose purpose is to act as an electronic vault for all product-related documentation. In a PDM platform, a user typically needs to *check out* a document to edit it. When they are finished editing the document, they *check in* the document to return it to the vault for the next person to use. Said user is the sole editor of the document while they have it checked out. If anyone tries to modify a document while it is checked out by another user, they will see an error message. This is called *pessimistic* conflict management [2]. The way in which documentation is managed is different from one software program to the next, as it is demonstrated later. This tedious way of managing documents can induce backlogs when complex documents like CAD models are involved. Each designer must wait their turn to work on a document. The serial way of designing seems dissonant with the current era of simultaneous engineering.

Collaborative CAD, which is also known as multi-user CAD (MUCAD) or *optimistic* conflict management, has been developed to be more compatible with simultaneous engineering. Its goal is to reduce, if not render obsolete, the need to check a document into and out of a PDM platform. Not only does collaborative CAD reduce design time, but users are more satisfied performing CAD collaboratively than alone according to Zhou, Phadnis and Olechowski [3].

There are two ways of working in collaborative CAD: *synchronously* and *asynchronously*. Synchronous work means that users are working on the same CAD document together in real time, with different parameters defining each user's area of work, like users being restricted to working on different paragraphs in a collaborative word processing solution. Asynchronous work involves users each working in a copy of the CAD document and their work being incorporated in the main document at different moments using various methods.

The objective of this article is to give a precise state of the art of collaborative CAD in terms of both research and some of the current commercial software. That is done through a literature review, the reporting of the results of tests conducted on three CAD software programs – 3DEXPERIENCE, Fusion 360 and Onshape – to explore their synchronous work capabilities, and a further study and analysis of Onshape's collaboration capabilities, in that order.

2 Literature review: Advancements in collaborative CAD

By diverging from the pessimistic approach and its document unicity, collaborative CAD opens itself to conflicts between copies. Hepworth et al. established two kinds of conflicts for feature-based systems [2]: *semantic* and *syntactic*. Semantic conflicts happen when two users don't have the same understanding of the original design intent. Since each interpretation is valid, the conflict must be resolved by getting the two parties to come to an understanding. Syntactic conflicts revolve around the fact that two copies are no longer geometrically sound because a change has been made. Three types of syntactic conflicts have been identified. The first one is called *feature/self* and is the result of a feature being modified in both copies. When the copies are merged, both

versions of the feature cannot coexist as they are supposed to represent the same thing. The second is called *parent/child* and is the result of a parent feature having been changed and a child of that feature not being able to be computed anymore. The third type of syntactic conflict is called *child/child*. It is similar to the feature/self conflict but different in that the feature didn't exist prior to the work of both users. The features created are children of the same parent and cannot coexist as they respond to the same design intent, but with a different interpretation.

The main framework proposed for the simultaneous use of a CAD document by multiple users is to create an editable copy of the document for each user. In most cases, the CAD document is stored on a server, and multiple users can access it. The conflict management method and communication between the server and a user can differ from one document to the next, but the layout of the framework is often similar to what is shown in **Fig. 1**.

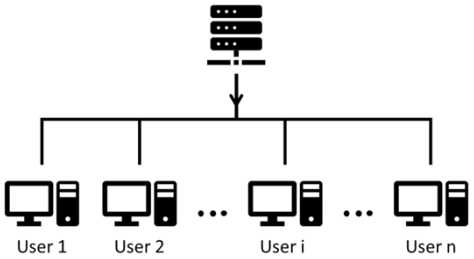

Fig. 1. Framework for multiple users to access the same document

The *first level* of conflict management in collaborative CAD proposes an approach that is neither pessimistic nor optimistic, but rather *hybrid*, and borrows from both of those approaches. It adopts the optimistic approach's ability for multiple users to access a document and the pessimistic approach's lock feature. In the hybrid approach, however, said lock exists *within* a CAD document. This method moves the collaboration limit's atomic value from the document to a semantic part of it.

Hepworth et al. proposed to use the CAD document feature as the atomic value. This means that when a user locks one or more feature(s) in the framework, this information is relayed to the server, which transmits it to the other users by preventing them from editing the locked features, as is illustrated in **Fig. 2** [2]. To make this solution more user-friendly, Moncur et al. proposed some visual improvements such as indicating on the CAD representation the areas that are locked, and a user interface that decomposes

a document into its features and reports in real time their state (locked or not) and the user with edit permission [4].

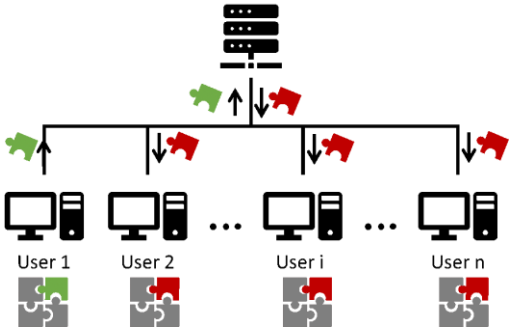

Fig. 2. Feature-level lock

Red et al. took a similar approach, but with more flexibility in terms of the atomic value. In their approach, there are three levels of decomposition, classified in order of complexity they bring [5]. The first one, at the lowest level, is feature-based. In this case, the elements of the model serve no purpose because the user determines which features to lock. The second one, at the middle level, is based on the experience and role of the user. In this case, some semantics are needed, as the location that is locked must correspond to the user. For example, if a bearing engineer opens the file, the bearing components in it are automatically locked. The third and most complex one is based on the specifications. Each part of the CAD model is decomposed and locked based on the engineering specifications. Take a simple part like a screw, for example. (It does not need multiple users to be designed, this is for illustration purposes.) Its specifications would be the type and size of head, its diameter, and the length and type of thread. The specifications could be separated into two topics: the head and the body. Hence, one user could lock all the features composing the head, while another could do the same with the body.

The *second level* of conflict management eliminates all notions of checking in and out. Instead, the communication between the server and the users increases in frequency. All copies of a CAD document are connected to the server at all times. Every modification made in one copy is carried over to the others to constantly keep all users working in the same version, as is stated by Hepworth et al. If two operations were to happen simultaneously, the first one to arrive to the server is the only one sent to update the other copies. The second operation is sent back to the user who performed it and needs to be fitted to the updated version of the document [6]. A solution to respond to this hurdle is proposed by Jing et al. They propose to store the document containing the second operation as a local copy. One of the problems encountered is the naming of the topological entities. If the new version of the document modified topological entities used in the local copy, the parent-child link would be broken. Two solutions are proposed to remedy this. The first one is to maintain a link between two topological entities with identical geometry but

different names. The second is to create a common name for topological entities that were originally one and the same, as shown in **Fig. 3**, where face F1 and edge E1 have been split into F1.1 and F1.2, and E1.1 and E1.2, respectively [7]. Cheng et al. improved upon this by identifying every operation so the metadata of the CAD document contains a history of all the operations performed. In the case of a conflict, it is transferred to a history buffer, and all the conflicting operations are tried together by an algorithm to propose a solution for the conflict [8].

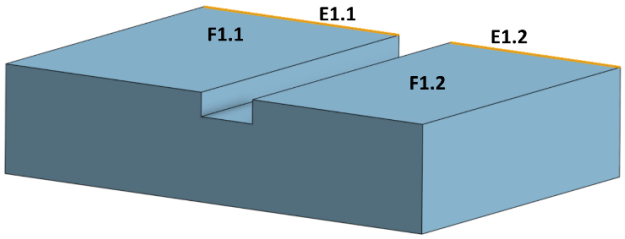

Fig. 3. Naming new topological entities based on their common ancestors

In that case, the first operation to arise would prevail. For Yu et al., however, prioritization is incomplete and should depend on the context. They propose to assign each operation a weight based on four criteria: designer authority, time order, operation type and model compatibility. The importance of these criteria can be tuned accordingly to the needs of the project [9].

All these solutions have been presented to resolve conflicts. But Hepworth et al. went one step further and proposed a solution to reduce the occurrence of conflicts. For context, they used the first level of conflict management, in which the feature is the atomic value. They incorporated two tools into a CAD software program. The first is a chat tool to facilitate communication while users modify a CAD document. The second is a task management tool whose purpose is to organize tasks beforehand. Doing so enables each user to know their own and others' areas of work and thus limits conflicts [10]. In addition, Stone et al. proposed a method to determine the optimal number of users needed to work on the same CAD model. Knowing this information would prevent a redundancy of tasks for the tool presented above. This method uses the taxonomy of the CAD document being worked on and the feature as the atomic value. The output is a tree describing the document. An optimal number of users can be determined from the shape (height and width) of the tree and the number of branching levels [11].

All these solutions have been presented in a synchronous CAD environment, but the collaborative CAD can also be performed asynchronously. While an analogy could be made between the previous methods and text editing software, the following method stems from the IT domain, like Git[5]. A lot of programming involves collaborative work on software files, and a lot of tools exist to manage this. Bricogne et al. explain this from a slightly different point of view than the IT one [12]. The tool highlighted in this

[5] https://git-scm.com/book/en/v2/Git-Branching-Basic-Branching-and-Merging, last consulted 05/05/2023.

case is branch and merge (B&M). Branch, as its name implies, creates a new branch in the CAD document's versioning tree. Merging can be done using two different methods: *2-way merge* and *3-way merge* [13]. In two-way merge, the two branches of the document are compared, and the differences highlighted. Depending on the algorithm used by the merge tool, the entirety of one document may overthrow the other, the differences may be able to be reviewed one by one, with the user choosing which operations to keep, or a rule (timeline, designer rank, etc.) may determine which operations prevail. In three-way merge, branches are compared between themselves and to the node they originate from. The merging options are similar to those available for 2-way merging.

As is explained above, collaborative CAD is split in two main categories: synchronous and asynchronous. The former is the most similar to PLM-based CAD and vastly more well-known than the latter. Tests were conducted on some CAD software programs to explore their collaboration capabilities and similarities with the literature. The testing results are reported next and bring to light another way of designing a product using currently available software with concrete examples.

3 Experimenting with collaborative CAD in synchronous and asynchronous scenarios

In this chapter, the testing methodology used is explained, and then the results of tests highlighting synchronous and asynchronous collaborative CAD are explored.

3.1 Methodology

To unravel the collaborative capabilities of the three software programs chosen, a test case was created that is as similar as possible to an industrial scenario. The software programs considered were chosen for their collaboration capabilities. Siemens's Teamcenter[6] and NX, PTC's Windchill[7] and Creo, 3DEXPERIENCE[8], and Fusion 360[9] all use a PLM-based architecture and therefore take a pessimistic approach. 3DEXPERIENCE and Fusion 360 were chosen from these programs for their integrated PDM-CAD solutions as well as their availability for testing. They are compared to Onshape, which takes an optimistic approach, as it is demonstrated below.

In the test case, independent factors have compelled two departments within the same aerospace company to modify a CAD document and two small changes made to different aspects of a part have resulted in a conflict. The first user, an engineer from the Engineering department (EE), has to locate and modify the parts that are affected by a certain issue. In this case, the wall thickness of part P and all of its series has to be

6 https://plmcoach.com/teamcenter-plm-architecture/, last consulted 20/02/2023.

7 https://www.ptc.com/en/products/windchill/architecture-and-deployment, last consulted 20/02/2023.

8 https://www.3ds.com/cloud/plm-innovation-platform, last consulted 20/02/2023.

9 https://www.autodesk.com/autodesk-university/class/Bring-It-All-Together-Fusion-Between-PDM-and-PLM-2019, last consulted 20/02/2023.

modified. EE then modifies all items in the P series to thicken their walls. In parallel, an engineer from the Manufacturing department (EM) needs to redesign some parts to fit them to new tools. Part P is one of the parts in question, as its corner and fillet radii were designed for the former tools. EM is tasked with applying the change to the whole P series. EE and EM are therefore modifying the same part simultaneously.

The test involved two users working simultaneously, and both of their screens were recorded. Since the purpose of the test was to verify the programs' collaboration capabilities, not their performance, there are no issues with it being conducted by a single pair of participants.

The test was run twice for the pessimistic programs. The first time it was run, the robust steps used in the industry were followed, meaning the part was reserved, the modifications were made, and then the part was released. The purpose of doing this was to represent this way of working and have a baseline against which to compare the optimistic approach. The second time the test was run, a collaborative CAD-type workflow was used to observe first-hand the pessimistic software's limitations. This means that the two users tried to modify the same document simultaneously, and the software's response was observed. The test was run only once for the optimistic software because the industry-standard workflow does not exist for it.

3.2 Exploration of synchronous work

Software with a pessimistic approach

The tests conducted confirm that although the workflow from one software program to another is different, the pessimistic way of working is the core of the paradigm used by 3DEXPERIENCE and Fusion 360. In the first run of the test, 3DEXPERIENCE had the following workflow: a document has to be checked out from the PDM before it can be opened in the CAD software. Once the document has been opened, modifications can be made and saved. Afterwards, the user needs to go back into the PDM to check the document back in before the next user can check it out and follow the same procedure, as **Fig. 4** shows. An explicit checkout process can be used. As for Fusion 360's workflow, the user only has to open the document and modify it. The action of checking the document out and in is implicit. It happens the moment a user creates a new feature or edits an existing one, as shown in **Fig. 4**.

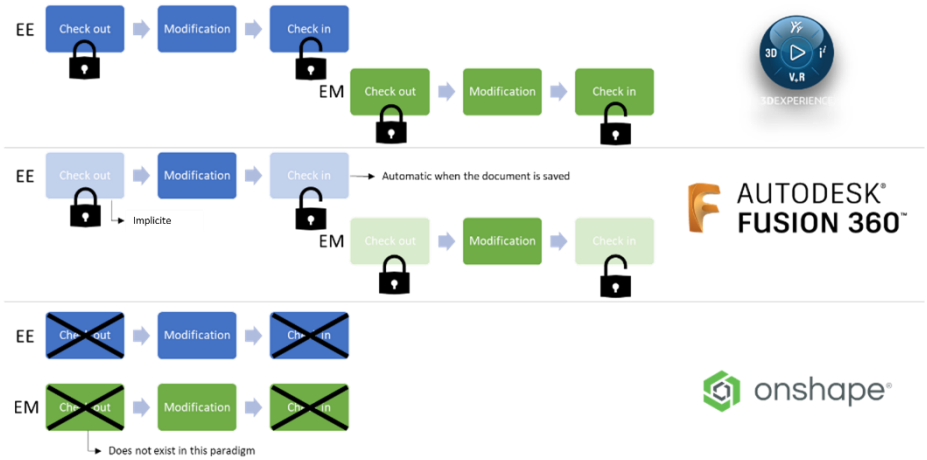

Fig. 4. Diagram of the synchronous workflow

In both cases, the document can be opened by other users simultaneously, but cannot be modified. The second run of the test, in which two users open the document and modify the same part simultaneously, highlights this. In 3DEXPERIENCE, both users could open and modify the document; however, only the first to open the document could save their changes made to the document. The other user received an error message saying that the document is already opened by another user. The only way they can save their modifications is by either creating a new version or saving the file as a new document. In both cases, the work of the two users cannot be easily combined. This happens because the check out and in steps were not properly done in accordance with the recommended guidelines. Had the users gone through the PDM, the first user would have checked out the document and the second user would have seen that the document was already checked out. By bypassing the checkout process, the software never warned the second user that their work could not be saved before they made changes and tried to save them. This result shows that the software relies heavily on its protocols and the publisher's vision of the workflow. If a single step is missed, a lot of work can be lost. This means that all users need to learn and understand the basics of PLM to efficiently use 3DEXPERIENCE. However, the program may be able to support implicit checkout with a customized configuration. The standard version was used for testing.

As for Fusion 360, the checkout process is implicit and happens directly in the CAD software instead of in a PLM platform. When the first user modifies a document, all other users are automatically locked out of the document. If a second user tries to modify the document, an error message immediately warns them that the document is already being modified by someone else. In addition, an icon appears in the CAD document indicating it is being modified. This approach lets users browse a little more freely by incorporating various discrete fail-safes. Less-thorough knowledge of PLM and version control are required to use Fusion 360.

Software with an optimistic approach

When testing Onshape, the two users could access and modify the document at the same time. Each user could see changes happen in real time as they worked. Additionally, both users could see which feature the other was working on, with an icon displayed next to the feature in the tree. The work took less time to complete as both tasks were done at the same time, as illustrated previously in **Fig. 4**. For testing purposes, both users tried to open the same feature simultaneously and modify the same value. When the first user opened the value window and modified it, the second could see it change in real time. The same thing occurred for a sketch, which has a smaller atomic value than the smallest atomic value proposed in the literature. While it was not used in this specific test, a tool does exist that lets a user adopt the point of view of any other user.

3.3 Exploration of asynchronous work

One of the software programs chosen also features asynchronous working tools and a branch-and-merge solution. As stated on its website[10], Onshape was inspired by Gitflow version management and its strong agile philosophy. It creates a lifecycle timeline for every document. Its representation is based on three core elements: a dot, which corresponds to the version; a thick line, which corresponds to the main document; and lines branching off of the thick line, which correspond to the branches, as **Fig. 5** shows. A branch stems from a version of the main document and contains all the history up to that version. All future modifications made to the main document will not impact the branch, and vice versa. A user then applies all modifications needed to the branch. Any branch can be compared with another or with the main document at any moment. Comparison is both graphic, with a 3D model representing the differences, and textual, with all the features modified or created in a branch or deleted from a branch mentioned, as illustrated in **Fig. 5**.

When the modifications done on a branch are sufficiently mature, it can be merged with another branch or the main document. The merging order is important, as the software's behavior is dependent on it. Two entities are recognized: the source, which is the branch containing the modifications, and the target, which is the line it is applied to. When a merge happens, there are three possibilities. The first is to overwrite the target with the source. The second is to combine the target and the source by either searching for a common ancestor and applying the modifications, or adding/removing features. The third is a mirror of the first possibility, as the target overwrites the source.

[10] https://learn.onshape.com/learn/article/gitflow-version-management, last consulted 20/02/2023.

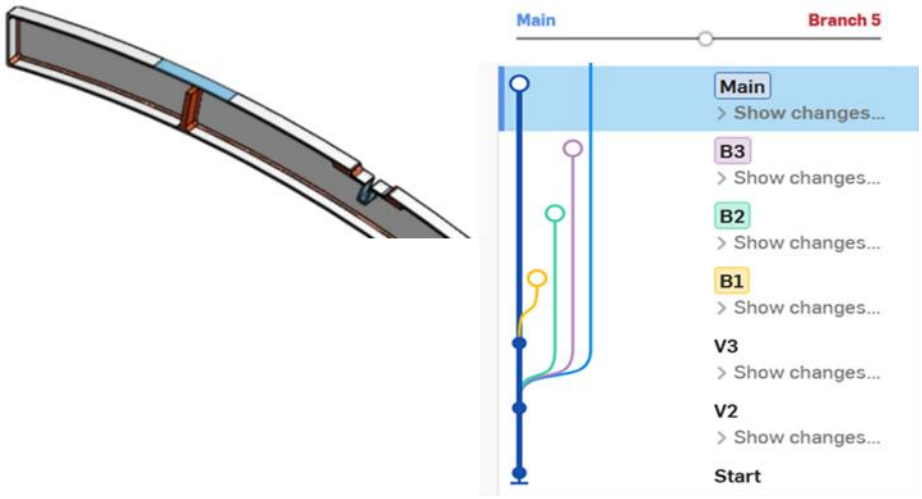

Fig. 5. Graphic representation of the differences between two branches (left); Branches of a part in Onshape (right)

The shortcomings of asynchronous work

The two main shortcomings of asynchronous work are the level of detail the compare tool is limited to an imprecise atomic value, and the absence of a link between the elements of the model and the intent of the designer, both of which are explained below.

A description of Onshape's document structure is necessary to explain these points. A document groups everything about the part/product it represents, including its features and 3D model, part assembly constraints, drawings, and materials. All of this is decomposed into tabs of different attributes. For example, there is a specific tab called Part Studio in which one or more parts can be created or modified, and another tab called Assembly where all assembly constraints and animations are created and executed. When a merge is requested, and the three choices are proposed. To make an analogy with synchronous work, the tab seems to be the atomic value Onshape uses for its merge tool. The atomic value could be reduced if the element in the tabs could be chosen like the features compared in **Fig. 5**. Additionally, when comparing a branch with another branch or the main document, all the features are listed as they appear in the history tree. If users didn't name them as they worked, a complex part may have numerous different features with generic names, which renders comparison incredibly difficult. A folder tool exists when editing a part, to be able to classify a group of features. However, the folder is not accounted for in comparison. This means any shred of intent the designer input into the document is lost. To continue with the analogy to IT, this would be equivalent to omitting all the commentary in a software file.

4 Conclusion

A lot of research has been done exploring the possible future of computer-aided design (CAD). CAD has been used for a long time in the PLM paradigm, most often in combination with a PDM platform to ensure version control is as robust and secure as possible. Most attempts to create an alternative that allows multiple users to edit a document at the same time have involved introducing an atomic value, which lessens the rigidity of PDM, but they have essentially been adaptations of the pessimistic way of approaching CAD. With its Onshape software, PTC proposes a novel solution that dissociates PDM and CAD by relying on Gitflow version management to handle the version control of CAD documents. It has created a new paradigm that supports both completely synchronous and asynchronous CAD. The testing done to compare the pessimistic and optimistic CAD software programs showed a clear difference in the workflow and the time needed to edit a document. The asynchronous branch-and-merge solution showed another possible way of collaborating, in which users can each work in a separate copy of a document and pool their efforts by merging their work afterward.

Another aspect of these last two solutions for collaborative CAD is their connectivity capacity. While a synchronous collaborative CAD solution requires a constant connection to the server and the network to interact with other users, an asynchronous CAD solution does not. Once a branch has been created, the asynchronous solution does not require connectivity for modifications, only for merging.

To conclude, three approaches to CAD are presented in this article. The first one, and the most widely used, is the PLM-based pessimistic approach. It revolves around the unicity of information and the robustness of use to the detriment of certain collaboration capabilities. The second one is the optimistic approach, which prioritizes collaboration between users even if it induces some conflicts between their work. Finally, the hybrid approach is a solution that falls in between the pessimistic and optimistic approaches. While it emphasizes collaboration, it limits itself to avoid conflicts by implementing certain sets of rules. Two ways of working – synchronously and asynchronously – are also highlighted for the optimistic and hybrid approaches.

References

[1] F. Mas, R. Arista, M. Oliva, B. Hiebert, I. Gilkerson, et J. Rios, « A Review of PLM Impact on US and EU Aerospace Industry », *Procedia Eng.*, vol. 132, p. 1053-1060, 2015, doi: 10.1016/j.proeng.2015.12.595.

[2] A. I. Hepworth, K. Tew, T. Nysetvold, M. Bennett, et C. G. Jensen, « Automated Conflict Avoidance in Multi-user CAD », *Comput.-Aided Des. Appl.*, 2014, doi: 10.1080/16864360.2014.846070.

[3] J. Zhou, V. Phadnis, et A. Olechowski, « ANALYSIS OF DESIGNER EMOTIONS IN COLLABORATIVE AND TRADITIONAL COMPUTER-AIDED DESIGN », 2019.

[4] R. A. Moncur, C. G. Jensen, C. C. Teng, et E. Red, « Data consistency and conflict avoidance in a multi-user CAx environment », *Comput.-Aided Des. Appl.*, 2013, doi: 10.3722/cadaps.2013.727-744.

[5] E. Red, F. Marshall, P. Weerakoon, et C. G. Jensen, « Considerations for Multi-User Decomposition of Design Spaces », *Comput.-Aided Des. Appl.*, vol. 10, nº 5, p. 803-815, 2013, doi: 10.3722/cadaps.2013.803-815.

[6] A. Hepworth, B. DeFigueiredo, D. Shumway, N. Fronk, et C. G. Jensen, « Semantic conflict reduction through automated feature reservation in multi-user computer-aided design », in *2014 International Conference on Collaboration Technologies and Systems (CTS)*, Minneapolis, MN, USA: IEEE, mai 2014, p. 56-63. doi: 10.1109/CTS.2014.6867542.

[7] S. xu Jing, F. zhi He, S. hung Han, X. tao Cai, et H. J. Liu, « A method for topological entity correspondence in a replicated collaborative CAD system », *Comput. Ind.*, vol. 60, nº 7, p. 467-475, sept. 2009, doi: 10.1016/j.compind.2009.02.005.

[8] Y. Cheng, F. He, Y. Wu, et D. Zhang, « Meta-operation conflict resolution for human–human interaction in collaborative feature-based CAD systems », *Clust. Comput.*, vol. 19, nº 1, p. 237-253, mars 2016, doi: 10.1007/s10586-016-0538-0.

[9] M. Yu, H. Cai, X. Ma, et L. Jiang, « Symmetry-Based Conflict Detection and Resolution Method towards Web3D-based Collaborative Design », doi: 10.3390/sym8050035.

[10] A. Hepworth, K. Halterman, B. Stone, J. Yarn, et C. G. Jensen, « An integrated task management system to reduce semantic conflicts in multi-user computer-aided design », *Concurr. Eng. Res. Appl.*, vol. 23, nº 2, p. 98-109, juin 2015, doi: 10.1177/1063293X15573595.

[11] B. Stone *et al.*, « Methods for determining the optimal number of simultaneous contributors for multi-user CAD parts », *Comput.-Aided Des. Appl.*, vol. 14, nº 5, p. 610-621, juill. 2017, doi: 10.1080/16864360.2016.1273578.

[12] M. Bricogne, L. Rivest, N. Troussier, et B. Eynard, « Concurrent versioning principles for collaboration: towards PLM for hardware and software data management », *Int. J. Prod. Lifecycle Manag.*, vol. 7, nº 1, p. 17-37, 2014, doi: 10.1504/IJPLM.2014.065457.

[13] M. Bricogne, N. Troussier, L. Rivest, et B. Eynard, « Agile Design Methods for Mechatronics System Integration », in *Product Lifecycle Management for Society*, A. Bernard, L. Rivest, et D. Dutta, Éd., in IFIP Advances in Information and Communication Technology, vol. 409. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, p. 458-470. doi: 10.1007/978-3-642-41501-2_46.