Multi-disciplinarity and Collaboration in Computational Design Teams

Livanur Erbil Altintas[1]*, Altug Kasali*[2]*, Fehmi Dogan*[3]
[1,2,3]*İzmir Institute of Technology*
[1]*livanurerbil@gmail.com,* [2]*altugkasali@iyte.edu.tr,* [3]*fehmidogan@iyte.edu.tr*

This study reports cases involving computational practices in architectural design to understand how a distributed cognitive system supports multidisciplinary collaboration in design teams. In particular, we look into the role of coding languages in collaborative practices within interdisciplinary design teams. By providing an analysis on the distributed nature of the design process, this research aims to explain collaboration involving team participants with different skills in representation.

Keywords: *Collaboration, Distributed Cognition, Computation, Multidisciplinary.*

COLLABORATION IN DESIGN PRACTICE

Design is a social phenomenon (Bucciarelli, 1988; Schön, 1991) with an emphasis on knowledge-production involving multiple actors in collaborative environments (Lyon, 2011); and it traditionally requires engagement of many experts from different disciplines (Cuff, 1992). Apart from non-designers, architects usually collaborate with other architects who have different levels of expertise in terms of approach, techniques, representational systems and design tools. Design has been recognized as an autonomous discipline (Christensen & Ball, 2019) with a certain level of diversity and multiplicity, however, some have argued that collaboration introduces problems related to issues concerning disciplinary boundaries (Cross & Cross, 1995; Cuff, 1992).

Multi-disciplinary collaboration has always been an essential feature of architectural practice and with the introduction of computation and digital technologies it has become even more so. Recently, computational technologies had a significant impact on contemporary architectural design in which designers work together with a new generation of designers or non-designers who are equipped with coding and scripting skills.

Following the evolving nature of collaboration in architectural practice, this research frames contemporary design work employing computational technologies at different levels as a distributed cognitive system (Hutchins, 1995), and focuses on the problem solving activities of teams involving individuals, artifacts, tools, representations, and other individuals. We especially have in mind Burry's (2011) categorization of collaboration between coders and designers with different level of expertise in either or both fields. Burry (2011) states that collaboration is almost a must in computational design even if the designer in a team has programming knowledge. He states that designers "probably need assistance in code writing because of the time this can take, but this necessitates a degree of handing over" (p.31).

Design is a cognitive process that involves problem solution through interaction, computation, generation, communication, synthesis, and manipulation of tasks in sophisticated contexts (Cross, 2006; Lyon, 2005, 2011). By conducting situated observations of collaborative computational practices, this research investigates how a distributed cognitive system supports multidisciplinary collaboration in design teams consisting of members with complementary expertise in design and coding. In particular, we look into the role of coding languages in maintaining collaboration between actors with different backgrounds. By focusing on the distributed nature of the system, the research aims to explain the nature of collaboration among team participants who employ different design tools and representations, and to inquire how coding and digital models mediate the interaction between professionals in practice.

REDUNDANCY OF KNOWLEDGE

In analyzing the distributed cognitive system, we rely on Hutchins' notion of redundancy in the expertise levels of team members or with multiple and almost simultaneous representation of knowledge in the system through different representational modes. Hutchins (1995) defines redundancy in the system in reference to knowledge distribution in which the knowledge domains of experts are ensured to overlap partially to achieve robustness in the system.

The notion of distributed cognition involves various components within a system in which human agents -operating with different sets of knowledge bodies- are usually assigned to a variety of tasks to demonstrate their capacities as they engage with tools and representations in situated contexts. Hutchins (1995) defines two kinds of distribution of knowledge in a system, one is overlapping knowledge distribution and the other non-overlapping knowledge distribution (Figure 1a and Figure 1b). Figure 1a represents a hierarchical system in a team in which participants possess overlapping knowledge domains. The overlapping knowledge distribution is a characteristic of cooperative works and it avoids possible errors and interruptions. In the overlapping knowledge distribution, the system is a hierarchical system in which knowledge of the experts (P1 in Figure 1a) encompasses all or partial knowledge set necessary to complete a task with all its sub-tasks; while, the knowledge of the novice is relatively limited to the sub-task to be executed (P4 in Figure 1a) (Hutchins, 1995). Thus, a socio-technic practice is established to maintain progress, avoid errors, and ensure robustness in the system, while introducing redundancy with regard to knowledge distribution.

Alternatively, a non-overlapping system is one in which all or some knowledge domains are not commonly shared by all or some team members (Figure 1b). Commonly, a non-overlapping distribution of knowledge is considered to be more effective but, it results in a less robust system since it cannot monitor itself during execution of tasks (Hutchins, 1995). In Figure 1b, the knowledge possessed by team member Px does not overlap with any of the knowledge base possessed by other team members.

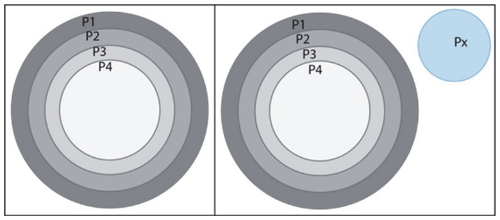

Figure 1a (left) Hutchins' overlapped distribution of knowledge.

Figure 1b (right) A non-overlapped distribution of knowledge in a design team
P: Person

In computational architectural design, the practice of coding has become a novel component of the overall design development process. In Figure 1b, a member of the design team (Px) who possesses a non-overlapping knowledge chunk, i.e., coding, enters into the equation as a person to contribute to design with a unique capacity which reduces redundancy in the system together with robustness. What is at stake in this type of collaboration is that teams have to ensure robustness in their collaborative system in which some individuals possess the knowledge of the computational tools while others might be oblivious to them and also ensure efficient communication among team members.

METHODS AND THE CASES

The research is a qualitative study which consists of ethnographic observations of two professional architectural teams (Team A, Team B) from two architectural design offices (Office A and Office B). Having integrated the digital technologies and computational tools in their design practices, the selected offices provide design services internationally.

The data collection included in-situ observations of two competition projects from beginning to end for a month. Ethnographic observations were employed to understand the nature of collective production and interactions in their everyday professional lives (Emerson et al., 1995). In tandem with our observations, we have conducted semi-structured interviews with the members of the design teams. The semi-structured interviews were face-to-face to provide a way to explore feelings, opinions, and behaviors (Sommer & Sommer, 1997).

Data analysis included the following three phases: description, analysis, and interpretation of the activities of the culture-sharing group (Creswell, 2007). In the first phase, the data was indexed in a time line to understand the phases and the overall trajectory of the teams' design processes. The set of visual and verbal data was also coded over the project timeline as sketch, photograph, field notes, meeting minutes, video records, audio records, screenshots, and e-mails. Concerning analysis, the interpretation adhered to the protocols of the Grounded Theory, involving an iterative procedure of qualitative coding to develop categories of information (Strauss & Corbin, 1990). Following our analysis, here, we present episodes from situated contexts to reflect emerging categories.

In this research, the teams under focus consisted of office leaders, team leaders, and coders (Table 1). Team A participated in an architectural competition for a municipal service building with office leader OL-A, team leader TL-A1, and coder C-A. Team B, consisting of office leader OL-B, Team leaders TL-B1 and TL-B2, and coders C-B1 and C-B2, worked on a tower building project.

Table 1
The table of the observed teams and participants.

	Case 1	**Case 2**	
Team	Team A	Team B	
Location	Office A	Office B	
Office Leader	OL-A	OL-B	
Team Leader(s)	TL-A1	TL-B1	TL-B2
Coders	C-A	C-B1, C-B2	
Design episode	Façade design	Tower design	

CASE 1: FAÇADE DESIGN

In our observations, the team leaders and the coders worked side by side in constant interaction over the course of the observed episodes in both offices.

As a vivid example to those instances, here, a detailed account of the efforts of Team A at generating the façade of the municipality service building is given. In this segment of our situated observations, the coder (C-A) is working with the Office Leader (OL-A) (Figure 2). OL-A is capable of using computational design tools and is an expert in computation. Her knowledge base and skills enable her to follow and to guide the coding phase by drawing on-paper sketches annotated with numbers, partial formulas, and various geometries to define the steps of coding for the façade design (Table 2). Her initial input broadly draws the boundaries of the coding algorithm to be used in the subsequent phases of design.

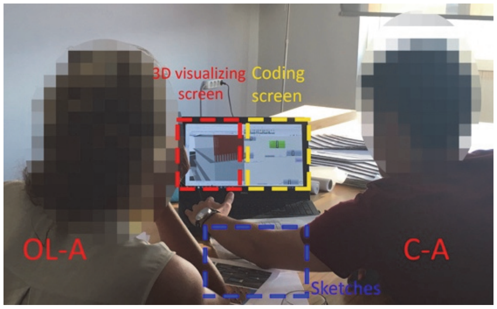

Figure 2
Team A collaborative work environment.

Along with sketches, OL-A verbally communicates her ideas supported by bodily gestures concerning the formal qualities of the facade to C-A. The intention in these conversations involving visual

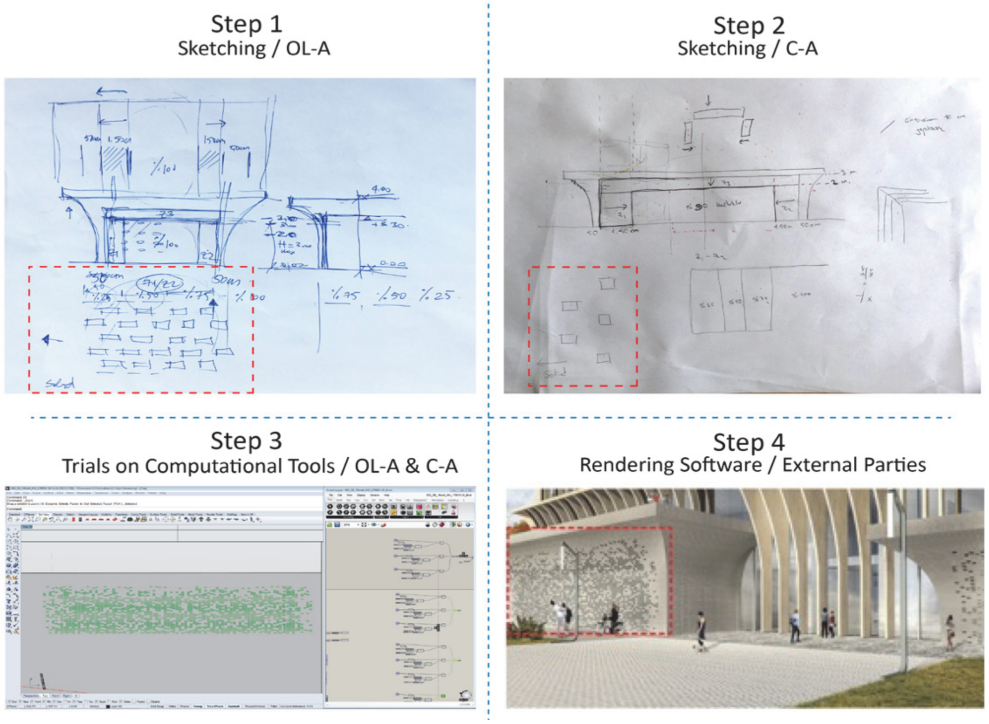
Step 1
Sketching / OL-A
Step 2
Sketching / C-A
Step 3
Trials on Computational Tools / OL-A & C-A
Step 4
Rendering Software / External Parties

Table 2
The improvement of the façade design.

representations on screen and on paper, and embodied representations are to articulate the design idea by deciding on the parameters to be coded through computational tools.

Generated earlier in the process, the visual in Table 2- Step 1 presents the sketches of OL-A with the solid and void percentages of the façade made up of bricks. The parameters considered in this episode were related to specific goals concerning: (1) providing various levels of natural light by arranging the façade composition, and (2) creating an intriguing pattern on the façade. The sketches, thus, define the distribution of solid and void surfaces across the façade. OL-A was observed to try out specific percentages to guide the perforation on the facade (Table 2 Step 1 with annotations of different percentages) which could allow varying degrees of sun light to the interior of the building. Following the sketches produced by OL-A, C-A re-sketched the design decisions on paper before creating scripts for the façade design (Table 2 Step 2). The re-sketching of OL-A's ideas on paper was mainly about focusing on how to apply this scheme into a computational design tool. The re-sketching, thus, introduces redundancy in the system by way of ensuring the accurate interpretation of the initial design decision given by the OL-A, hence establishing robustness in the system. The office leader's and the coder's overlapping knowledge domains of architecture and mathematics supported the efficient communication in the system.

Once OL-A's instructions are clearly understood by C-A, these are translated into computational codes accompanied with simultaneous digital visualizations of the façade design.

In the very first digital visualization of the façade design, OL-A assesses developing geometries of the codes created by C-A (Table 2 Step 3). During this assessment, OL-A's concern is about practical issues of construction rather than the geometric articulation of the pattern. OL-A eventually approved the codes created by C-A as they together inspected the features of the façade on the computer screen displaying both the codes and the 3D models of the design solution.

In the presented case, OL-A and C-A worked in collaboration from the very initial steps of the design process, both initially expressing their ideas in the form of sketches. OL-A and C-A had shared knowledge domains to encompass architectural design and scripting. The overlapped domains of knowledge domains of the team members provided robustness, through redundancy within the collaborative practice.

CASE 2: TOWER DESIGN

Case 2 introduces the design of tower structures for a mixed-use complex. In the initial phases of design, TL-B1 was the leader of the design team consisting of two coders (C-B1 and C-B2). TL-B1 was a passionate designer with limited knowledge in coding and computation. We mostly observed TL-B1 to convey her thoughts to the coders through talking, bodily gestures, and sketches on paper (Figure 3) –similar to the practices in Case 1- yet they lacked any computational dimension.

C-B1 and C-B2 were both separately instructed to translate these initial ideas into schemes through adaptation of available scripts together with their visualizations. TL-B1 oversaw the design process and the different alternatives produced by the coders by evaluating the visual end product without any form of reference to or assessment of the script (Figure 4). While C-B1 and C-B2 were experienced in both architecture and coding, TL-B1's practical knowledge in generating and manipulating codes was limited. Therefore, when gathered around the computer screen (Figure 3) the team's progress mostly relied on TL-B1's assessment of the formal qualities of the tower design which were shaped by the intentions verbally introduced by her. Accordingly, the coders' task involved a translation of design decisions into a code. As a routine practice in the office, Team B's office leader OL-B checks in with the team members in the afternoons, and evaluates what is on the computer screen. In the segment presented here, OL-B was only checking the end of the day outcomes so, he was not involved in each and every step of the coding and visualizing. OL-B had some understanding of how coding comes into play in architectural design but had no experience in practical means of creating codes and scripts. As a consequence, OL-B had to develop alternative methods to follow and guide the design phase by way of assessing the design schemes on the computer screen or on print-outs. At one of these segments, OL-B, frustrated with the design alternatives presented to him, expressed that he was unable to follow the design rationale behind these schemes and how they evolved into their final forms. OL-B stated as follows:

> 0:00:52 OL-B: … show the evolution of it! you should say "here it is!", then you jump to this, then you jump to that, then you jump to that. There is no such thing here, so the cause-effect relationship is broken!

Realizing the lack of clear direction in the coding and consequent arbitrariness in the design schemes (Figure 4), OL-B decided to change the team leader and continue with TL-B2, who has some coding experience if not total expertise, together with the same coders. TL-B2, being not a coder himself, ran the design process by using alternative communication methods such as sketching, referring to precedents, and, again, gestures to imitate formal qualities of surfaces or solids that shape towers. (Figure 5). However, differently from TL-B1, TL-B2 requested every alternative produced in the coding phase to be recorded as screenshots of 3D visualizations, which provided a visual narrative to explain the formal evolution of the towers. In this

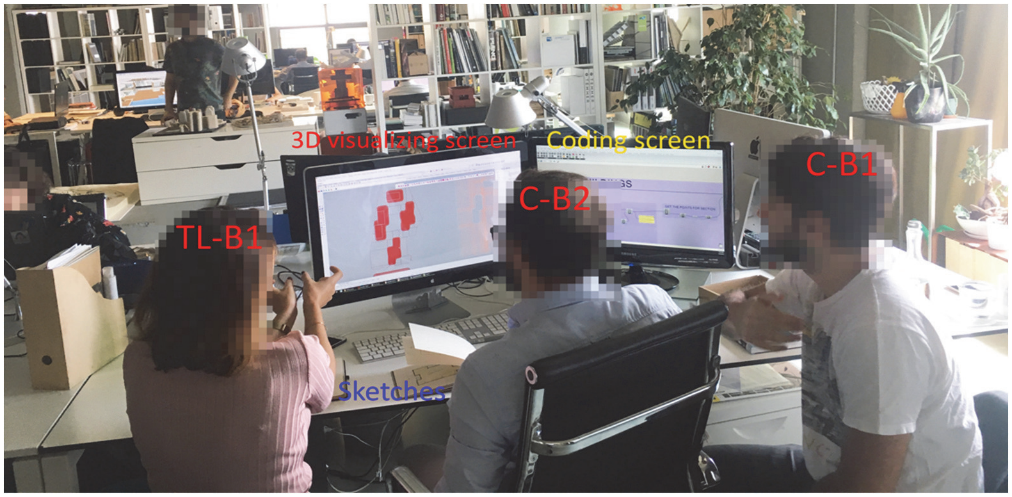

Figure 3
Team B1's collaborative work environment. TL-B1 is using gestures and body language to the coders while working collaboratively on the design project.

way, TL-B2 made sure that he and OL-B were both able to be involved in and be informed about the design process. TL-B2 and OL-B were able to provide feedback on different design schemes, while the coders were able to track the implications of their feedback with regard to related coding segments. Initially, TL-B2 and the coders collaboratively worked on finding a solution for the tower design.

TL-B2's efforts were concentrated on generating a design scheme that would be both appealing and understandable to OL-B. In order to achieve this goal, the coders were trying to figure out how to translate these schemes to coding by using computational tools.

Consequently, C-B1 and C-B2 decided to work in collaboration rather than producing different

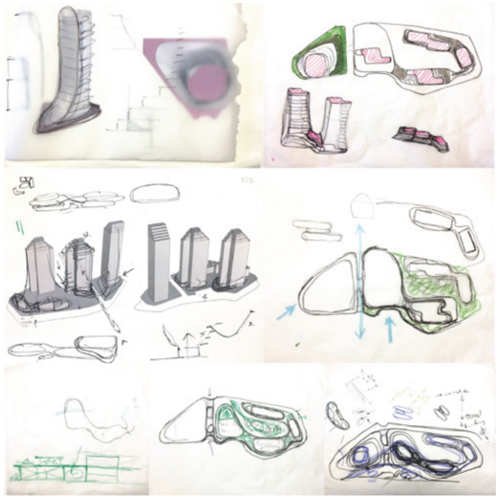

Figure 4
TL-B1's sketches

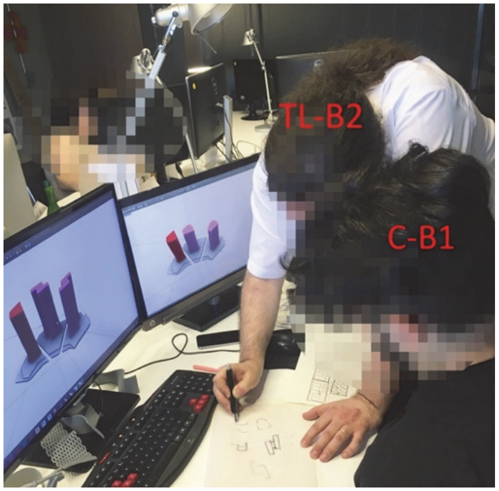

Figure 5
TL-B2 is expressing his ideas to the coder through the sketches in front of the computer.

alternatives individually. C-B1 developed a generic code to guide the formal qualities of the tower following the indirect instructions hidden in the precedents consulted, in the sketches and verbal descriptions of TL-B2; while C-B2 was responsible to adapt and apply that generic code to different levels of the towers, which were constantly changing from floor to floor (Figure 5).

The presented episodes at Office B involved two different team leaders who collaborated with the same coders and the office leader. TL-B1 was replaced with TL-B2 by OL-B because of arbitrariness in design decisions which failed to visually express the formal evolutions of the towers. TL-B2 overcame this problem by keeping a record of steps introduced into the ongoing design work. The series of design moves were recorded as screenshots; which created an external memory in the immediate design environment. These visuals, then, helped the design team, including individuals operating on different domains of knowledge, to negotiate and to progress in a collaborative fashion. The non-overlapped knowledge domains of the team leaders and the coders were mediated through alternative ways of knowledge sharing including sketching on paper, verbal instructions, bodily gestures, and recording the visuals to present interstitial steps within the evolution of towers.

DISCUSSIONS

This research presented episodes from two different offices with teams employing computational design tools in a multi-disciplinary environment.

In Case 1, we focused on two participants of Team A undertaking a façade design, i.e., the office leader (OL-A) and the coder (C-A1). In the first stages of the design, OL-A produced a series of instructions that verbally and visually described the problem and the solution. C-A1 re-examined the given instructions in visual and numerical explorations. Having a good enough understanding of OL-A's instructions, C-A1 re-represented the idea through computational design tools. Following the developing geometries of the façade, OL-A and C-A1 communicated by way of looking at the same computer screen which simultaneously represented the visual images of the design schemes and the related codes.

In the second case, we focused on five members of the team. TL-B1 led the coders, C-B1 and C-B2, for a short period in the design process. TL-B1 aimed to guide the coders with sketches, verbal explanations and gestures. After four days of intense work, OL-B found insufficient progress in the project and decided to assign a new team leader, TL-B2, who was somewhat familiar with coding. In addition to sketching, verbal explanations, and gestures, TL-B2 preferred to explain his ideas with visual precedents. In Team B, two coders worked collaboratively in writing a code to generate design alternatives. OL-B was generally checking the progress at the end of the day and usually was missing the part of the explorations in the computational tools. So, OL-B requested each alternative on 3D view screen to be saved as screenshots. As these interstitial steps come into being as external representations, the team, then, was able to observe the evolution of design and progress.

Multi-disciplinarity in the teams in both cases was supported by team members rather than incorporating external parties. Architects in the teams undertook a double role as scriptwriters/coders and designers, whom Burry (2011) calls '*designerscripters*'. As D'souza (2020) argues contemporary designer needs to be a '*multi-skilled designer*', stating that only being an design expert may not be enough. A designer needs to be 'skilled' in many issues including in collaborating or in adapting remote disciplines into design. In our cases, designer-coders in the teams needed to have an understanding of their team leaders' and their office leaders' design approaches and needed to find a way to translate and sustain their design approaches via computational design tools.

The observed teams' participants initially used 'paper-based sketches' to propagate the knowledge among the actors in the team. In such collaborative environments, drawings with definite details are considered as explicit modes of representations that partially eliminate interpretation compared to other forms of coordination (Nomura & Hutchins, 2006). To achieve coordination, in the Team A, OL-A preferred to produce relatively more explicit detailed drawings to guide the work of coders.

In our observations the teams mainly supported their imagination with computer visualizations on screens, sketching on paper, talking, referring to precedents, and gestures. Thus, the knowledge was propagated in the distributed cognitive system via multiple modes that ensured robustness through redundancy.

In both cases, transfer of ideas was mostly unidirectional; from a senior designer to a junior designer with coding skills. The coders, then, followed alternative ways to transfer and represent the idea through computational design tools.

Having coding skills, OL-A was able to communicate her intentions about design directly, which helped the team to arrive at a solution quickly. Concerning coding, the overlapping knowledge domains possessed by both OL-A and C-A1 ensured robustness in the system (Figure 6a). In the second case, the knowledge overlapping situation was different (Figure 6b). OL-B, who did not have coding skills, used other ways of conveying his ideas, which created more ambiguity in pursing acceptable design solutions. Thus, Team B produced more alternatives and their search space was enlarged unnecessarily. In comparison, the first case, where the robustness in the distributed system was ensured, the solution was reached quickly, but it was not possible to explore possible alternatives because the search space remained limited.

Talking about a hierarchy among undiscovered design solutions, Woodbury and Burrow (2003) state that 'effort' and 'connectivity' are important in terms of accessibility to design solutions. In our cases, OL-A streamlined C-A1's efforts in exploring design solutions by providing clear instructions via her sketches with numeric expressions and visuals, which were already implicitly suggesting a rule-based exploration. OL-A did not create scripts herself, but she explained the governing logic hence, C-A1 used visual images and numeric values to represent the solutions in digital media. According to Burry (2011), this points to an inevitable handing over of some responsibilities to juniors who are more up to date with current advancements in computational tools and techniques. However, OL-B gave instructions in sketches and in reference to precedents that could potentially help the coders; but, these inputs were only visual rather than exact parameters to be translated into computational tools.

In the sketching process between OL-A and C-A1, the idea was double-checked and transferred to the computational tool. In this way, the possible error margin is minimized. A similar situation may have occurred between C-B1 and C-B2. While adapting the key solution developed by C-B1, C-B2 may have eliminated possible problems and errors while applying to the facades of the towers. Contrary to Team A, Team B office leader (OL-B) intended to follow closely every alternative that the coders produced in their explorations.

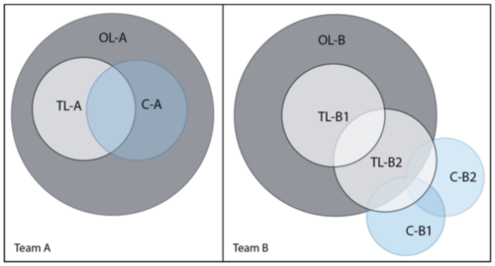

Figure 6a (left) Overlapping knowledge distribution of Team A

Figure 6b (right) Non-overlapping knowledge distribution of Team B

CONCLUSIONS

McComb & Jablokow (2022) state that the collaboration of different disciplines generate a new disciplinary understanding, and could be consolidated with a lingua franca. Our cases suggest that not only the terminology, but the skills and capacities activated within collaborative computational practices are necessary to facilitate a sphere to translate design intentions into representations of design. In our cases, the coders who were also junior designers acted as '*mediators*' between the office and team leaders and computational design tools.

Computational design tools and especially parametric modeling tools such as Rhino and Grasshopper facilitated cooperation by way of representing the design information on two different screens; one representing codes and the other 3D view screen. The simultaneous use of two screens made the collaboration between the coders and the team leaders simpler, by creating a medium to merge knowledge from different domains.

The multiplicity of interfaces on computer screens –to reflect both codes and the architectural qualities – and on paper in different sketches, and in different modalities allowed participants to negotiate and progress. Often these different versions of representations denoted the same content ensuring again redundancy in the system.

The research offers instances in which design teams leaders coordinated and developed the architectural design proposals in tandem with individuals with extended skills in computing. While the developing common language helps achieving the consensus and progress in design, the redundancy of overlapped knowledge domains (Hutchins, 1995) was observed to be critical in creating a robust collaborative practice to create multiple alternatives.

ACKNOWLEDGEMENTS

We would like to express our thanks to Melike Alltınisik and Gokhan Avcioglu who accepted our request to conduct research at their offices Melike Alltınisik Architects (MAA) and Global Architectural Development (GAD). We are also grateful to the design team members who agreed to participate in this study.

REFERENCES

Bucciarelli, L. L. (1988). An Ethnographic perspective on Engineering Design. *Design Studies*, *9*(3), pp.159-168.

Burry, M. (2011). Cultural Defence. In *Scripting Cultures: Architectural Design and Programming*, UK: Wiley Academy, pp. 27-71.

Christensen, B. T., & Ball, L. J. (2019). Building a discipline: Indicators of expansion, integration and consolidation in design research across four decades. *Design Studies*, *65*, pp. 18-34.

Creswell, J. (2007). *Qualitative Inquiry and Research Design: Choosing Among Five Approaches*. Sage.

Cross, N. (2006). Understanding Design Cognition. In N. Cross (Ed.), *Designerly Ways of Knowing*, pp. 77-93.

Cross, N., & Cross, A. C. (1995). Observations of teamwork and social processes in design. *Design Studies*, *16*(2), pp. 143-170.

Cuff, D. (1992). *Architecture: The Story of Practice*. MIT Press.

D'souza, N. (2020). *The Multi-Skilled Designer: A Cognitive Foundation for Inclusive Architectural Thinking*. Routledge.

Emerson, R. M., Fretz, R. I., & Shaw, L. L. (1995). *Writing Ethnographic Fieldnotes*. The University Of Chicago Press.

Gabriel, G. C. (2000). *Computer Mediated Collaborative Design In Architecture: The Effects Of Communication Channels On Collaborative Design Communiation* University of Sydney]. Sydney.

Hutchins, E. (1995). *Cognition in the Wild*. MIT press.

Lyon, E. (2005). Autopoiesis and Digital Design Theory: CAD Systems as Cognitive Instruments. *International Journal of Architectural Computing*, *pp. 317-334*.

Lyon, E. (2011). Emergence and Convergence of Knowledge in Building Production: Knowledge-Based Design and Digital Manufacturing. In T. Kocatürk & B. Medjdoub (Eds.), *Distributed Intelligence in Design* (pp. 71-98). Wiley-Blackwell.

McComb, C., & Jablokow, K. (2022). A conceptual framework for multidisciplinary design research with example application to agent-based modeling. *Design Studies*, *78*(C).

Nomura, S., & Hutchins, E. (2006). Study for Bridging between Paper and Digital Representations in the Flight Deck. *Computer Supported Cooperative Work (CSCW)*.

Schön, D. A. (1991). *The Reflective Practitioner: How Professionals Think in Action*. Ashgate Publishing Limited.

Sommer, B., & Sommer, R. (1997). Interviews. In B. Sommer & R. Sommer (Eds.), *A Practical Guide to Behavioral Research: Tools and Techniques* (Fourth ed.). Oxford University Press.

Strauss, A., & Corbin, J. (1990). *Basics of qualitative research: Grounded theory procedures and techniques*. Sage.

Woodbury, R., & Burrow, A. (2003). Notes on the Structure of Design space. *International Journal of Architectural Computing*, *1*(4), pp. 517-532.