JVLC

Journal of Visual Language and Computing

Volume 2019, Number 1

Journal of Visual Language and Computing

journal homepage: www.ksiresearch.org/jvlc

CROSSIDE: A Design Space for Characterizing Cross-Surface Collaboration by Sketching

Jorge-Luis Pérez-Medina[a,*], Jean Vanderdonckt[b,**] and Santiago Villarreal-Narvaez[b]

[a]Intelligent & Interactive Systems Lab (SI^2 Lab), Sede Queri, Av. De los Granados, Universidad de Las Américas (UDLA), Quito 170504, Ecuador
[b]LouRIM Institute, Place des Doyens, 1, Université catholique de Louvain (UCL), B-1348 Louvain-la-Neuve, Belgium

ARTICLE INFO

Article History:
Submitted 4.8.2019
Revised 4.30.2019
Second Revision 5.20.2019
Accepted 8.15.2019

Keywords:
Design methodology
Displays
Gesture recognition
Graphical User Interfaces
Interactive systems
Pervasive computing
Ubiquitous computing

ABSTRACT

This paper introduces, motivates, defines, and exemplifies CROSSIDE, a design space for representing capabilities of a software for collaborative sketching in a cross-surface setting, i.e., when stakeholders are interacting with and across multiple interaction surfaces, ranging from low-end devices such as smartwatches, mobile phones to high-end devices like wall displays. By determining the greatest common denominator in terms of system properties between forty-one references, the design space is structured according to seven dimensions: user configurations, surface configurations, input interaction techniques, work methods, tangibility, and device configurations. This design space is aimed at satisfying three virtues: *descriptive* (i.e., the ability to systematically describe any particular work in cross-surface interaction by sketching), *comparative* (i.e., the ability to consistently compare two or more works belonging to this area), and *generative* (i.e., the ability to generate new ideas by identifying potentially interesting, under covered areas). A radar diagram graphically depicts the design space for these three virtues to enable a visual representation of one or more instances.

1. Introduction

In many domains of human activity, across them [1] and within [2], *sketching* is largely used as a quick, efficient, and cost-effective tool for expressing ideas, illustrating design concepts and solutions as well as for sharing them with diverse people [3]. During the early stages [4] of product development until its final completion, sketching immediately plays some role when co-design is a must [5]. Sketching facilitates the sharing of abstract ideas and the insights inside a team while offering a common ground for the discussion, especially when the participants come from different cultural and social backgrounds. Sketching usually uses basic drawings to foster everyone's participation [6].

Sketching, as one particular form of drawing, belongs to the first human intellectual skills and abilities that are acquired even before speaking, writing, and precise drawing. By one year, infants understand that a sequence of sounds form a word that symbolically represents an action, a relation, or an object they can point to rather than reaching it. Our human ability to perceive and to recognize elements in a graphical representation is in itself a statement of the expression power yielded by such form of expression.

A *sketch*, coming from the Greek word $\sigma\chi\epsilon\delta\iota o\zeta$ (schedios - *done ex tempore*), consists in a quick provisional freehand drawing executed without any constraint in any medium. A sketch may convey a message, record or develop an abstraction or may be used as a mean for explaining something, for example an image. A sketch is considered as a very quick drawing aimed at communicating a message, which can be understood, misinterpreted, or even ignored. But there is always a message that is different from the one of a *drawing*. For these reasons, within many professions such as industrial and architecture design [5], representing actions, ob-

*Corresponding author
**Principal corresponding author
jorge.perez.medina@udla.edu.ec (Jorge-Luis Pérez-Medina); jean.vanderdonckt@uclouvain.be (Jean Vanderdonckt); santiago.villarreal@uclouvain.be (Santiago Villarreal-Narvaez)
http://investigacion.udla.edu.ec/udla_teams/jorge-perez (Jorge-Luis Pérez-Medina); https://www.uclouvain.be/jean.vanderdonckt (Jean Vanderdonckt); https://uclouvain.be/fr/santiago.villarreal (Santiago Villarreal-Narvaez)
ORCID(s): 0000-0003-4864-0480 (Jorge-Luis Pérez-Medina); 0000-0003-3275-3333 (Jean Vanderdonckt); 0000-0001-7195-1637 (Santiago Villarreal-Narvaez)

DOI reference number: 10-18293/JVLC2019-N1-016

jects, and their relations using different forms of sketches such as drafts, blueprints, and prototypes, has always been very important. They make visible the stakeholder's contribution, which could be different than expected.

The audience of stakeholders involved in sketching activities is now wider. Consequently, the pool of software tools to support collaborative sketching, both commercial and from the academia, also becomes more filled. *How much sketching activities could be supported by collaborative tools* is therefore a key question addressed in this paper. Researchers should be informed about capabilities of existing tools to update their research agenda and to better understand similarities and differences. Practitioners should also be informed about which tool would match their requirements for conducting collaborative sketching.

The remainder of this paper is structured as follows: Section 2 reviews sketching across disciplines involving some collaboration and discusses selected works; based on this, Section 3 introduces, motivates, and defines CROSSIDE, a design space for characterising capabilities of a software for collaborative sketching; Section 4 exemplifies some instantiations of this design space; Section 5 concludes this paper by explaining how this design space could be systematically used to describe, compare, and invent tools for collaborative sketching on multiple surfaces of interaction.

2. Related Work

In Information and Communications Technologies (ICT) as well as in computer and software systems engineering, a significant amount of resources is devoted to designing a concept, a service, a solution which could be later on revealed as inadequate. This could happen when the functional requirements are not satisfied, when the user experience is not met, or when the concept is simply not technologically feasible or too expensive. When such as mismatch is discovered late in the development life cycle or after the deployment, adapting what is required to be changed represents a high cost.

Early sketching in the development life cycle helps creating alternative solutions and comparing them. It helps to ensure that the right approach to design the right solution is put in place. It keeps the design and development right. With a sketch, designers and other stakeholders are able to identify figures, arrows, symbols and other elements that were deliberately chosen by a computer actor to communicate with stakeholders, to illustrate the requirements and share design ideas. It is more efficient and effective than any textual, graphical or formal specifications.

By using sketches, designers become more motivated, more creative, and perhaps more able to address the challenges of creating a successful design, and thus to produce a better design outcome [7]. By definition, prototypes are scaled-down versions of what will be built. Designers use them because they are faster and cheaper to create than the final blueprints. Sketching is a quick provisional drawing, it therefore matches the requirements of prototyping [3].

In Human-Computer Interaction (HCI) design, Collaborative User-Centered Design (CUCD) process suggests using sketch to understanding and designing all the aspects of a user interface (UI) design and for getting involved a wide community of stakeholders, such as, but not limited to: user researcher, information architect, user interface interaction designers, user testing specialists, software developers, marketing personnel and software products leaders. Early sketches inform the development of User interface conceptual, interaction style, and even other related material such training resources, support services, online help, etc. UI usability can be seen as a design problem of "wicked nature", as a problem to be solved – the more you try to solve the problem, the more you discover the complexity of the UI usability and more you are able to suggest solutions in the early design phase of the CUCD life cycle. This is because usually original usability problem has implications/consequences that cannot be known in advance. Sketches have been shown very powerful to build a consensus and a trade-off when usability problems are conflicting with other major quality factors, such as security. A UI sketch reveals how much a system could be usable, secure.

Sketching is the practice of drawing a rough outline or rough draft version of a final piece of art. Sketching is an aid to thought. Sketches are used as a mean of designing. Design by sketching has its foundations on the participatory design approach [8] in which a person not trained, qualified or experienced is an active and essential participant in the design process. As a communication tool, sketching can be used as a way of graphically specifying abstract ideas. It is a message from the designer to stakeholders. It can be understood by the receiver, misinterpreted, or ignored, but there is definitely a message. This message must be validated when there is a consensus to be achieved between the designer and someone else, for which designers often use limited or scaled versions of what is being designed.

A common ground to disciplines relying on is that stakeholders, particularly designers and end-users, whatever their background and skills in sketching and design are, feel actively engaged [9]. End-users help in materializing some requirements such as usability [10]. Designers then annotate original sketches introduce by end-users, add illustrations, and further develop them. They usually proceed by iteratively sketching a concept at different levels of abstraction [11]. Ambler [12] defines UI Prototyping as an iterative analytical technique in which end users are actively involved, namely by providing feedback since the early development stages and continuously afterwards.

Sketching covers many domains of human activity and inside these domains, there are several works exploiting some form sketching for one or many sub-activities such as for example: problem analysis in general [13], computer science [2] (e.g., user experience support [14, 15], user interface design, prototyping, and recognition [16, 17, 18, 19, 20, 21, 22], cross-device UI design [4], user-centered design in agile projects [23, 24], system walkthrough [25]), system development (e.g., QUILL [26] for model-based design of web applications), flexible modelling [27] (e.g., FlexiSketch [28, 29]

for model sketching), RAPIDO [30] for web API development, sketching UML models (e.g., TAHUTI for sketching UML Class diagrams [31] and SketchML for various UML diagrams [20]), distributed software design [32, 33], task modelling [34], notation creation [29]), computer-supported collaborative work [35] (e.g., stakeholders' meetings [36], collocated tables for meetings [37] and interactive design spaces [38]), product and service design (e.g., sketching in design [35], extreme designing [39], industrial design [5], shape-changing products [40]), public displays [41], learning (e.g., classroom design studio [42], teaching geometry [43]), ideation [44] and concept generation [7], knowledge design, capture, and sharing [1], design in any area of engineering [7] (e.g., knot diagramming [45]). From these references, we can observe that a significant amount of work has been devoted to using sketching as a way to support collaboration among stakeholders during the software development life cycle, starting from requirements engineering to detailed design. It is particularly useful for those stages involving some form of graphical representation of artifacts, whether they are informal (as a screen shot or wireframe) or formal (e.g., a UML model). Many techniques have been successfully reported for expressing sketch grammars [6], with the need to take into account the context of use (i.e., the user, the devices/platforms, and the environment) [46] to get context-aware sketching [43].

3. A Design Space for Cross-Surface Collaboration by Sketching

We determined the greatest common denominator in terms of properties between the 41 aforementioned references, independently of their domain, provided that collaborative sketching is involved to some extent. This identification resulted into CROSSIDE, a design space expressing collaborative sketching according seven dimensions represented clockwise in Fig. 1: user configurations, surface configurations, input interaction techniques, work methods, tangibility, layout, and device configurations. Each dimension is organized according to a progressive degree of sophistication: each step starts from the simplest value found in the literature until the most sophisticated degree.

We considered this representation as adequate to satisfy three virtues that are considered important to characterize interaction as a model [47]: *descriptive* (the design space should be able to describe any work on collaborative sketching based on these seven dimensions), *comparative* (the design space should be able to compare two or more works on collaborative sketching to identify their similarities and differences and foster consistency based on the previous description) and *generative* (once a comparison is performed, the design space should be able to identify undercovered areas and generate new and interesting ideas, configurations).

These seven dimensions are not intended to be completely independent of each other. Rather, they are aimed at serving these three virtues. Moreover, a radar chart can be effectively used to graphically represent the values along these

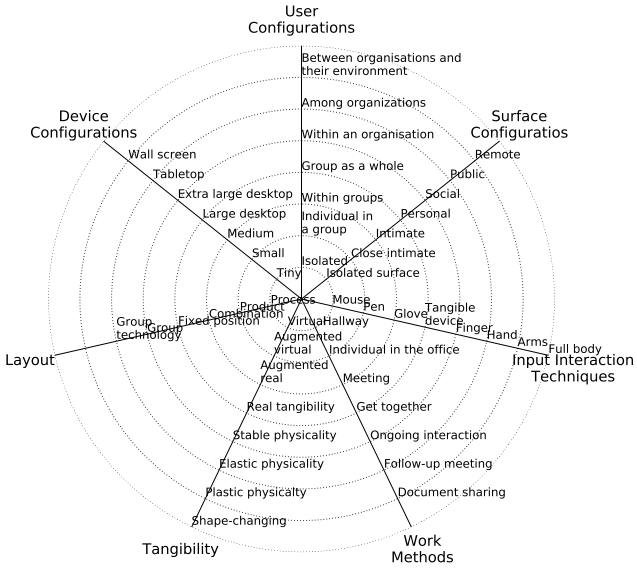

Figure 1: A design space for Cross-Surface Collaboration by Sketching.

dimensions by representing them on axes starting from the same origin. The steps on each dimension are not intended to be aligned or to be corresponding by concentric level. Therefore, steps joined on a same circle are not necessarily dependent, they only represent some progression.

3.1. User Configurations

The *user configurations* provide multiple ways to carry out a task by different stakeholders. Since many people could be located in different places, possibly at different levels of various organisations, the design space should consider the group configuration. Fig. 2a depicts various configurations among stakeholders involved in a distributed task [9]: *individual* (one task is carried out by one person in a group), *within groups* (one task is distributed across persons of a same group in the organisational structure), *group as a whole* (one task is carried out by one group of persons, independently of its internal organisation), *among groups* (one task is passed from one group to another), *within organisation* (one task is distributed across entities of the organisational structure), *among organisations* (when one task is distributed across several different organisations, all having their own internal structure), and *between organisations and their environment* (when tasks are exchanged between organisations and their common environment). Collaborative sketching requires stakeholders within groups because sketching tasks are distributed across people of a same group of the same organisational structure or not.

Fig. 3 graphically depicts an evolutive scale of the user configuration for different user configurations with respect to number of sessions and scenarios: an isolated stakeholder working alone, an individual stakeholder working as a group member, a single group of stakeholders, multiple groups of different stakeholders, multiple groups within the same organization (e.g., a group of representative end-users in a bank vs a group of interaction designers in the same bank), several

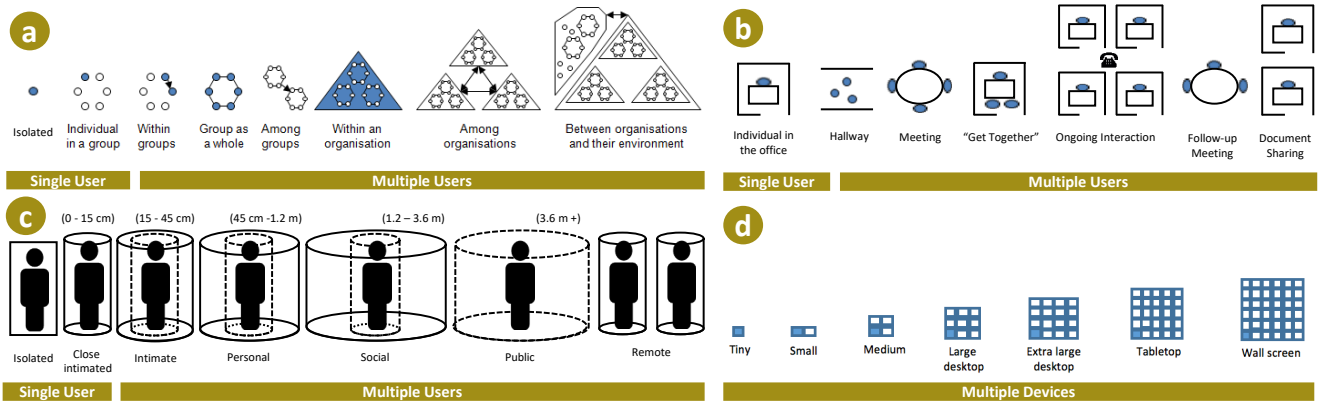

Figure 2: CROSSIDE dimensions: (a) the "user" dimension, (b) the "work method" dimension, (c) the "surface" dimension, and (d) the "device" dimension.

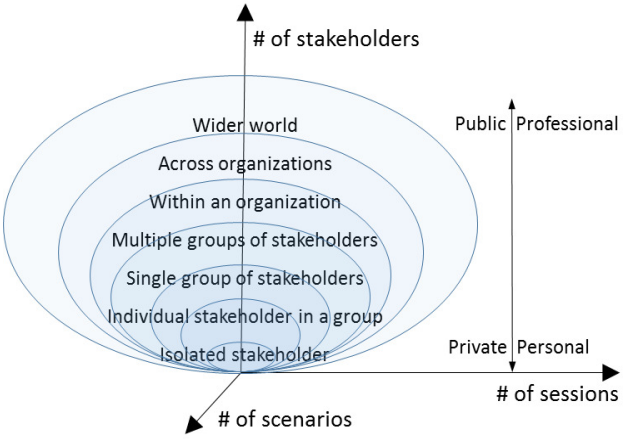

Figure 3: User configuration with sessions, stakeholders, and scenarios.

groups across organizations (e.g., an association of interaction designers in a particular domain of human activity) or the whole world (e.g., a community of practice could be developed that gathers people involved in a particular family of designs).

3.2. Surface Configurations

The *surface configurations* express a hypothetical distance between a stakeholder working alone and multiple ones collaborating in a personal, social, public or remote environment, thus touching the notion of territoriality. Fig. 2c differentiates configurations depending on concentric spatial zones centered around the stakeholder. The distance at which users feel comfortable interacting with others in tabletop environments depend on age and culture [37]. This study has contemplated the type of task or activity in which the users are engaged to influence tabletop design. In a completely isolated or close intimated space, a user works with personal interaction surfaces [48].

For example, a collaborative sketching system enables users testing UI prototypes on any surface size if the user is completely isolated. When users require a close intimate space, tiny or small surfaces will be used instead because they facilitate face-to-face collaboration and eye-contact [5]. Stakeholders who are not designers or developers usually come to meetings with their own interaction surface (e.g., a smartphone, a tablet) and would like to see the UI prototype on their very right devices, even if they drew something on another, possibly larger, surface. Personal spaces offer more ample possibilities than intimate spaces because users sitting around a table or standing side-by-side can share work-spaces by using multiple displays, large, or extra large desktops, tabletops, and wall screens. The social space takes place in a business-based interaction. Stakeholders and final users could interacting in shared work-spaces by using large tabletop display or wall screens.

For example, a social space is adopted when some kind of multiple user's interactions in taking place in a space between 1.2–3.6 m. Public space allows the user to interact with a sizable audience. In this context, the users make use of multiple display. Small surfaces allow manipulation of the work-space, while large surfaces serve to display the information to the entire audience. Informational wall surfaces, like public displays [41], provide shared views to the users standing far away from the display. Interactive walls enable users to walk up to the display and interleave interaction and discussion among participants. Finally, in remote spaces, users share work-spaces with same or different times. A collaborative sketching system could offer most surface configurations. Stakeholders interact simultaneously with any device. In a collaborative session, users can sketch scenes in a private way or separated into small groups, perform a task privately and then communicate the results to all stakeholders [11]. They can interact using integrated environments composed of horizontal or vertical displays regardless of the dimensions of these devices.

3.3. Input Interaction Techniques

The *input interaction techniques* express the sophistication degree with which sketching is supported, ranging from an indirect manipulation to full direct manipulation. This dimension takes into account the use of materials and the naturalness of the interaction. A classical indirect pointing device is the "mouse" because its locus of control (i.e., the

physical space in which actions occur) is different from or outside the locus of application (i.e., where the actions are applied). "Pen-based" interaction enables the user to interact with the device by using a passive or active stylus rather than a mouse. While pen devices initially supported indirect manipulation, they now support direct manipulation, namely by touch technology, regardless the fingers or the pointers involved. The "glove" considers the family of lightweight and stretchable devices that combines hand posture sensing and tactile pressure. Manipulation is direct, by device intermediation, like with smartwatches, armbands, and rings. "Tangible" devices naturally offer direct manipulation by connecting objects and surfaces to digital information. Tangible UIs typically work on tabletop surface and embed the tracking mechanism inside or outside the surface [49]. The "finger" considers situations where the end user interacts directly with the surface by finger tracking. This dimension could be extended to using the whole hand, but requires technologies for hand pose recognition. The last step, i.e. "full body", involves full-body gesture recognition, but its benefits for sketching are yet to be demonstrated.

3.4. Work Methods

The *work methods* characterise how interaction surfaces are spatially arranged according to a territoriality in the environment and how these environments are connected together (e.g., through Wi-Fi, LAN, WAN). Interaction surfaces could be *tiled*, *coupled*, *uncoupled*, or *positioned side-by-side* [48]. This dimension subsumes the physical location of each surface and how it is positioned (i.e., vertically, horizontally, in an oblique way). The dimension also considers the devices configurations of the five categories of technologies for collocated collaborative work classified by Wang et al. [5] namely: horizontal displays, large vertical displays, multiple displays, tangible interfaces, and integrated environments. Fig. 2b depicts typical setups of single and multiple users. Users can work alone in their own organisations. Multiple users work in corporate environments [9]: *hallway* (when an environment consists of any informal place where users could meet), *individual in the office* (when an environment only accommodates one user at a time, although this user can change over time), *meeting* (when the environment accommodates several users at a time for conducting a meeting), *get together* (when the environment accommodates several users for collaboration in general), *ongoing interaction* (when different environments involve many different users).

3.5. Tangibility

The *tangibility* dimension expresses to what extent the materialization of collaboration space is digital, physical, or mixed, ranging from applications in digital environments to the use of shape-changing devices. Digital environments define most classical applications where UI can be 1D (based on lines), 2D (based on surfaces), 2D1/2 (based on a space projected onto a surface) or 3D (in space). 1D UIs are typically based on command lines or instructions. In digital environments, users interact with a minimal immersion degree. Information is presented as a stream of characters in a text terminal. 2D UIs are incorporated in an environment based on conventional bi-dimensional representations where the interaction techniques are supported by events. Their widgets and their composition enable creating complex interfaces. The 2D1/2 and 3D interfaces extend classic GUIs with the notion of overlap and depth perceptions.

Virtual describes real environments simulated by a computer where users are involved with a high immersion degree. Users interact with virtual objects in an infinite 3D space. On the opposite of the dimension, applications are developed in real environments and use of mechanisms to increase the perception of the user. *Augmented Virtual* are applications including virtual worlds generated by a computer. These applications incorporate virtual reality to replace the physical world and the virtual world predominates over the real. Augmented Virtual extends the physical reality perceived by incorporating virtual objects into the physical world, thus increasing the degree of immersion. In contrast, *Augmented Real* considers an otherwise real environment augmented by means of virtual objects [50]. In the Augmented Real, virtual objects increase the real world, the dominant medium.

Tangible UIs give physical form to digital information, employing physical artifacts both as representations and controls for computational media [51]. This is further refined into four steps depending how tangible objects are materialized: real tangibility, stable physicality, elastic physicality, and plastic physicality. Real tangibility attempts the reproduce the physical behavior of a real world object into the tangible object, or a sub-set of it. Stable physicality occurs when tangible objects never change their behavior, whether the are expected to mimic some real work or not. Elastic physicality groups all environments that use both physical materials with computational analysis and simulation. These environments are used to understand and represent the world. Plastic physicality occurs in the area of shape-changing UI, where the devices are artifacts whose surface and/or volume can be articulated and modulated with their spatial domain [40]. In physics and materials science, the physicality property is one form of adaptation, which could be decomposed into three major properties: *plasticity*, which describes the deformation of a material undergoing (non-reversible) changes of shape in response to applied forces; *elasticity*, which is the tendency of solid materials to return to their original shape after being deformed; and *viscosity*, which expresses to what extent a material is resistant with respect to deformation. Solid objects will deform when forces are applied on them ; if the object is elastic, it will return to its initial shape and size when these forces are released. These two terms can be used for tangibility.

3.6. Layout

The *Layout* dimension refers to the facility layout concept [52] borrowed from Production and Operations Management aimed at optimizing the physical arrangement of

resources (e.g., a machine, a device, an operator) to maximize the quality and the quantity of the output, while minimizing the cost of involved resources. The different types of layout are [52, 53]:

- *Process* layout, when all resources performing similar type of operations are grouped at one location according to their functions. In our case, this means that all human, software and hardware resources required for each phase are gathered in the same place to support the collaboration, such as all resources for design in one place while development occurs in another place. The flow paths of information from one functional area to another vary from product to product. Usually the paths are long with backtracking possible.
- *Product* layout, when all resources are located according to the processing sequence of the product. In our case, resources are gathered in one location at a time depending on the phase and the locations are arranged in a sequence that follows the product life cycle, such as requirement, early analysis and design, advanced design, development, deployment, evaluation.
- *Combination* layout, when input to be processed come in different types and sizes. The combination layout is process layout where resources are arranged in a sequence to produce various types and sizes of products. The sequence of phases remains the same for the products having different types (e.g., only designing a concept vs sketching and designing) and sizes (e.g., sketching the home page of a web site or sketching the whole web site).
- *Fixed position* layout, when major physical resources remain in a fixed location (e.g., devices and platforms for sketching) and human resources are sent to this location depending on the phase. For example, a tabletop setup for collaborative sketching is heavy to move from one location to another and complex to re-calibrate, therefore stakeholders are brought to this location to ensure the phase. On the other hand, a small mobile sketching station could move from one location to another.
- *Group* layout, when product and process layout are combined in a particular layout called cell that satisfy a predefined set of requirements.
- *Group Technology layout*, when a group layout emerges from so-called Group Technology [53], which is aimed at analysing and comparing items to group them into families with similar characteristics. Families of inputs sharing similar requirements are grouped into cells, each cell being capable of satisfying all the requirements assigned to it. For example, phases sharing similar user requirements, even from different projects, will be conducted in the same place with the same resources.

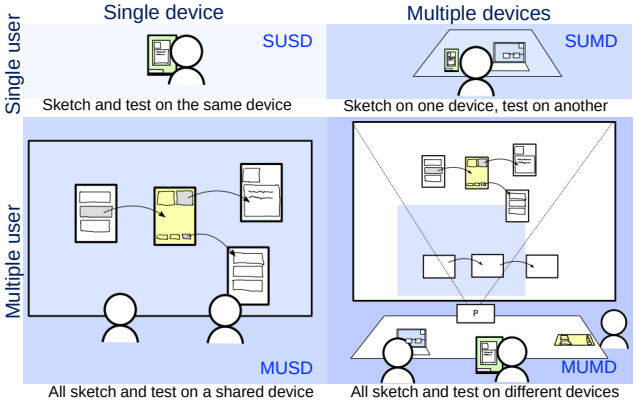

Figure 4: Quadrant of the four configurations.

Figure 5: Example of a SUSD configuration: one user with one device among several.

3.7. Device Configurations

The *device configurations* refer to the surface size of device independently of their density and orientation. Fig. 2d depicts the different dimensions for the device configurations. The sizes are categorized by generalised dimensions ranging from tiny to wall screen. A system could run on a variety of devices offering different sizes, performing scaling and resizing to accommodate variations induced by different screens. Four typical configurations based on stakeholders (or users) and their surfaces therefore emerge (Fig. 4):

1. *Single user-Single device* (SUSD): a single stakeholder, such as an end-user, is working in isolation on one device only to sketch a UI (see figure 4 - top left). This device could be the very right device on which the final UI should run or another one. Therefore, this configuration is appropriate for conducting the sketching, prototyping, and testing activities as defined.
2. *Single user-Multiple devices* (SUMD): a single stakeholder is sketching on multiple devices either simultaneously or asynchronously in order to assemble the various sketches into a coherent design scenario (see figure 4 - top right). Therefore, this configuration is appropriate for conducting the sketching, prototyping, and sharing/testing activities.

Figure 6: Example of a SUMD configuration: one user sketching on one device (a tablet) and testing on another (a wall screen).

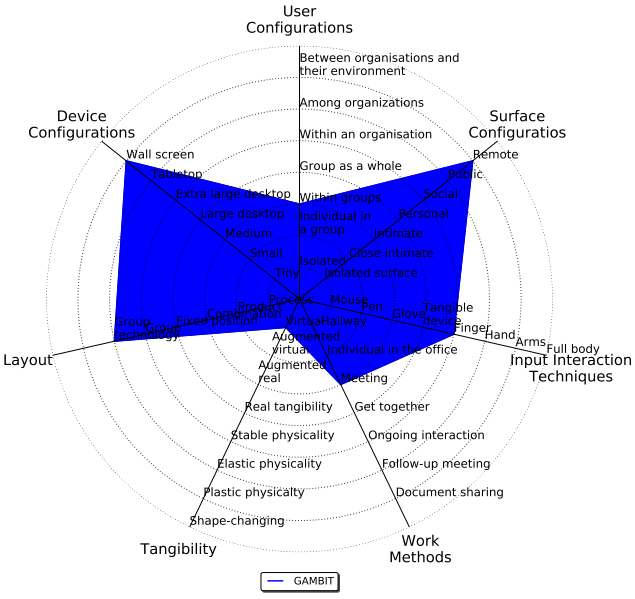

Figure 7: CROSSIDE instantiation for GAMBIT.

3. *Multiple user-Single Device* (MUSD): several stakeholders share a same device in order to interact together and to come up with an agreement on some design questions (see figure 4 - bottom left). This configuration is expected to mimic the classical whiteboard configuration where different stakeholders are together in front of a whiteboard and sketching things all together. Therefore, this configuration is appropriate for supporting the sharing/testing and discussing/reflecting activities, while it could be also used for the sketching and prototyping activities, but not primarily.
4. *Multiple users-Multiple devices* (MUMD): several stakeholders exploit private and public devices to apply any decided modification on the design scenario (see figure 4 - bottom right). They could perform these actions first on their own private device (e.g., their tablet), and then propagate modifications to the whole scenario (displayed on a wall screen). They straightforwardly interact on the public device to collect immediate feedback from other stakeholders. Therefore, this configuration is appropriate for supporting the sharing/testing and discussing/reflecting activities.

3.8. Building process of the Design Space

The design space has been built by identifying the greatest common denominator of software features described from forty-one references in various domains. The process followed to build the design space was basically a middle-out approach that combines two sub-processes, while trying to satisfy the principle of separation of concerns regarding the three main aspects of a context of use (e.g., users and their interactive tasks, their platforms and devices, their physical and organisational environments [46]):

1. A *bottom-up approach* consisting of browsing each reference at a time, identifying any high-level factor supporting collaborative sketching by separating them for users, platforms, and environments. All values found for each factor were collected in the same set.
2. A *top-down approach* consisting of determining such high-level factors by looking after a theoretical definition in the literature.

For example, the "User configurations" factor was identified among several references, therefore encouraging us to find a taxonomy that is relevant enough to express values found for this factor. We discovered a similar concept in Mandviwalla and Olfman [54]. To make it suitable for our purpose, we expanded the possible values from values collected in the references and we adapted the definition accordingly to make it categorical. Fig. 2a graphically represents the resulting factor presented as a progressive dimension.

Another example is related to the "Surface configurations" factor. By extracting from each use case found in every reference, we wanted to categorise the surface projected on the ground that delineates the interaction space. When only one person is involved in a sketching, there are various comfort zones from social psychology. When several persons are involved in a collaborative sketching, the way they interact with their system depends on what they want to do with their sketch. The sketching activity itself is more frequently found in small surfaces, while sharing and discussing the sketching is more frequently found in medium to large surfaces. The scope of the input and the output determines the surface.

4. Instantiation of the Design Space

This section first instantiates CROSSIDE on GAMBIT, one of our systems for collaborative sketching, to check how the descriptive virtue is addressed. It then performs the instantiations for the external tools: BELONGINGS [55], SKETCHML [20], FLEXISKETCH [28, 29, 56], CALICO [57] and EVE [22]. For the instantiations of each of the tools, we superimpose the first drawing to address the comparative virtue.

At the end, all instances are presented in the same design space allowing to see their similarities and differences.

4.1. The CROSSIDE instantiation for GAMBIT

GAMBIT[1] (Gatherings And Meetings with Beamers and Interactive Tables) [11] is a freely accessible[2] multi-platform HTML5 environment for collaborative design by sketching for UI design, but not only, by supporting (Fig. 7):

- *Multi-fidelity*: it should enable stakeholders to provide material at any level of fidelity and to easily switch from one level of fidelity to another.
- *Multi-format*: while sketching remains at the heart of the service, it should accommodate various formats of data both as input and as output.
- *Multi-session*: it should offer the possibility to create, edit, one or many sessions simultaneously.
- *Multi-scenario*: within a single session, it should be able to manage several alternate scenarios at once, namely by representing explicitly the different scenarios and by linking them. For instance fork and join mechanisms could be used to map scenarios together and create different design paths for idea writing. Different stakeholders make explicit more design ideas so as to generate new design paths that were remained under explored before. This is very important to avoid the "Do not fall in love of your own design" observation: once a design alternative is selected as an official one, some stakeholders tend to keep it (they fall in love of their design) and to spend remaining time in refining this design as opposed to exploring alternate designs.
- *Multi-stakeholder*: a same session could be created to involve a wide community of stakeholders whose roles and responsibilities may evolve over time. When the role or the responsibility of a stakeholder changes depending on the context of the problem, the tool should accommodate these changes flexibly.
- *Multi-access*: each stakeholder should contribute to a design session with her own devices. End-users as well as designers tend to create smaller sketches on small screen devices and larger sketches when the interaction surface increases, without necessarily adding more details. Multi-device is equally important to support cross-device interaction, e.g., when a stakeholder may switch from one device to another while participating in the same session, but with different roles.

[1]The word "gambit" is used in chess in reference to a movement in which the player sacrifices a piece, usually a pawn, in exchange to some future advantage; in user interface design the gambit would be made by the designer who sacrifices sketches in exchange to know in advance how the interface would function in the hands of the end-users, allowing her to learn from the eventual observation of this usage.

[2]Accessible at http://gambitsketch.appspot.com/

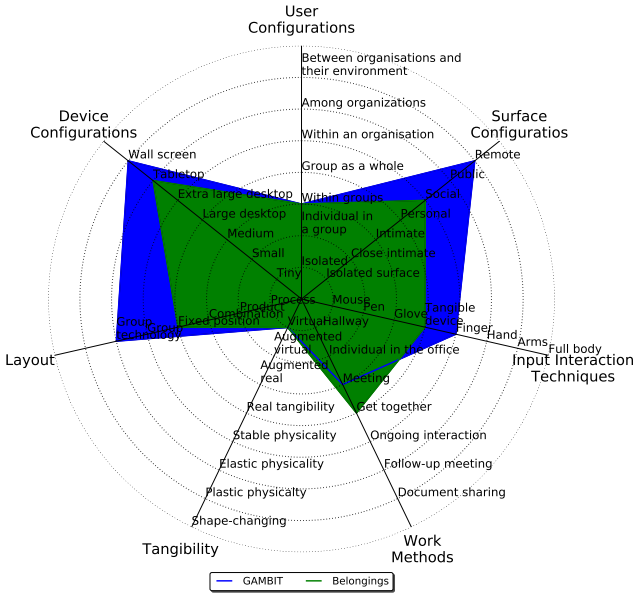

Figure 8: CROSSIDE instantiation for BELONGINGS superimposed to the instantiation for GAMBIT.

4.2. The CROSSIDE instantiation for BELONGINGS

BELONGINGS [55] consists of an interactive tangible tabletop surrounded by multiple interactive surfaces and smaller devices for collaborative query by sketching. The table was installed in a museum and designed to communicate indigenous traditional knowledge and cultural values. Fig. 8 depicts the design space instantiated for this environment superimposed on the previous one (Fig. 7) to compare the coverage of both tools. BELONGINGS can be operated in a fixed tabletop by two users working together in a social context. Unlike GAMBIT, BELONGINGS supports the use of tangible devices, thus being better with that respect on the dimension. Both software share a participatory design [8] approach as part of the development process: designers and final users work together to produce the final application.

4.3. The CROSSIDE instantiation for FLEXISKETCH

FLEXISKETCH [28, 29, 56] comes in two versions: the first uses tablets as sketching media and supports inexpensive, mobile sketching at any time and in any place, while the second version, known as "FlexiSketch Desktop", runs on electronic whiteboards and provides a wide screen for collocated meetings. Multiple tablets can be connected to the desktop version over Wi-Fi, thus enabling end users to collaborate and simultaneously work in the same workspace regardless the multiple screens they are using. FLEXISKETCH enables participants to edit the workspace simultaneously, offers shared and private views, and allows group work and/or facilitating individual work. Fig. 9 depicts how FlexiSketch is instantied on the design space. As a way for comparison, the instantiation of Gambit is also superimposed. FlexiSketch and Gambit have the similarities of considering the participatory design as part of the development process. Both tools

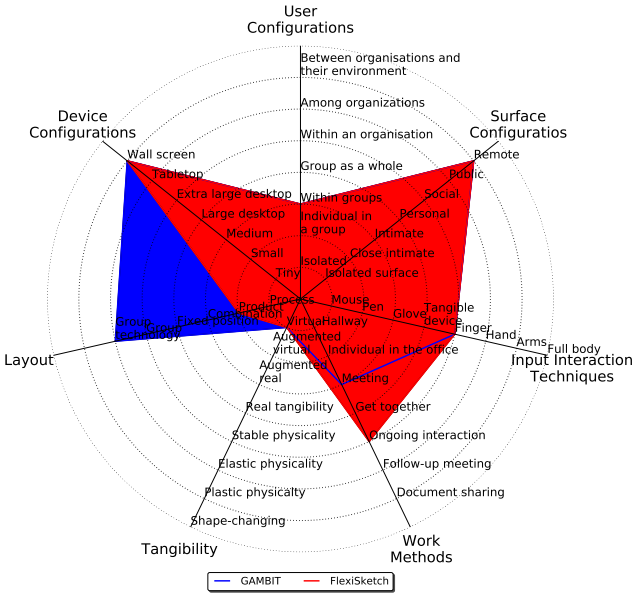

Figure 9: CROSSIDE instantiation for FLEXISKETCH on GAMBIT.

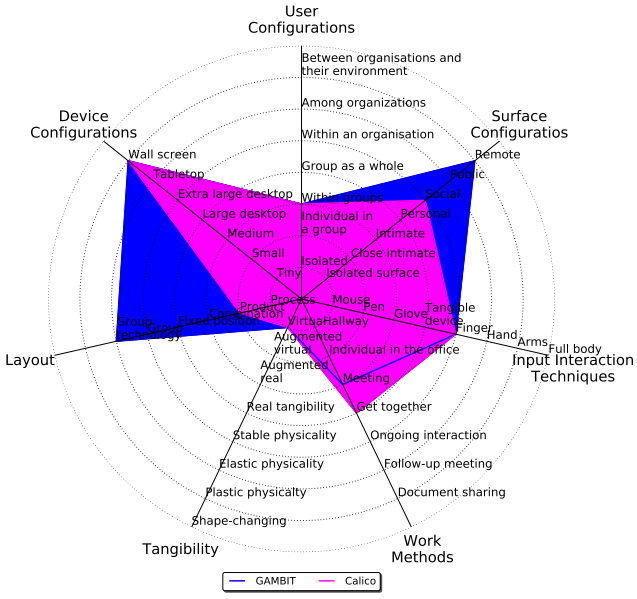

Figure 10: CROSSIDE instantiation for CALICO superimposed to the instantiation for GAMBIT.

allow multiple concurrent accesses through the simultaneous use of multiple devices.

4.4. The CROSSIDE instantiation for CALICO

CALICO [57] is a free hand rapid design tool supporting early software design activities to be used with touch screen interfaces, such as interactive whiteboards and tablet PCs. CALICO enables designers creating designs on multiple canvases based on a client-server architecture, supporting up to 20 simultaneously active users [58]. A CALICO client is portable, supporting computers connected to electronic whiteboards, laptops, and tablets. Thus, CALICO supports collaborative work across multiple devices, allowing multiple designers to work synchronously on the same canvas or asynchronously on different canvases. This allows designers working in a group to branch off to their own canvas. Fig. 10 combines the instantiations of CALICO and GAMBIT.

4.5. The CROSSIDE instantiation for SKETCHML

SKETCHML [20] is a framework that offers the ability to define and recognize every kind of 2D graphical library, by using freehand drawing, to be used in the construction of user interfaces. The framework uses an empirical language based on XML. It language allows the compatibility of SketchML with other applications and services through various devices. Fig. 11 combines the instantiations of SKETCHML and GAMBIT.

4.6. The CROSSIDE instantiation for EVE

EVE [22] is a Sketch-based prototyping workbench that facilitates end-users to define their design through a set of low-fidelity sketches. The Low-fidelity representations are recognized and translated in medium fidelity representations, as well as in high fidelity prototypes. End-users realize the representations in a canvas of two dimensions. End-users

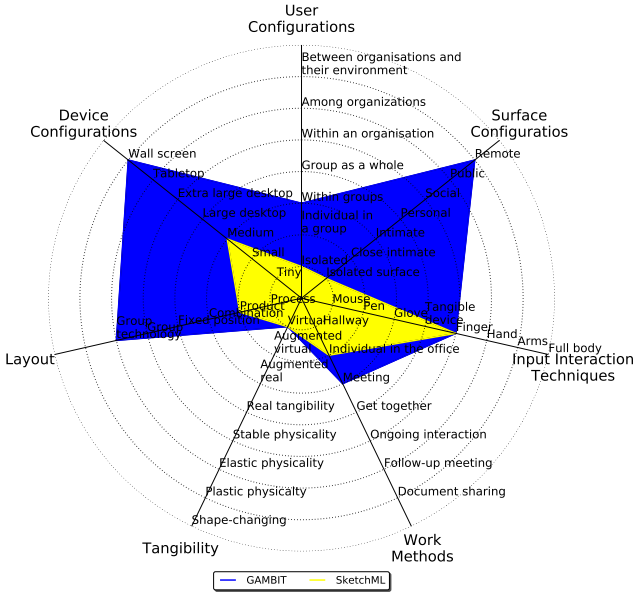

Figure 11: CROSSIDE instantiation for SKETCHML superimposed to the instantiation for GAMBIT.

can navigate through the three levels of loyalty. At each level it is possible to make the desired changes. For each of the fidelity, end-users can operate three modes. The design functionality, the configuration of the interaction and the preview of the prototype. Fig. 13 combines the instantiations of EVE and GAMBIT.

4.7. The comparative virtue of CROSSIDE

Fig. 14 combines all the instantiations of tools studied insofar in a radar diagram to facilitate the visual comparison of the tools (provided that they are not too numerous), such as the similarities and differences. The goal is to satisfy

Figure 12: Meeting work method for GAMBIT.

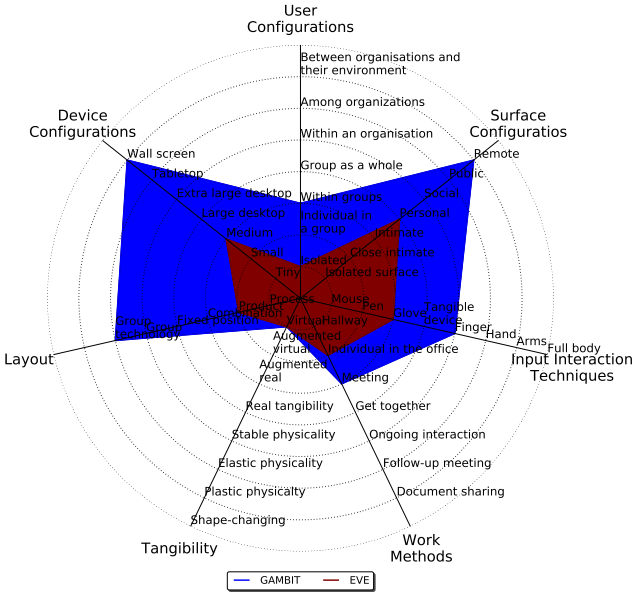

Figure 13: CROSSIDE instantiation for EVE superimposed to the instantiation for GAMBIT.

the comparative virtue of the design space, not to promote an ideal tool being the best along all dimensions. Rather, some tools are more advanced along some dimensions, say the "Device configurations", while others are targeting more flexibility for other dimensions, like "User configurations". For example, the range of "Device configurations" is well covered by both GAMBIT and CALICO, up to the three first steps of "Work methods". Fig. 12 illustrates the "Meeting" work method, where two designers share a tabletop for sketching a prototype by pen that is rendered in real-time on the smartphone of an end-user. Sketching is in direct manipulation: the designer is sketching directly on the tabletop and manipulating the sketch in the same way.

4.8. The generative virtue of CROSSIDE

A close examination of Fig. 14 leads to the observation that some dimensions are pretty well covered, like "Device configurations" and "Surface configurations", some others are moderately covered like "Layout", "Work methods", and "Input interaction techniques" while the remaining ones are limited, such as "User configurations" and "Tangibility". This suggests that these dimensions are welcome to be explored in further activities of a research agenda.

Regarding the "User configurations" dimension, we observe that the maximum step is "Within groups" because existing software do support multiple users (as represented in Fig. 4), but do not explicitly record and maintain the organisational structure among stakeholders. Their roles remain undifferentiated and the organisation structure is absent. A possible extension along this dimension would incorporate the explicit definition of such a structure, along with the roles played by stakeholders, especially in teams distributed in time and space. In this way, various types of organizations could be included, such as design teams, development companies like off-shore companies.

Regarding the "Device configurations" dimension, we observe that this dimension is the best one covered in the design space since the ultimate step is reached. Indeed, most software accommodate various types of devices and platforms, usually in a multi-platform or cross-device fashion.

Regarding the "Input Interaction techniques" dimension, the maximum step reaches finger-based interaction when drawing or sketching is conducted based on the physical movements of fingers, usually represented as uni- or multi-stroke gestures, thus limiting the interaction to 2D. 3D interaction is not really exploited, unless it is for the purpose of 3D objects. One could imagine for instance full-body gesture interaction to enable stakeholders to arrange pages of a web site dynamically in front of a wall-screen instead of moving them by point and click.

Regarding the "Work methods" dimension, the typical method observe seems to be "Ongoing interaction", which means that interaction capabilities remain opportunistic and constant whatever the phase is and whoever the stakeholders are. Some software support design history with do, undo, redo (e.g., by replaying the sketching actions), but this activity is not synchronized with a software management tool or with a document sharing system to store the current status of a design. However, most software includes a facility to export its contents to be integrated in the software documentation.

Regarding the "Tangibility" dimension, we notice that this is the most limited dimension in all tools examined so far: most of them represent digital solutions where sketching is achieved on a 2D surface with limited beautification performed in this space. There are probably other tools for collaborative sketching in 3D exhibiting the capability to virtually augment the real world, but they do not belong to our initial list of references. For the practitioners, this means that no tool is available today to fulfill these needs. The design of interest remains also only digital. Although some software exist that address the needs of physical-digital interfaces or objects, such as so-called *phygital objects*, they are not integrated with collaborative design tools.

Regarding the "Layout" dimension, it is difficult to assess this dimension since the physical setup and arrangement of interaction surfaces and their users is not explicitly repre-

sented, as in a design topology. Most interaction surfaces are mobile or partially mobile, thus enabling them to be rearranged depending on the requirements of the phase. But there is no explicit mechanism to represent the requirements of a design that would be turned automatically into a physical configuration to support it. Instead, stakeholders change the layout themselves depending on the constraints they perceive, apart from fixed or heavy interaction surfaces that are only found in dedicated locations. An interesting extension her would be to explicitly consider the notion of *territoriality* [59] to express the public, private, and common spaces for collaborative actions that would be then transformed into an adaptable layout of interaction surfaces.

Regarding the "Device configurations" dimension, the covered part shows again that a wide range of device is typically supported, ranging from small to very large devices. This is especially the case when such devices benefit from a HTML5-compliant browser that enables them to communicate easily.

The design space obtained so far only reflects some significant dimensions identified among a set of forty-one references considered as representative instances of collaborative sketching in various domains of human activity, ranging from learning to industrial design. While this set of references covers several domains, we do not argue that its coverage is complete or representative enough of the vast majority of tools of interest. Therefore, our next step consists of conducting a Systematic Literature Review (SLR) [60] for identifying references relevant to collaborative sketching for multiple purposes:

1. To expand the coverage of reviewed works from our 41 selected references to a larger panel.
2. To address the reproductibility of the procedure for guaranteeing the coverage of the design space.
3. To address explicitly the generative virtue by discussing the most promising configurations on this design space which may serve for a research agenda in the near future.

For the moment, we can superimpose the instantiations of the design space performed for the 41 references. On one hand, this superimposition enables us to identify portions of each dimension that are more or less frequently covered, or not covered at all. But this analysis considers only one dimension at a time, which may be considered as reductive. On the other hand, the superimposition also enables us to identify the configurations that are the most or the least frequently adopted by tools. This does not mean that they are appropriate or not, but simply the coverage could be discussed. We prefer to perform this analysis on a set of references resulting from the SLR instead of our initial set.

5. Conclusion and Future Work

We presented CROSSIDE, a design space for representing capabilities of a software for collaborative sketching in a cross-surface setting. This design space consists of seven

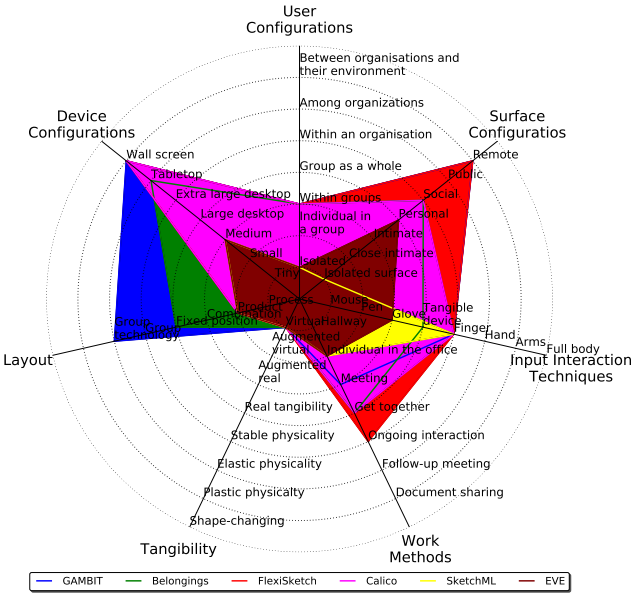

Figure 14: CROSSIDE instantiations for BELONGINGS, FLEXISKETCH, CALICO, SKETCHML, EVE and GAMBIT all combined at once.

dimensions (i.e., user configurations, surface configurations, input interaction techniques, work methods, tangibility, layout, and device configurations) resulting from a comparative analysis of 41 references in the domain. Each instantiation of these 41 references on the design space is graphically depicted as a radar diagram, which visually supports three virtues: descriptive, comparative, and generative.

Acknowledgments

The authors thank the anonymous reviewers for their constructive and patient comments on earlier versions of this manuscript.

References

[1] S. McCrickard, Making Claims: The Claim as a Knowledge Design, Capture, and Sharing Tool in HCI, Morgan & Claypool, June 2012. URL: https://www.morganclaypool.com/doi/abs/10.2200/S00423ED1V01Y201205HCI015. doi:10.2200/S00423ED1V01Y201205HCI015, Synthesis Lectures on Human-Centered Informatics.

[2] C. Gonzalez-Perez, Filling the Voids - From Requirements to Deployment with OPEN/Metis, in: Proc. of the Fifth Int. Conf. on Software and Data Technologies, Volume 1, ICSOFT' 10, SciTePress, 2010.

[3] R. van der Lugt, Functions of sketching in design idea generation meetings, in: Proc. of the 4th Conf. on Creativity & Cognition, C&C '02, ACM, New York, USA, 2002, pp. 72–79. URL: http://doi.acm.org/10.1145/581710.581723. doi:10.1145/581710.581723.

[4] J. Lin, J. A. Landay, Employing patterns and layers for early-stage design and prototyping of cross-device user interfaces, in: Proc. of the SIGCHI Conference on Human Factors in Computing Systems, CHI '08, ACM, New York, NY, USA, 2008, pp. 1313–1322. doi:10.1145/1357054.1357260.

[5] H. Wang, E. Blevis, Concepts that support collocated collaborative work inspired by the specific context of industrial designers, in: Proc. of the ACM Conf. on Computer Supported Cooperative Work, CSCW

'04, ACM, New York, USA, 2004, pp. 546–549. URL: http://doi.acm.org/10.1145/1031607.1031698. doi:10.1145/1031607.1031698.

[6] G. Costagliola, V. Deufemia, M. Risi, Sketch Grammars: A Formalism for Describing and Recognizing Diagrammatic Sketch Languages, in: Proc. of Eighth Int. Conf. on Document Analysis and Recognition, 29 August - 1 September 2005, ICDAR' 05, 2005, pp. 1226–1231. URL: https://doi.org/10.1109/ICDAR.2005.218. doi:10.1109/ICDAR.2005.218.

[7] M. C. Yang, Observations on concept generation and sketching in engineering design, Research in Engineering Design 20 (2009) 1–11.

[8] M. J. Muller, S. Kuhn, Participatory Design, Communications of the ACM 36 (1993) 24–28.

[9] E. Berglund, M. Bång, Requirements for distributed user interface in ubiquitous computing networks, in: Proc. of Conf. on Mobile and Ubiquitous MultiMedia, MUM '02, ACM Press, New York, USA, 2002.

[10] J. Vanderdonckt, A. Beirekdar, Automated web evaluation by guideline review, J. Web Eng. 4 (2005) 102–117.

[11] U. B. Sangiorgi, F. Beuvens, J. Vanderdonckt, User interface design by collaborative sketching, in: Proc. of the ACM Int. Conf. on Designing Interactive Systems, DIS '12, ACM, New York, NY, USA, 2012, pp. 378–387. URL: http://doi.acm.org/10.1145/2317956.2318013. doi:10.1145/2317956.2318013.

[12] S. W. Ambler, 2007, Agile adoption rate survey results: March 2007, URL: http://www.ambysoft.com/surveys/agileMarch2007.html.

[13] P. Sachse, W. Hacker, S. Leinert, External thought-does sketching assist problem analysis?, Applied Cognitive Psychology 18 (2004) 415–425.

[14] B. Buxton, Sketching User Experiences: Getting the Design Right and the Right Design, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007.

[15] S. Greenberg, S. Carpendale, N. Marquardt, B. Buxton, Sketching User Experiences - The Workbook, Academic Press, 2012. URL: http://store.elsevier.com/product.jsp?isbn=9780123819598.

[16] J. A. Landay, B. A. Myers, Interactive Sketching for the Early Stages of User Interface Design, in: Proc. of the ACM Int. Conf. on Human Factors in Computing Systems, CHI '95, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1995, pp. 43–50. URL: http://dx.doi.org/10.1145/223904.223910. doi:10.1145/223904.223910.

[17] A. Coyette, S. Faulkner, M. Kolp, Q. Limbourg, J. Vanderdonckt, SketchiXML: Towards a Multi-agent Design Tool for Sketching User Interfaces Based on UsiXML, in: Proc. of 3rd Int. Conf. on Task Models and Diagrams, TAMODIA '04, ACM, NY, 2004, pp. 75–82. URL: http://doi.acm.org/10.1145/1045446.1045461. doi:10.1145/1045446.1045461.

[18] A. Coyette, S. Kieffer, J. Vanderdonckt, Multi-fidelity prototyping of user interfaces, in: M. C. C. Baranauskas, P. A. Palanque, J. Abascal, S. D. J. Barbosa (Eds.), Proc. of 11th IFIP TC13 Int. Conf. on Human-Computer Interaction, September 10-14, 2007, volume 4662 of *INTERACT '07*, Springer, 2007, pp. 150–164. URL: http://dx.doi.org/10.1007/978-3-540-74796-3_16. doi:10.1007/978-3-540-74796-3_16.

[19] Z. Obrenovic, J.-B. Martens, Sketching Interactive Systems with Sketchify, ACM Transactions on Computer-Human Interaction 18 (2011) 1–38.

[20] D. Avola, A. Del Buono, G. Gianforme, S. Paolozzi, R. Wang, SketchML: a Representation Language for Novel Sketch Recognition Approach, in: Proc. of the 2nd Int. Conf. on PErvasive Technologies Related to Assistive Environments, PETRA '09, ACM, New York, NY, USA, 2009, pp. 31:1–31:8. URL: http://doi.acm.org/10.1145/1579114.1579145. doi:10.1145/1579114.1579145.

[21] D. Avola, L. Cinque, G. Placidi, SketchSPORE: A Sketch Based Domain Separation and Recognition System for Interactive Interfaces, in: A. Petrosino (Ed.), Image Analysis and Processing, ICIAP' 13, Springer, Berlin, Heidelberg, 2013, pp. 181–190.

[22] S. Suleri, V. P. Sermuga Pandian, S. Shishkovets, M. Jarke, Eve: A sketch-based software prototyping workbench, in: Extended Abstracts of the ACM Int. Conf. on Human Factors in Computing Systems, CHI EA '19, ACM, New York, NY, USA, 2019, pp. LBW1410:1–LBW1410:6. URL: http://doi.acm.org/10.1145/3290607.3312994. doi:10.1145/3290607.3312994.

[23] P. McInerney, F. Maurer, UCD in agile projects: dream team or odd couple?, Interactions 12 (2005) 19–23.

[24] T. Buchmann, Towards tool support for agile modeling: Sketching equals modeling, in: Proceedings of the 2012 Extreme Modeling Workshop, XM '12, ACM, New York, NY, USA, 2012, pp. 9–14. doi:10.1145/2467307.2467310.

[25] C. Lewis, P. G. Polson, C. Wharton, J. Rieman, Testing a walkthrough methodology for theory-based design of walk-up-and-use interfaces, in: Proc. of the ACM Conf. on Human Factors in Computing Systems, CHI '90, ACM, New York, USA, 1990, pp. 235–242. doi:10.1145/97243.97279.

[26] V. Genaro Motti, D. Raggett, S. Van Cauwelaert, J. Vanderdonckt, Simplifying the development of cross-platform web user interfaces by collaborative model-based design, in: Proceedings of the 31st ACM International Conference on Design of Communication, SIGDOC '13, ACM, New York, NY, USA, 2013, pp. 55–64. URL: http://doi.acm.org/10.1145/2507065.2507067. doi:10.1145/2507065.2507067.

[27] H. Ossher, A. van der Hoek, M.-A. Storey, J. Grundy, R. Bellamy, Flexible modeling tools (flexitools2010), in: Proc. of 32nd ACM/IEEE Int. Conf. on Software Engineering-Volume 2, ICSE '10, ACM Press, New York, USA, 2010, pp. 441–442.

[28] D. Wüest, N. Seyff, M. Glinz, FlexiSketch: A Mobile Sketching Tool for Software Modeling, in: D. Uhler, K. Mehta, J. Wong (Eds.), Proc. of Mobile Computing, Applications, and Services, volume 110 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, Springer, 2013, pp. 225–244. doi:10.1007/978-3-642-36632-1_13.

[29] D. Wuest, N. Seyff, M. Glinz, Flexisketch team: Collaborative sketching and notation creation on the fly, in: Software Engineering (ICSE), 2015 IEEE/ACM 37th IEEE Int. Conf. on, volume 2, 2015, pp. 685–688. doi:10.1109/ICSE.2015.223.

[30] R. Mitra, Rapido: A Sketching Tool for Web API Designers, in: Proc. of 24th Int. Conference on World Wide Web, WWW '15 Companion, Int. World Wide Web Conf. Steering Committee, Republic and Canton of Geneva, Switzerland, 2015, pp. 1509–1514. doi:10.1145/2740908.2743040.

[31] T. Hammond, R. Davis, Tahuti: A geometrical sketch recognition system for uml class diagrams, in: AAAI Spring Symposium on Sketch Understanding, 2002, pp. 59–68.

[32] N. Mangano, M. Dempsey, N. Lopez, A. van der Hoek, A demonstration of a distributed software design sketching tool, in: Proc. of the 33rd Int. Conf. on Software Engineering, ICSE '11, ACM, New York, NY, USA, 2011, pp. 1028–1030. URL: http://doi.acm.org/10.1145/1985793.1985985. doi:10.1145/1985793.1985985.

[33] J. Melchior, J. Vanderdonckt, P. Van Roy, A model-based approach for distributed user interfaces, in: Proc. of the 3rd ACM Symposium on Engineering Interactive Computing Systems, EICS '11, ACM, New York, NY, USA, 2011, pp. 11–20. URL: http://doi.acm.org/10.1145/1996461.1996488. doi:10.1145/1996461.1996488.

[34] J.-L. Pérez-Medina, S. Dupuy-Chessa, A. Front, A Survey of Model Driven Engineering Tools for User Interface Design, in: M. Winckler, H. Johnson, P. Palanque (Eds.), Proc. of 6th Int. Workshop on Task Models and Diagrams for User Interface Design, TAMODIA '07, volume 4849 of *Lecture Notes in Computer Science*, Springer, Berlin, 2007, pp. 84–97. doi:10.1007/978-3-540-77222-4_8.

[35] G. Johnson, M. D. Gross, J. Hong, E. Y. Do, Computational support for sketching in design: A review, Foundations and Trends in Human-Computer Interaction 2 (2009) 1–93.

[36] M. Johansson, M. Arvola, A case study of how user interface sketches, scenarios and computer prototypes structure stakeholder meetings, in: Proc. of the 21st British HCI Group Annual Conference on People and Computers, vol. 1, BCS-HCI '07, British Computer Society, Swinton, UK, 2007, pp. 177–184. URL: http://dl.acm.org/citation.cfm?id=1531294.1531318.

[37] J. R. Wallace, S. D. Scott, Contextual design considerations for co-located, collaborative tables, in: Proc. of the 3rd IEEE Int. Workshop on Horizontal Interactive Human Computer Systems, TABLETOP' 08, IEEE, 2008, pp. 57–64.

[38] J. Bowen, A. Dittmar, A Semi-formal Framework for Describing Interaction Design Spaces, in: Proc. of the 8th ACM Symposium on Engineering Interactive Computing Systems, EICS '16, ACM, New York, NY, USA, 2016, pp. 229–238. URL: http://doi.acm.org/10.1145/2933242.2933247. doi:10.1145/2933242.2933247.

[39] B. S. da Silva, V. C. O. Aureliano, S. D. J. Barbosa, Extreme designing: binding sketching to an interaction model in a streamlined HCI design approach, in: Proc. of 7th Brazilian symposium on human factors in CS, ACM Press, New York, USA, 2006, pp. 101–109.

[40] M. K. Rasmussen, G. M. Troiano, M. G. Petersen, J. G. Simonsen, K. Hornbæk, Sketching shape-changing interfaces: Exploring vocabulary, metaphors use, and affordances, in: Proc. of the ACM Int. Conf. on Human Factors in Computing Systems, CHI '16, ACM, New York, NY, USA, 2016, pp. 2740–2751. URL: http://doi.acm.org/10.1145/2858036.2858183. doi:10.1145/2858036.2858183.

[41] J. Müller, F. Alt, D. Michelis, A. Schmidt, Requirements and design space for interactive public displays, in: Proc. of the 18th ACM Int. Conf. on Multimedia, MM '10, ACM, New York, NY, USA, 2010, pp. 1285–1294. URL: http://doi.acm.org/10.1145/1873951.1874203. doi:10.1145/1873951.1874203.

[42] D. Loksa, N. Mangano, T. D. LaToza, A. v. d. Hoek, Enabling a classroom design studio with a collaborative sketch design tool, in: Proc. of the Int. Conf. on Software Engineering, ICSE '13, IEEE Press, Piscataway, NJ, USA, 2013, pp. 1073–1082. URL: http://dl.acm.org/citation.cfm?id=2486788.2486935.

[43] G. Costagliola, M. De Rosa, V. Fuccella, Local context-based recognition of sketched diagrams, Journal of Visual Langages and Computing 25 (2014) 955–962.

[44] R. Bellamy, M. Desmond, J. Martino, P. Matchen, H. Ossher, J. Richards, C. Swart, Sketching tools for ideation (nier track), in: 33rd Int. Conference on Software Engineering, ICSE '11, ACM, New York, NY, USA, 2011, pp. 808–811. doi:10.1145/1985793.1985909.

[45] M. D. Rosa, A. Fish, V. Fuccella, R. Saleh, S. Swartwood, G. Costagliola, A toolkit for knot diagram sketching, encoding and re-generation, in: G. Polese, V. Deufemia (Eds.), The 22nd International Conference on Distributed Multimedia Systems, DMS 2016, Salerno, Italy, November 25-26, 2016., KSI Research Inc. / Knowledge Systems Institute Graduate School, 2016, pp. 16–25. URL: https://doi.org/10.18293/DMS2016-035. doi:10.18293/DMS2016-035.

[46] G. Calvary, J. Coutaz, D. Thevenin, Q. Limbourg, L. Bouillon, J. Vanderdonckt, A Unifying Reference Framework for multi-target user interfaces, Interacting with Computers 15 (2003) 289–308.

[47] M. Beaudouin-Lafon, Designing interaction, not interfaces, in: M. F. Costabile (Ed.), Proc. of the ACM Int. Working Conference on Advanced Visual Interfaces, AVI '04, ACM Press, 2004, pp. 15–22. URL: https://doi.org/10.1145/989863.989865. doi:10.1145/989863.989865.

[48] J. Coutaz, C. Lachenal, S. Dupuy-Chessa, Ontology for multi-surface interaction, in: Proc. of the IFIP TC13 Int. Conf. on Human-Computer Interaction, 1st-5th September 2003, INTERACT '03, IOS Press, 2003.

[49] O. Shaer, E. Hornecker, Tangible user interfaces: past, present, and future directions, Foundations and Trends in Human-Computer Interaction 3 (2010) 1–137.

[50] P. Milgram, F. Kishino, A taxonomy of mixed reality visual displays, IEICE Transactions on Information and Systems 77 (1994) 1321–1329.

[51] B. Ullmer, H. Ishii, Emerging frameworks for tangible user interfaces, IBM Systems Journal 39 (2000) 915–931.

[52] B. J. Finch, Operations Now: Profitability, Processes, Performance, McGraw-Hill/Irwin, Boston, 2006.

[53] J. Wu, J.-y. Lv, Z.-k. Ye, M. Rui, Optimization Design of Facilities Layout in a Certain Machining Shop, in: Proc. of 2nd Int. Conf. on Information Technology and Management Engineering, ITME' 17, DEStech Publications, Inc, Lancaster, Pennsylvania, 2017. URL: http://dpi-proceedings.com/index.php/dtcse/article/view/7986. doi:10.12783/dtcse/itme2017/7986.

[54] M. Mandviwalla, L. Olfman, What Do Groups Need? A Proposed Set of Generic Groupware Requirements, ACM Transactions on Computer-Human Interaction 1 (1994) 245–268.

[55] R. Muntean, Considering Collaboration in ?eleẁk̀w—Belonging, in: Proc. of the 3rd Int. Workshop on Interacting with Multi-Device ecologies" in the wild", Cross-Surface' 16, 2016. URL: http://cross-surface.com/papers/Cross-Surface_2016-2_paper_9.pdf.

[56] D. Wüest, N. Seyff, M. Glinz, Collaborative sketching and notation creation with FlexiSketch Team, Software Engineering 2017 (2017).

[57] N. Mangano, A. Baker, A. Van Der Hoek, Calico: a prototype sketching tool for modeling in early design, in: Proc. of the 2008 Int. Workshop on Models in software engineering, ACM Press, New York, USA, 2008, pp. 63–68.

[58] N. Mangano, T. D. LaToza, M. Petre, A. van der Hoek, Supporting informal design with interactive whiteboards, in: Proc. of the ACM Int. Conf. on Human Factors in Computing Systems, CHI '14, ACM, New York, NY, USA, 2014, pp. 331–340. URL: http://doi.acm.org/10.1145/2556288.2557411. doi:10.1145/2556288.2557411.

[59] J. Thom-Santelli, D. Cosley, G. Gay, What do you know?: Experts, novices and territoriality in collaborative systems, in: Proc. of the ACM Int. Conf. on Human Factors in Computing Systems, CHI '10, ACM, New York, NY, USA, 2010, pp. 1685–1694. URL: http://doi.acm.org/10.1145/1753326.1753578. doi:10.1145/1753326.1753578.

[60] B. Kitchenham, R. Pretorius, D. Budgen, O. P. Brereton, M. Turner, M. Niazi, S. Linkman, Systematic literature reviews in software engineering – a tertiary study, Information and Software Technology 52 (2010) 792 – 805.