Abstract—We introduce KnoCap, a tool that we are developing to collect design knowledge from conversations. It is well-known that design meetings create knowledge, not all of which is typically captured. KnoCap addresses this problem by relying on Echo buttons for designers to indicate when important knowledge is being talked about, so that that knowledge is captured as a voice note that can be played at a future time. In this paper, we motivate our work, present KnoCap’s features, and discuss where we want to take KnoCap so it is not just useful in capturing knowledge, but also relaying it when so needed.

Keywords—Knowledge capture, design meetings, voice notes

I. INTRODUCTION

Note-taking, however mundane, is a key activity in many different domains [13]. It has a particularly important role in the workplace, where recording (aspects of) conversations is a priority, since it is impossible to remember everything that was said – or even just ‘important’ things – at a later point in time [13]. Yet, it is often crucial to be able to refer back to some of what was said [19]. Thus, taking notes and refining them post-meeting are behaviors that are often seen in professional collaborative environments.

In software development, meetings are equally prevalent and, thus, note-taking is equally important [6]. In this paper, we focus on one particular type of meetings: design meetings at the whiteboard [14]. In this setting, note-taking represents a challenge. On one hand, the content being produced on the whiteboard represents some form of knowledge that is being produced. On the other hand, much knowledge is lost, as the content of the whiteboard often does not capture all of what was said during the meeting. Key spoken knowledge evaporates, short of individuals remembering. This leads to a variety of problems, including a loss of design rationale [7], a lack of recording decisions being made[21], and more.

To counter this problem, some meetings involve a designer explicitly being assigned the role of note-taker. This is rare, however, and note-taking usually remains ad-hoc: individuals have to remember to do so. Particularly in the case of design sessions, however, this is difficult as the primary activity of the meeting—making design progress—nearly never pauses, especially when designers are in a flow [14]. Designers, thus, have to take notes very quickly and they face a tradeoff between taking notes and participating in the design discussion [13]. This situation leads them to take low-quality notes or not take them at all. Thus, much essential design knowledge goes undocumented, unless the designers make an effort to reconstitute it afterwards, perhaps with the help of whiteboard pictures and informal notes taken. Performing this activity after-the-fact is time-consuming, and often does not take place or cover all knowledge generated.

We introduce KnoCap, a novel note-taking tool that leverages physical Echo buttons to collect knowledge as voice notes. Our vision for KnoCap is dual: we intend for it to support lightweight capture of knowledge during meetings, and for it to deliver that knowledge (semi-)automatically precisely when it is needed in a future meeting. To date, we focused on the first aspect, which we describe in this paper.

The remainder of this paper is organized as follows. In Section 2, we motivate our work with a scenario. In Section 3, we introduce the primary design decisions underlying our work. We introduce KnoCap in Section 4 and related work in Section 5. We conclude in Section 6.

II. MOTIVATING SCENARIO

Consider the following scenario, inspired by the sessions conducted with professionals as part of the Studying Professional Software Design workshop [14]. Two software engineers are tasked with designing an educational traffic simulator for civil engineering students to use in practicing traffic signal timing and its effect on traffic flow. One of the designers (D1) turns to the whiteboard and starts designing the user interface. Meanwhile, the other designer (D2) examines the provided requirements:

- D2: One, students need to change the layout of the map to accommodate roads.
- D1: mmm... yep [D1 is sketching the interface].
- D2: Two, students need to modify map parameters, lights’ timing, the number of cards, and their respective speed, right?
- D1: Yes, they need some kind of [D1 makes a pause] dashboard [D1 starts drawing a dashboard].
- D2: Three, students need to watch all these objects interacting in real time.
- D1: Yes, maybe we can have two modes: an editing mode and a simulation mode.
- D2: Ok, students also need to visually validate their configuration is correct. I mean, that it avoids collisions.

Representing only a short portion of a near 2-hour meeting, the discussion reveals different types of knowledge being introduced by the two designers. D2 clearly introduces several requirements from the requirements document, which serve as the motivation for some of the subsequent ideas that D1 has and the decisions that they make. But not all the material being discussed is captured. For instance, the fact that students need to modify map parameters, lights’ timing, and so on is neither in the document that D2 is reading (it is their interpretation of what is needed to address a requirement in the document), nor in the content being drawn on the white-
board. As another example, the fact that the simulator should have two modes is again only verbalized, and not drawn. A final example concerns D2’s last sentence, with them stating the rationale (avoiding collisions) for the validation feature.

The workshop videos contain many of these kinds of discussions, with the designers rarely explicitly taking notes. It is easy to see why: in this short segment, a lot of potentially important knowledge is being generated. Writing it all down is simply too cumbersome, and taking pictures is ineffective since it would not capture much of what they said.

III. GUIDING PRINCIPLES

KnoCap’s design is guided by five principles that shape the features we present in the next section. Inspired partially by previous attempts at capturing knowledge from voice and visual recordings (e.g., [15, 16, 18, 19], these principles are:

A. Voice-oriented

While tools exist that help capture the history of sketches (e.g., [11,4]), our work instead focuses on capturing aspects of the conversation. We do not seek to record the entire conversation, however, as doing so would defeat the purpose in two ways: (1) much conversation would be captured that is not design knowledge, and (2) no one wants to go through a lengthy recording to find just one piece of design knowledge.

B. Retroactive, but in the moment

Clearly, humans cannot predict whether something important is going to be said, but they usually can recognize it once someone has started talking. Thus, our approach necessarily has to be retroactive (not pausing people to ask them to say it again just so a recording can be made). It also has to be in the moment, as soon as someone recognizes that important content is being shared. Listening to a full conversation after a meeting is over to tag knowledge simply will not happen.

C. User control

Ultimately, it still should be the designer who has to be in charge of what does and does not get captured. Automation can help, but also may inadvertently capture parts of the conversation that should not be captured (chit chat or commentary on the project or company).

D. Lightweight

Any solution to knowledge capture has to be lightweight so not to get in the way of the task of designing itself. If a designer gets too distracted by the tool, they will not use it.

E. Usable both instantly and later

Any knowledge captured should be usable right away, as well as at a later time. Without requiring manual processing, a designer should be able to access the knowledge captured as part of a current meeting. It is equally important that future tools can access the knowledge in ways that allow those tools to perform advanced processing on it.

IV. KNOCAP

KnoCap consists of four components: (1) an application that continuously records the design conversation in progress in the background, (2) an Echo Dot and associated Echo buttons which designers use to mark, while they are designing, important parts of the conversation that they want kept, (3) a database that stores kept voice notes as well as automatically created transcriptions of each note, and (4) a web application through which the voice notes can be accessed. Figure 1 presents the typical setup (we put the Echo button and Echo Dot in a conspicuous place for visibility; the Dot would usually not be visible and the button—or multiple buttons ideally—would be placed alongside the whiteboard, on a table where a few of the team members might sit, or even in one’s hand).

![Figure 1. KnoCap in use.](image)

While KnoCap is always recording the design conversation, it ultimately only saves voice notes per the desire of the user (that is, all other content is eventually deleted). Recording everything tends to create mistrust with people feeling on edge because of the hidden recording taking place. Moreover, long streams of audio are difficult to process afterwards, meaning that they would just remain that. It is the function of the Echo button to overcome both problems: by pressing the button, a designer explicitly signals to all that a recording is being made of what was said previously, and the recording is relatively short. Pressing the button once records the past 30 seconds of the conversation; twice records 60 seconds. Keeping the button pressed also records the conversation, but it is the exception in recording forward. This last use case is useful when one does know that an important part of the conversation is forthcoming, for instance when summarizing the previous step or listing next steps. Note that, because the button press is physical and noticeable by others, it is possible for someone to object to a recording, so it can be deleted right away.

For convenience, buttons for recording are also available in KnoCap’s Web application, which can run on any device with Internet access (i.e., a mobile phone in one’s hand), so to enable KnoCap’s use in situations where an Echo button is not at hand. The complete interface of the Web application is shown in Figure 2. Besides the virtual buttons to record any voice notes, the interface offers a playable list of voice notes that extends with each new note being recorded.

The Web application also allows access to recorded voice notes per design meeting, as shown in Figure 3. Through this screen, it is possible to search for meetings, and, per meeting, see all the voice notes that were recorded for it together with their transcriptions. As opposed to having to listen to an entire voice note, these transcriptions allow quick scanning of a voice note as to whether it has something relevant in it.
Reviewing our guiding principles discussed in Section 3, we note that our approach is voice centered, supports retroactive in the moment capture of short parts of conversations, puts the user in control, and is lightweight. For our motivating scenario of Section 2, for instance, all that is needed for the pair to capture the short bit of discussion is to press the Echo button once (perhaps twice if they talked slowly). For the final guideline, we observe that the associated Web application makes the voice notes usable immediately: they can be replayed at any time. However, they are equally usable in future, since they are stored permanently in the database.

We note that at this time we have only designed and implemented the core elements of KnoCap. We envision many enhancements for knowledge capture: using multiple Buttons to identify individual designers who feel certain knowledge is important to keep; leveraging overlapping timing of button presses by individual designers to determine an implicit level of importance (i.e., a note requested by two designers at once is listed above notes requested by a single designer), adding a mobile phone app with advanced and more precise controls for recording notes; and more. We also plan to explore alternative speech recognition toolkits (currently PocketSphinx1), in order to improve the quality of the transcriptions.

V. RELATED WORK

Finding effective ways of capturing design knowledge is not a new problem in software engineering. Indeed, much work has been done on this problem, particularly so (but not exclusively) in the research community. Below, we discuss key research in this regard, its contributions, as well as the needs that remain unaddressed today.

A. Design knowledge tools

Many tools have been developed to address both capturing and delivering knowledge. Examples include Arel [17], Decision Architect [12], ADDSS [3], Archium [10], Knowledge Architect [9], and SEURAT [1] (see [2] for a comparative study). These tools have been shown to have the potential of improving over manually capturing knowledge. Most of them, however, suffer from the problem that knowledge capture is a separate activity, with overhead for designers’ workload [7]. Only two of these tools address immediate knowledge collection: SEURAT (an Eclipse plug-in to capture rationale while programming) and the Knowledge Architect (a plug-in to capture “artifact fragments” from Word and Excel documents, which can then be reviewed later in a unified management interface). However, in both cases the interface for doing so is text-based; conversations cannot be captured. KnoCap differs from all of these tools in focusing on design at the whiteboard, capture of conversations, and being lightweight in its approach.

B. Note-taking tools

In software development, far more time is spent in face-to-face communication than any other communication modality [20]. Yet, in the software engineering literature, note-taking is not a subject of much research; instead, a focus on notations and tools that support precise specification of artifacts dominates the field. A few exceptions exist. For example, [5] presents an approach to include pictures into notes being taken, with the explicit intention of increasing the note-taker’s ability to pay attention to the meeting instead of taking notes. As a second example, [8] introduces a user study comparing four ways of skimmin presentation materials (e.g., reviewing presentation transcripts with highlighted sections, listening to audio-video summaries), finding that people prefer to watch multimedia summaries.

C. Voice-based note-taking

Several approaches aim at capturing knowledge directly from conversations ([15, 16, 18, 19]), recording the entire meeting (video or audio) and offering mechanisms to correlate the recorded video or audio stream with handwritten notes. In SAAMPlayer [15], annotation with handwritten notes happens after the fact. In the Audio Notebook [16], Filochat [18], and Dynomite [19], the physical device supports users in note-taking while the device records audio and synchronizes them on replay. Users can select an annotation to hear the conversation at that time, or select a specific moment on the audio timeline and be displayed the annotation they created at the time. KnoCap is related to these tools, but takes a different angle in remaining focused on knowledge as audio snippets and being more lightweight in its capture.

---

1 https://cmusphinx.github.io/wiki/download/
VI. CONCLUSIONS AND FUTURE WORK

During design meetings, much knowledge can be generated that is ultimately lost because it is not captured, whether in whiteboard sketches, personal notes, or even just in someone’s memory. To enable capturing knowledge in a way that does not necessarily distract software designers from the task at hand, we are creating KnoCap, a novel approach that relies on voice recording together with Echo buttons to keep critical snippets of the design conversation. So far, we have developed a relatively minimal working prototype, one that we are now both extending and starting to experiment with.

We have already described several extensions in Section 4 through which we plan to augment the knowledge capture side of KnoCap. An additional important area of future work relates to knowledge delivery. Having a database with voice notes and transcriptions enables two important avenues. The first is building an Alexa skill that allows developers to query the knowledge, for instance asking ‘What was the second thing that D2 said that students need to be able to do?’, ‘Why did D1 say they need a dashboard?’ or ‘What models did D1 propose?’

The second avenue concerns a proactive Alexa skill, one that continuously listens to the conversation, interpreting it in order to be able to make automatic suggestions. This clearly is longer-term research, but we believe that active participation by an Alexa skill may enhance design meetings in ways previously not possible. If, in some future meeting, D2 asks D1 ‘Did we talk collisions already?’, the skill could remind them they did and offer them to repeat what they said. Alternatively, if the skill overhears a designer asking about applicable design patterns, it could interpret the whiteboard sketch and search on the internet for patterns that match the sketch.

As a final direction, we want to explore machine learning to automatically identify important knowledge. If we are able to collect a large collection of button clicks with their associated voice notes and the original, complete conversations, we want to explore whether this data might be sufficient to train a machine learning algorithm. We do not necessarily see this as superseding our approach, given that false positives and false negatives are a fact of life in machine learning. Rather, we foresee cues that the designer can confirm or refute.

ACKNOWLEDGMENTS

We thank Juwin Viray, David Garcia, Pang Hangzhi, Zachary Little, Claudia Mastan, and Nijian Al-Amin for their work in developing KnoCap’s initial prototype, and David Kutas and Aditya Nair for being willing first guinea pigs.

REFERENCES


