Image-Gesture-Voice: A Web Component for Eliciting Speech

Mat Bettinson[1] **and Steven Bird**[2,3]

[1]Department of Linguistics
University of Melbourne

[2]Northern Institute
Charles Darwin University

[3]International Computer Science Institute
University of California, Berkeley

Abstract

We describe a reusable Web component for capturing talk about images. A speaker is prompted with a series of images and talks about each one while adding gestures. Others can watch the audio-visual slideshow, and navigate forwards and backwards by swiping on the images. The component supports phrase-aligned respeaking, translation, and commentary. This work extends the method of Basic Oral Language Documentation by prompting speakers with images and capturing their gestures. We show how the component is deployed in a mobile app for collecting and sharing know-how which was developed in consultation with indigenous groups in Taiwan and Australia. We focus on food preparation practices since this is an area where people are motivated to preserve and disseminate their cultural and linguistic heritage.

Keywords: language documentation, procedural discourse, mobile apps, crowdsourcing, web technologies

1. Introduction

The program of language documentation is one response to the rapid decline in linguistic diversity (Himmelmann, 1998; Woodbury, 2010). We are challenged to design scalable methods for documenting thousands of languages while there is still time. More generally, the wider problem space is the search for digital knowledge preservation at scale.

One promising avenue is in seeking collaboration with the wider audience speakers, or *crowdsourcing*. Crowdsourcing in other domains, from Google Maps to Wikipedia, is an established pattern for collective intelligence. These patterns are seen in countless user-contributed content apps. The architecture of crowsourcing apps is no different from any (Chatzimilioudis et al., 2012). However there has been little attention towards minority community use cases.

Practical crowdsourcing depends on a confluence of interests. App developers may seek collection and preservation of knowledge, while the target audience are seeking solutions to their problems. This calls for a process of exploration, negotiation and user-centered design. In 2016, a series of app design workshops explored ways to include linguists, technologists and speech community members (Bird, 2018). One of the emerging designs was an app for capturing talk about food preparation (Mettouchi et al., 2017; Bettinson and Bird, 2017; Bettinson, 2017).

This paper describes an approach to documenting procedural knowledge, or any kind of know-how. We report on the Android mobile app Zahwa, which we have tested with speakers of endangered languages in Taiwan and Australia. We also seek to bootstrap the creation of similar knowledge preservation apps in the future. To this end we discuss ongoing efforts to develop a library of reusable software components based on the emerging Web Component standard.

This paper is organised as follows. In Section 2. we discuss previous work. Then Section 3. proposes a new method for documenting procedural knowledge. The Zahwa app implementation is described in Section 6. We discuss recent work on Web Components for knowledge preservation in Section 6.

2. Previous work

There are very few apps specifically intended to let app users document and share their know-how. It's much more common to find apps that serve as a vehicle to publish content compiled by experts, such as dictionaries, or language teaching apps. Digital knowledge preservation lags the *Web 2.0* trend towards user-contributed and socially contextualised participation.

Crowdsourcing acoustic data is one established genre of mobile app. *Voice App* collects regional speech data for Swiss-German to study dialectal variation (Goldman et al., 2014). *English Dialects App* includes a dialectal prediction feature a form of 'gamification' to encourage people to use the app (Leemann et al., 2016; Leemann et al., 2018).

Dictionaries are a popular genre of mobile app. While many have been made for endangered languages, the majority don't allow user-contributions. The *Ma! Iwadja* app incorporated a "crowdsourcing lexicon development system" but the app is no longer functional. We have found it sadly quite common that language apps vanish without a trace. This serves as a reminder of the ongoing challenge of sustainable development.

The *Aikuma* mobile app is mobile app capable of crowdsourcing natural language (Bird et al., 2014). *Aikuma-LIG* is a further development aimed at collecting data speech processing (Blachon et al., 2016). While capable of crowdsourcing, these apps are intended for researcher-driven usecases. They are not designed with general audiences in mind and don't readily support sharing with other users.

The web platform is a viable choice for building the next generation of digital knowledge preservation tools (Bettinson and Bird, 2017). The acoustic waveform display library Wavesurfer[1] is an example of a successful open source software component in the web domain. The Web Component[2] (WC) standard is now supported or 'pollyfilled' for all major web browsers. There's growing momentum behind a library of library of freely available WC components[3].

[1] https://wavesurfer-js.org/

[2] https://developer.mozilla.org/en-US/docs/Web/Web_Components

[3] https://www.webcomponents.org/

Commercial apps live or die by their ability to engage users. The ability to share content via existing social networks is ubiquitous across popular mobile apps. App users are motivated to share for a number of reasons including reciprocity, social engagement and reputation building (Oh and Syn, 2015). Aside from fulfilling user expectation, sharing also serves a helpful marketing function akin to *viral marketing* (Subramani and Rajagopalan, 2003). Finally, sharing meets the need of disseminating cultural knowledge. Digital dissemination is particularly helpful for maintaining indigenous knowledge (Chikonzo, 2006).

3. Documenting procedural knowledge

Procedural discourse is defined as "an explanation or description of a method, process, or situation having ordered steps. Examples of procedural discourses include recipes, instructions, and plans" (Johnson and Aristar Dry, 2002). In the context of endangered languages, traditional practices and rituals are falling out of use. The practices include how to prepare medicinal remedies, how to build a canoe, how to grow yams, the stages of initiation, and so forth. Of particular significance are the traditions concerning food preparation, since these connect with the local environment, the seasons, the calendar of rituals, family ties, and cultural identity. Documenting procedural knowledge is a recognised part of language documentation (Pollock, 2011).

Many speakers of endangered languages are mindful that the younger generation are not learning traditional crafts, rituals, methods for food preparation, and so on. For instance, by the time a girl grows up and has a family of her own, it may be too late to ask her grandmother how to prepare the dish that marks a particular rite of passage. Similarly, geographical separation between generations can make it difficult to transmit knowledge.

In literate societies one could email the instructions. Another approach would be to record a video of the procedure, where it may be later shared on YouTube. However, video recording demands either a single fluid performance or a video editing process in order to achieve a succinct result. It is difficult to self-record video. When multiple participants are involved, it becomes more of a performance and needs to be planned. Later, in translation, it is harder to replace the audio track and keep everything synchronised.

The mobile phone platform provides the ability to take photos to use as prompts, in a way that is similar to the use of stimulus in non-digital documentary methods (Lüpke, 2010, p.58). Further more, mobile phones offer a tactile user interface that offers the additional benefit of capturing gesture at the same time as spoken voice. In the following section we discuss a design process of an app that incorporates a new method for documenting procedural knowledge.

4. App design

Attention to the design process is important where our goal is to create apps that people are self-motivated to use. We are particularly at risk where app development and app deployments take place in different cultures. In this section we illustrate the design process from an app workshop we organised in Darwin in September 2016.

Following the basic principles of user-centered design, we describe the "personas" involved in the app (Norman and Draper, 1986; Bird, 2018). We make these as real as possible, inventing names, and discussing motivations.

Taos left the village of her childhood when she was 8. Now she is 23 and lives in Algiers with her family and studies at the university. When she is in the dorms she cooks meals with other young women. One time while preparing food they were talking about their childhoods and Taos spoke about her holidays in her village and her grandmother's delicious cooking. Because they had similar experiences, the young women decided to collect and share these traditional recipes. The next holidays, Taos is visiting her grandmother, and decides to photograph the stages of preparing each recipe.

Zahwa has always lived in the village, and has learnt to cook with her own mother and grandmother. She enjoys cooking, and is very happy when her granddaughter comes to visit her from Algiers and brings her news of the capital and her life at university. Zahwa doesnt know how read or write, so when Taos asks her about her recipes, she tells her that the best way to learn is to watch and do it with her. But then Taos says it would be better if she recorded her grandmother making the recipe, so that she would be able to do it herself, and that she would be proud to share it with her friends and maybe other people too.

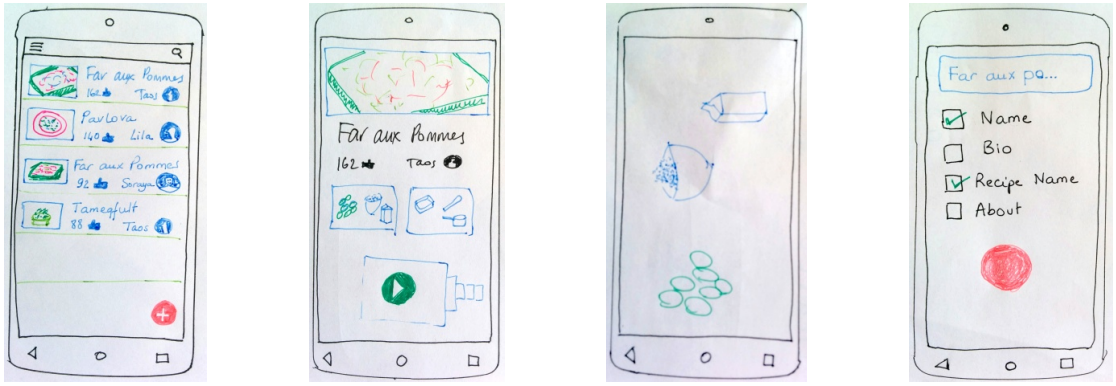
Next, we write out the value proposition, i.e. how would our proposed solution connect with the jobs, pains, and gains experienced by one of the personas (Bird, 2018). In this case, we take the perspective of Taos, as the one who drives the creation of the content (see Figure 1).

Finally, we express the design in terms of a series of app screens (see Figure 2). When the app is opened we see a list of popular recipes (Fig 2(a)) Perhaps we can follow other people, or see recent changes, etc. For each recipe we can see the number of "likes". There's a + button that lets us add a new recipe. We can open an individual recipe to see a larger image (Fig 2(b)). There are pictures which show the ingredients and utensils, so you can quickly see if you have everything you need in order to make this recipe. If we're interested we can press play to playback the recording. During playback the app goes full-screen in landscape mode, and we can touch the screen to pause or resume (Fig 2(c)). We see a series of images. We can swipe the screen to navigate to a different image and resume playback from that point. When we want to create a new recipe, we are prompted to enter a title (Fig 2(d)). Then we do a summary recording including name of the speaker, short bio, name of the recipe, something about it e.g. when it is cooked, for what ceremony, or in what season.

In field-testing early designs in rural Taiwan, we noticed that app users would touch the screen while talking. This motivated us to add a feature whereby we opportunistically capturing touch-screen gestures during recording. In the following section we describe the Image-Gesture-Voice method incorporating gesture.

Jobs: What does this persona want to do?	*Pains: What challenges does this persona face?*	*Gains: How might the app help?*
• cook for friends in traditional style • reconnect with traditional culture • connect with grandmother • document family culinary tradition so she can pass it on	• doesnt know how to make grandmothers recipe • doesnt have way to share grandmothers recipe with friends that captures whole process • cannot carry around all the recipes of her family's heritage • cannot easily compare ingredients, different methods	• provides a template for documenting grandmothers recipes including photos • allows capturing alternate versions of same recipe for comparing • provides a series of cherished recipes, told in the voice of an older relative • makes it possible to compare her traditional recipe with that of friends families

Figure 1: Value Proposition Table, enumerating jobs, pains, and gains from the standpoint of a persona (here, Taos), to better understand why someone would want to use an app



(a) Home: popular recipes.

(b) Recipe summary view.

(c) Playback: audio slideshow.

(d) Audio metadata.

Figure 2: Initial app design involving pen-and-paper drawings for four screens.

5. Image-Gesture-Voice

We propose a new Image-Gesture-Voice (IGV) documentary method, extending Basic Oral Language Documentation (BOLD) with images and touch-gestures. The method was motivated by the use case of documenting procedural discourse by recording spoken language linked to a series of still image prompts. IGV consists of separate recording and annotation activities. We first describe the recording activity via the use case of documenting a cooking recipe.

Before we begin recording, we obtain a series of still images. Photos can be taken before, during and after the food preparation activity. We might begin by photographing the ingredients and cooking utensils, then take photos of each step of preparation, and finally an image of the completed dish. A benefit of this approach is that it removes the pressure of spoken performance from the procedure.

The IGV record activity displays a slideshow of images. Recording can be initiated, paused, and resumed. When recording, only forward navigation to the next image is possible. When paused, the user can navigate forwards and backwards between images. When beginning or resuming a recording, the slideshow will seek to the first image that has not been discussed. The recording is completed when the user records to the final image.

When recording, the user's gestures on the current image are captured, and visual feedback is given. The same visual effect is used during playback. Our implementation uses a 'particle effect' which serves to reducee the precision of a gesture, and to enhance the sense of region, and of motion.[4] Capturing voice and gesture inputs simultaneously enables a dual expressive modality. Speech and gesture are semiotic resources that speakers can coordinate (Kendon, 2004; Kendon, 2008), and which have been found to boost precision of referencing in user interfaces (Bolt, 1980). For example, a user might gesture over a bowl of ingredients in a circular motion while describing the mixing action. This data may increase the precision of linking appropriate 'mixing' verbs with the audio signal. Aside from referentiality, the gesture gives additional information such as the speed and direction of mixing.

The basic layout of the IGV differs for landscape and portrait view ports. In landscape, we recommend showing three images, previous, current and next, with the previous and next images only being partially displayed. In portrait view, the current image is best displayed full width, accompanied by a three-image row of previous, current and next images beneath the main image. The previous and next images are important for navigation, including selecting the next image while recording.

After accepting a completed recording, IGV begins review playback. During playback the slides will advance automatically, and recorded gestures will be displayed on screen. If the user selects slides during playback, the audio will seek to the appropriate point. Finally, the user has the

[4]We have made available a generic web-based gesture recording and visualisation library called Gestate which was designed specifically for this purpose

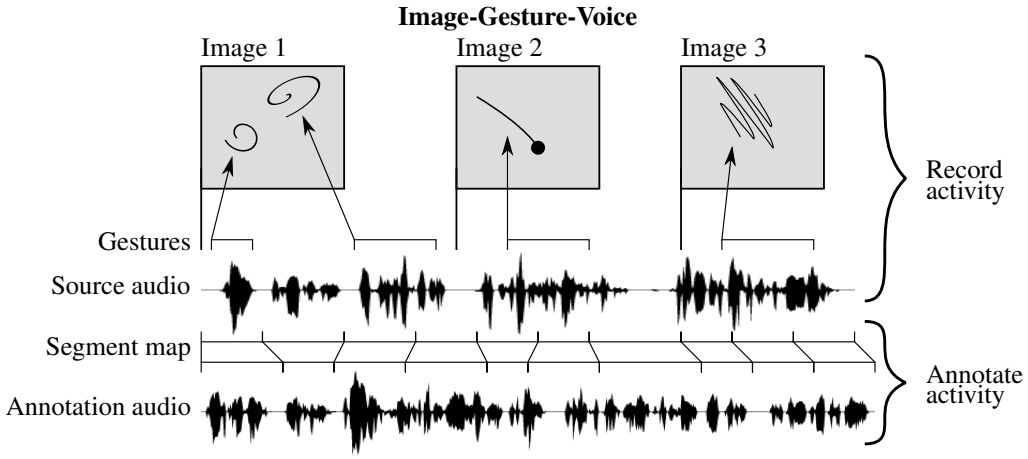


Figure 3: The IGV record activity aligns image prompts and touch gestures to the source audio signal. The annotate activity results in a segment map of source audio segments to annotation audio segments.

option to re-do (clear) or accept the recording.

The second IGV activity is for producing phrase-by-phrase oral annotations (Bird, 2010; Reiman, 2010). A common use case is performing a respeaking or a spoken translation. The activity is accomplished by a series of alternating play and record actions. During source playback, the prompt image will change automatically and gestures are displayed with the same visual effects as seen in the IGV record activity. The resulting data consists of an audio file and a segment map which associates source audio playback spans with recorded audio spans, see Figure 3.

Our mobile implementations have utilised separate playback (left) and record (right) controls. The operator plays by pressing and holding the play button, pauses by lifting off, replays by re-pressing play. Similarly, recording is a press and hold action, which also allows for momentary pausing. Play and record are alternated until all of the source audio has been played.

We may wish to play back the respeaking or translation with time-aligned images and gestures. We are able to display images at the correct time thanks to the segment map from the IGV annotation activity. However gestures occur within the image spans, and the operator may refer to things in a different order, especially when translating into a language that has a different word order. The only true way to ensure good gesture alignment for audio annotations is to recapture gestures during the annotation activity, which goes beyond our current implementation.

In the following sections we describe two implementations of IGV. Section 6. presents Zahwa, an Android app which implements both IGV activities. Section 7. discusses our recent efforts to implement IGV as open source Web Components.

6. The Zahwa App

This section presents a production quality Android app, called Zahwa, along with discussion of Web technologies and hybrid mobile app development. Zahwa has been co-designed and field tested with speakers of the endangered Austronesian language Saisiyat in rural North Western Taiwan. The app has also been field tested with indigenous Australian communities in Far North Australia. The app, and project web site, is offered in English and Traditional Chinese to support these fieldwork projects.

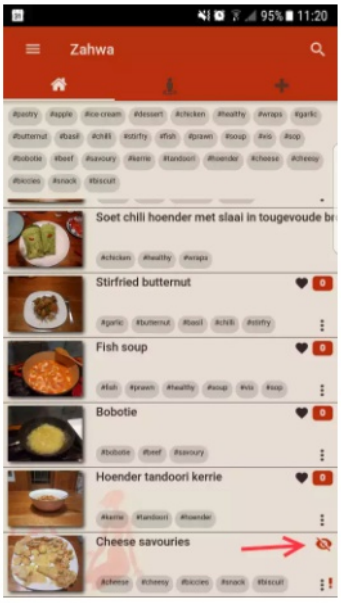
On first run, a new Zahwa user authenticates using their phone's existing Google or Facebook account. They are then shown a user agreement and given the opportunity to create a user display name. A pop-out menu provides access to features such as editing the user's profile (setting languages, taking a new photo), app settings, and data back-up/restore.

Zahwa's recipe record process is an implementation of the IGV method described in Section 5.. Creating a recipe is a three-step process; importing images, recording audio, and finalizing with descriptive metadata (see Figure 4(b)). The process of importing images requires selecting photos from the phone's gallery and putting them in sequence. The second step prompts for the language then proceeds to an IGV record activity, shown in Figure 4(c)). A finalization tasks has the user provide typed or spoken (speech-to-text) metadata including the recipe name, optional description and series of tags.

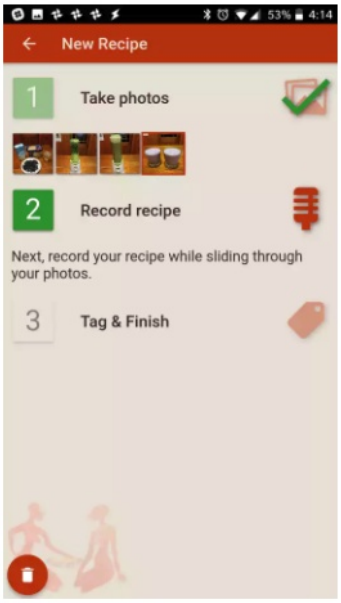
The Zahwa app is structured on three views accessed via UI 'tabs'. The rightmost *kitchen* view is a workflow management view where incomplete recipes appear. Creating a recipe is a multi-stage process and we handle these as stateful asynchronous tasks that users return to when convenient. The role of the *kitchen* is to guide the authoring of recipes through to a *minimally complete* form suitable for social interaction.

Completed recipes are moved from the kitchen view to the *social* view on the leftmost default tab (see Figure 4(a)). Here the user's recipes appear alongside downloaded recipes from other users. Recipes displayed in this view are presented with UI measures which draw attention to further actions we encourage users to perform, such as adding metadata including tags, and performing oral translations.

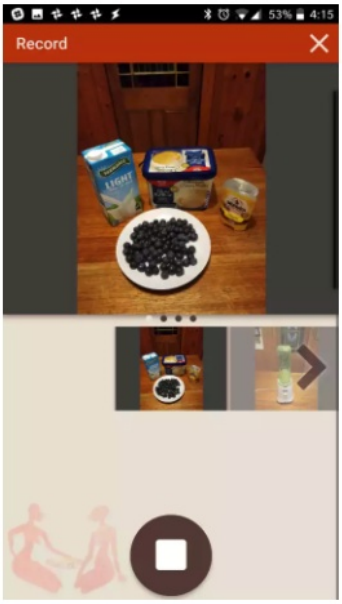
The *social* view also encourages the user to publish the recipe, so it can be found by others via in-app searches or



(a) Summary Screen: map the location of participants



(b) After importing photos...



(c) Recording a recipe...

Figure 4: The Zahwa app

the via the project website.[5] This view also allows users to share recipes,by launching the phone's native sharing activity, allowing the user to share via any installed app on the phone, including email, Facebook, text message and so on. The sharing scheme is based on a unique URL which also specifies the chosen translation for playback. The URL launches a mobile web app which fetches the recipe assets and plays the recipe in a desktop browser or on a mobile phone. Unpublished recipes can be shared by this unique URL, but are otherwise not discoverable (a feature inspired by YouTube).

An optional activity accessed via the social view is producing oral annotations, see Figure 5. Translations can be performed by other app users, such as bilingual speakers who want to make recipes more widely accessible.

The *map* view (middle tab) displays a geographical map with nearby recipes appearing as pins. Search is possible from any tab via the top right search icon. Zahwa adheres to a principle of being meaningfully usable when offline, including the ability to record recipes, and to search for recipes. The app caches nearby recipe metadata including thumbnail images. Users can find recipes by metadata and mark recipes for download when they have an internet connection. Similarly, the users actions such as publishing recipes are deferred until a network becomes available.

In field testing earlier prototypes we noticed that app users somewhat reluctantly re-oriented a phone to landscape if a particular view required it. This motivated us to support both portrait and landscape use. The IGV implementations adjust to the different aspect ratios by reorganising the position of buttons, and using different slide layout schemes as discussed in Section 5. (see also (Bettinson and Bird, 2017)). Considering that vision-impaired elders are particular candidates for apps of this type, running the app on an inexpensive Android tablet in landscape mode results in a suitably large image, about the size of a standard photograph.

6.1. The Web Technology Stack

The web technology stack allows us to deploy a common software component model across all platforms including Android, iPhone and the web. By web technologies, we refer to a single-page web app delivering rendered by a web browser engine. Zahwa is the latest product of a sustained program of research into web technologies that began with Aikuma-NG (Bettinson and Bird, 2017) in 2016.

Web technologies have been advancing at a dizzying pace. In the last few years there have been major advances in web browser APIs, JavaScript language specification[6], and the vibrant ecosystem of JavaScript 'frameworks'. Just two years ago, JavaScript frameworks suffered from significant performance issues and there was a lack of robust tooling in support of larger software projects. The situation today is much improved. Technology giants such as Google have done much to improve the capabilities of web browser APIs, and by extension the user experience of web apps.

The vibrancy of the JavaScript framework ecosystem is both a strength and a weakness. The ecosystem has arisen to support the industry of web app designers which tend to be concerned with short term projects. Remaining abreast of web technologies is a significant burden oft described by web developers as 'JavaScript fatigue'. There is a risk that the burden could exceeds the effort of developing two different apps via the Android and iOS native SDKs (or three, including the web).

We have minimized the web stack maintenance cost by

[5] http://zahwa.aikuma.org (includes a detailed how-to section)

[6] More accurately ECMAscript, as the standard is known

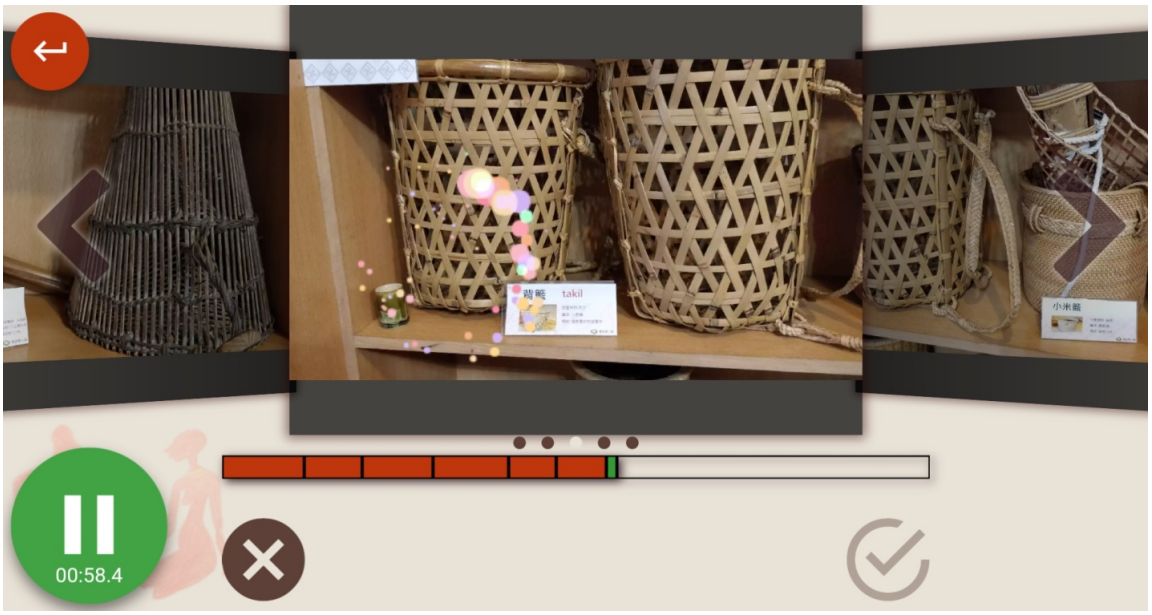


Figure 5: Zahwa's translation activity, e.g. IGV annotation. The source is a Saisiyat elder describing traditional crafts. Touch gesture particle effects are visible around the back basket (takil) currently being described. This activity was used to translate into Chinese Mandarin for the Taiwanese audience.

adopting the Ionic framework,a complete solution for deploying web and hybrid mobile applications. Ionic is based on the Angular JavaScript framework, Typescript, and a library of components that replicate common native UI elements for Android and iOS. Ionic includes 'tooling' for a complex web app, and integrates Apache Cordova to build installable hybrid mobile apps discussed in the next section. Frameworks are usually an architectural choice that applies app-wide. The risk is therefore that we might create components that require one of the several popular frameworks, such as Angular in the case of Ionic. We would hope to be able to recycle major software components to reduce the burden of developing the next app. This is hampered somewhat by frequent breaking changes in major revisions of JavaScript frameworks.

Thankfully web standards such as browser APIs are far more stable. An alternative emerging pattern is one based on the Web Component standard discussed further in Section 7.. However in the next section we discuss the more general mobile implementation pattern of hybrid mobile apps.

6.2. Hybrid Mobile

Apache Cordova is an open source tool chain that produces a native installable application which can be published and installed from app stores. However the the app views themselves are effectively web apps rendered by the mobile phone's native web view API. The resulting *Hybrid mobile* allow crafting of apps by using the same technologies as web apps, thus ensuring cross-platform compatibility across the mobile platforms and desktop web.

Cordova also offers a plug-in architecture which allows us to call native features of the mobile platform. One use case is utilising the mobile platform's helpful content sharing activity, which is automatically aware of the user's installed sharing-capable apps such as email, Facebook, Google+ or even text messages and so forth. Using native mobile features can also result in a better user experience than the web API alone. An example here is the image picker plug-in that Zahwa uses to import photos.

The hybrid development model is not without drawbacks. The Cordova tool-chain is complex and occasionally unreliable. The open source ecosystem of plug-ins is sadly mired by a large number of abandoned repositories. In one case we required a clustering feature in the Google Map plug-in used for recipe discovery in Zahwa. We required a modification so we could override the default click-to-zoom behaviour to display a list of recipes. The project maintainer was unwilling to accept a pull request, forcing us to fork the plug-in and creating a new dependency with a maintenance burden.

That said, our experience of the hybrid app software pattern is broadly positive and is a world apart in terms of development productivity. The Zahwa app demonstrates it's possible to build apps which are virtually indistinguishable from native apps. We should also point out that pure web apps, e.g. web sites rather than installed apps, are a viable proposition in many use cases. On Android, these can operate offline and full-screen just like 'real' apps, but offer the benefit of a very rapid onboard process where sharing of a URL is all one needs to recruit a participant.

7. Web Component Implementation

Web Components (WC) are a live web standard[7] that has been fully implemented in Chrome, with Firefox still in progress. All remaining browsers can be 'polyfilled' to support WC now until they finish their native implementations, which should be complete by the end of 2018. WC offers a number of advantages including performance and interoperability regardless of JavaScript frameworks.

[7] https://github.com/w3c/webcomponents

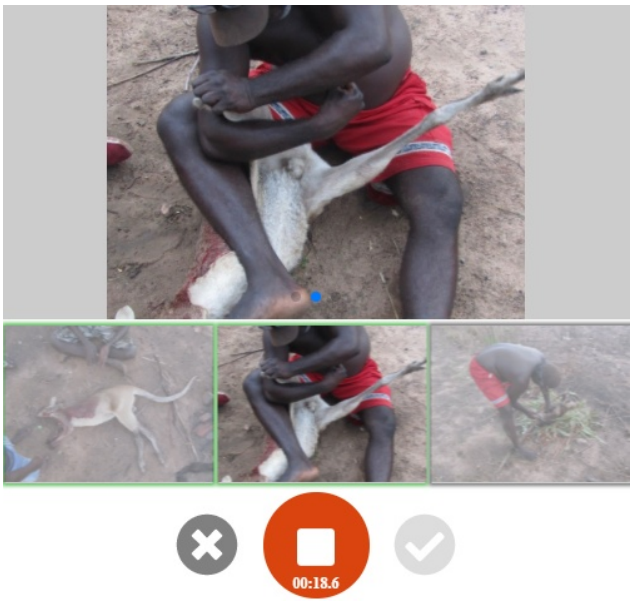


Figure 6: A Web Component implementation of the IGV record activity. Used here to document preparation of scrub wallaby in Arnhem Land, North Australia.

The typical approach for modularity in the web domain depends on directly low-level manipulation of the web browser DOM API. One example is the Dragula[8] drag 'n drop library which Zahwa uses to facilitate reordering of images. Dragula proved to be problematic given the multi-layered interface of an app, where the drag 'n drop component was merely the top most layer.

Part of the WC specification is the *Shadow DOM*, an isolated a DOM scope encapsulated within the web component. We are then free to scaffold our app with any JavaScript framework, without unwanted side-effects elsewhere in the app. Stencil[9] is a 'compiler' of Web Components, developed by the makers of the Ionic platform. Stencil allows us to continue to use current best-practices in web development, while producing standardised Web Components.

We used Stencil to implement the IGV recording and annotation activities as Web Components, see Figure 6. In the remainder of this section we demonstrate including the IGV WC on a web page, and interacting with the component via a JavaScript API.

```
1 <html>
2 <head>
3 <script src='https://unpkg.com/aikuma@0
    .0.1/dist/aikuma.js'></script>
4 </head>
5 <body>
6 <aikuma-image-gesture-voice></aikuma-image-
    gesture-voice>
7 </html>
```

The general principle is that Web Components may be selected like any other HTML element. They can also register public methods on the element. To illustrate, the following listing is an asynchronous JavaScript function:

```
1 async function doIGV() {
2   let wc = document.queryselector('aikuma-
        image-gesture-voice')
3   wc.loadFromImageURLs(['image1'...])
4   let results = await wc.waitForComplete()
5   console.log('IGV returned', results)
6 })
```

Line 2 is a standard Web DOM API call to find the first tag that matches the selector. On line 3, *wc* is now the Web Component instance and we invoke a method specific to this Web Component which imports a series of images by providing a list of URLs. Line 4 uses another method which returns a JavaScript Promise, thus we use the await keyword to yield execution back to the main thread until the Promise resolves. When the user has cancelled or completed the activity, the code block in the *then* statement is executed, printing the data structure to the web browser console.

WCs as plain JavaScript modules works particularly well with the existing JavaScript ecosystem. We have published the IGV components on Node Package Manager (NPM) repository under the @aikuma scope[10]. Please refer to the GitHub documentation for a full description of methods and IGV data structures. We also plan to release generic JavaScript libraries helpful for knowledge preservation apps.

8. Conclusion

We aim to develop mobile software that guides users through a series of steps to preserve their knowledge in the digital domain. The mobile platform offers the means for scaling up this preservation activity. We have described a general Image-Gesture-Voice documentary method and shown how it can be instantiated in a social app for preserving and sharing know-how.

Web technologies boost development productivity yet force us to engage with a complex software stack that is in a continual state of flux. This paper is a case in point: we have just built a production quality tool yet must anticipate the next implementation pattern. Thankfully, the situation is improving thanks to Web Components. As a official web standard, Web Components will have longer lifecycles than components built on this month's most popular JavaScript framework.

The time is ripe to build a common library of Web Components and other JavaScript libraries to reduce duplication and help the small community of people developing software for language documentation to focus our limited resources on well-engineered and field-refined components. We encourage others to apply and extend the work presented here, and to contribute to the improvement of software components to support the vision of scaling up digital preservation of cultural knowledge.

Acknowledgements

This research was supported by NSF grant 1464553 *Language Induction meets Language Documentation: Leveraging bilingual aligned audio for learning and preserving*

[8] https://bevacqua.github.io/dragula/

[9] https://stenciljs.com/

[10] http://www.aikuma.org/components.html

languages. We are grateful to Amina Mettouchi for supplying details of the Taos and Zahwa personas, and to Alexandra Marley for the scrub wallaby images.

9. References

Bettinson, M. and Bird, S. (2017). Developing a suite of mobile applications for collaborative language documentation. In *Second Workshop on Computational Methods for Endangered Languages*, pages 156–164.

Bettinson, M. (2017). Crafting the next generation of language documentation tools. Fifth International Conference on Language Documentation and Conservation.

Bird, S., Hanke, F. R., Adams, O., and Lee, H. (2014). Aikuma: A mobile app for collaborative language documentation. In *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 1–5.

Bird, S. (2010). A scalable method for preserving oral literature from small languages. In *Proceedings of the 12th International Conference on Asia-Pacific Digital Libraries*, pages 5–14.

Bird, S. (2018). Desiging mobile applications for documenting endangered languages. In Kenneth Rehg et al., editors, *Oxford Handbook on Endangered Languages*. Oxford University Press.

Blachon, D., Gauthiera, E., Besacier, L., Kouaratab, G.-N., Adda-Decker, M., and Rialland, A. (2016). Parallel speech collection for under-resourced language studies using the LIG-Aikuma mobile device app. In *Proceedings of the Fifth Workshop on Spoken Language Technologies for Under-resourced languages*, volume 81, pages 61–66.

Bolt, R. A. (1980). "put-that-there": Voice and gesture at the graphics interface. In *Proceedings of the 7th Annual Conference on Computer Graphics and Interactive Techniques*, volume 14, pages 262–270. ACM.

Chatzimilioudis, G., Konstantinidis, A., Laoudias, C., and Zeinalipour-Yazti, D. (2012). Crowdsourcing with smartphones. *IEEE Internet Computing*, 16:36–44.

Chikonzo, A. (2006). The potential of information and communication technologies in collecting, preserving and disseminating indigenous knowledge in Africa. *The International Information and Library Review*, 38(3):132–138.

Goldman, J.-P., Leemann, A., Kolly, M.-J., Hove, I., Almajai, I., Dellwo, V., and Moran, S. (2014). A crowdsourcing smartphone application for Swiss German: Putting language documentation in the hands of the users. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 3444–47.

Himmelmann, N. P. (1998). Documentary and descriptive linguistics. *Linguistics*, 36:161–195.

Johnson, H. and Aristar Dry, H. (2002). OLAC discourse type vocabulary. `http://www.language-archives.org/REC/discourse.html`.

Kendon, A. (2004). *Gesture: Visible action as utterance*. Cambridge University Press.

Kendon, A. (2008). Some reflections on the relationship between gestureand sign. *Gesture*, 8(3):348–366.

Leemann, A., Kolly, M.-J., Purves, R., Britain, D., and Glaser, E. (2016). Crowdsourcing language change with smartphone applications. *PloS one*, 11(1):e0143060.

Leemann, A., Kolly, M.-J., and Britain, D. (2018). The English Dialects App: The creation of a crowdsourced dialect corpus. *Ampersand*, 5:1–17.

Lüpke, F. (2010). Research methods in language documentation. *Language documentation and description*, 7:55–104.

Mettouchi, A., Bettinson, M., and Bird, S. (2017). Documenting recipes. Fifth International Conference on Language Documentation and Conservation.

Norman, D. A. and Draper, S. W. (1986). User centered system design. *New Perspectives on Human-Computer Interaction*, 3.

Oh, S. and Syn, S. Y. (2015). Motivations for sharing information and social support in social media: A comparative analysis of Facebook, Twitter, Delicious, YouTube, and Flickr. *Journal of the Association for Information Science and Technology*, 66(10):2045–2060.

Pollock, N. (2011). The language of food. In Nick Thieberger, editor, *Oxford Handbook of Linguistic Fieldwork*, pages 235–49. Oxford University Press.

Reiman, W. (2010). Basic oral language documentation. *Language Documentation and Conservation*, 4:254–268.

Subramani, M. R. and Rajagopalan, B. (2003). Knowledge-sharing and influence in online social networks via viral marketing. *Communications of the ACM*, 46(12):300–307.

Woodbury, A. C. (2010). Language documentation. In Peter K. Austin et al., editors, *The Cambridge Handbook of Endangered Languages*. Cambridge University Press.