

Developing without Developers

Software Development Methods for LD Apps

Luke D. Gessler
lgessler.com

Department of Linguistics
Georgetown University

ComputEL-3
February 26, 2019



Outline

- 1 Introduction
- 2 The State of LD Apps
- 3 Software Development Methods
- 4 Experiment: Cloning ELAN
- 5 Conclusion



Introduction

- Central claim: language documentation (LD) apps will succeed when their developers use **software development methods** that are **adapted** to the **economic conditions** of language documentation



Introduction

- Central claim: language documentation (LD) apps will succeed when their developers use **software development methods** that are **adapted** to the **economic conditions** of language documentation
- Debatable assumptions:
 - Documentation is worthwhile even without revitalization efforts
 - Software can help fieldworkers be more productive
 - (Worth discussing, but out of scope)



Introduction

- Central claim: language documentation (LD) apps will succeed when their developers use **software development methods** that are **adapted** to the **economic conditions** of language documentation
- Debatable assumptions:
 - Documentation is worthwhile even without revitalization efforts
 - Software can help fieldworkers be more productive
 - (Worth discussing, but out of scope)
- Will necessarily critique existing LD apps. Extant apps are very impressive, not saying I could do better



Introduction

- Central claim: language documentation (LD) apps will succeed when their developers use **software development methods** that are **adapted** to the **economic conditions** of language documentation
- Debatable assumptions:
 - Documentation is worthwhile even without revitalization efforts
 - Software can help fieldworkers be more productive
 - (Worth discussing, but out of scope)
- Will necessarily critique existing LD apps. Extant apps are very impressive, not saying I could do better
- Will only consider “traditional” LD workflows



Introduction

- Central claim: language documentation (LD) apps will succeed when their developers use **software development methods** that are **adapted** to the **economic conditions** of language documentation
- Debatable assumptions:
 - Documentation is worthwhile even without revitalization efforts
 - Software can help fieldworkers be more productive
 - (Worth discussing, but out of scope)
- Will necessarily critique existing LD apps. Extant apps are very impressive, not saying I could do better
- Will only consider “traditional” LD workflows
- Roadmap:
 - The state of LD apps
 - Development in industry vs. academia
 - Experiment
 - Conclusion



Outline

- 1 Introduction
- 2 The State of LD Apps**
- 3 Software Development Methods
- 4 Experiment: Cloning ELAN
- 5 Conclusion



LD apps today

- Many fieldworkers use generic “apps” for most of their workflow
 - E.g. Excel, paper, ...
 - No computer assistance in time-intensive tasks like transcription, glossing, concordancing



LD apps today

- Many fieldworkers use generic “apps” for most of their workflow
 - E.g. Excel, paper, ...
 - No computer assistance in time-intensive tasks like transcription, glossing, concordancing
- Others use a combination of linguistic apps
 - E.g. FLEx, ELAN, Praat, ...
 - Not always easy to move data from one into the other
 - Difficult to maintain a source of truth



LD apps today

- Many fieldworkers use generic “apps” for most of their workflow
 - E.g. Excel, paper, ...
 - No computer assistance in time-intensive tasks like transcription, glossing, concordancing
- Others use a combination of linguistic apps
 - E.g. FLEx, ELAN, Praat, ...
 - Not always easy to move data from one into the other
 - Difficult to maintain a source of truth
- Ideal: a **comprehensive** LD app
 - Built-in support for most tasks
 - No shuffling data around between apps



Why haven't comprehensive LD apps taken over?

- Successful comprehensive apps have been made



Why haven't comprehensive LD apps taken over?

- Successful comprehensive apps have been made
- Still, none has gained very widespread use



Why haven't comprehensive LD apps taken over?

- Successful comprehensive apps have been made
- Still, none has gained very widespread use
- Common complaints:
 - “X is too frustrating/hard to learn” ! **usability issues**
 - “I can't use X because it doesn't realize my language's diacritics are making lexical distinctions” ! **missing/incomplete features**
 - “I can't use X because I don't use Windows” ! **platform limitations**



Why haven't comprehensive LD apps taken over?

- Successful comprehensive apps have been made
- Still, none has gained very widespread use
- Common complaints:
 - “X is too frustrating/hard to learn” ! **usability issues**
 - “I can't use X because it doesn't realize my language's diacritics are making lexical distinctions” ! **missing/incomplete features**
 - “I can't use X because I don't use Windows” ! **platform limitations**
- LD apps are not a new idea—why are these still issues?



Why haven't comprehensive LD apps taken over?

- Successful comprehensive apps have been made
- Still, none has gained very widespread use
- Common complaints:
 - “X is too frustrating/hard to learn” / usability issues
 - “I can't use X because it doesn't realize my language's diacritics are making lexical distinctions” / missing/incomplete features
 - “I can't use X because I don't use Windows” / platform limitations
- LD apps are not a new idea—why are these still issues?
- One major reason: development methods in LD have been copied from industry



Outline

- 1 Introduction
- 2 The State of LD Apps
- 3 Software Development Methods**
- 4 Experiment: Cloning ELAN
- 5 Conclusion



Developing software in industry vs. academia

- For software development, industry has more of every resource than academia



Developing software in industry vs. academia

- For software development, industry has more of every resource than academia
- A typical industrial app has:
 - Lots of money
 - Multiple full-time professionals
 - Veteran (5+ yrs) developers
 - Support staff





Developing software in industry vs. academia

- For software development, industry has more of every resource than academia
- A typical industrial app has:
 - Lots of money
 - Multiple full-time professionals
 - Veteran (5+ yrs) developers
 - Support staff
- A typical academic app has:
 - A professor overseeing it (?)
 - Fixed-term funding at best
 - A graduate student or two with moderate programming experience





Developing software in industry vs. academia

- For software development, industry has more of every resource than academia
- A typical industrial app has:
 - Lots of money
 - Multiple full-time professionals
 - Veteran (5+ yrs) developers
 - Support staff
- A typical academic app has:
 - A professor overseeing it (?)
 - Fixed-term funding at best
 - A graduate student or two with moderate programming experience
- ...and the graduate student graduates

Latest commit b12c2a2 on Dec 8, 2011

7 years ago

7 years ago

7 years ago

7 years ago

7 years ago

7 years ago



Adapting development methods

- The difference in economic conditions is obvious
- Less obvious: development practices from *industry* are suited to *industry*



Adapting development methods

- The difference in economic conditions is obvious
- Less obvious: development practices from *industry* are suited to *industry*
- LD apps would succeed more if methods were aligned with conditions



Adapting development methods

- The difference in economic conditions is obvious
- Less obvious: development practices from *industry* are suited to *industry*
- LD apps would succeed more if methods were aligned with conditions
- Hypothesis: to alleviate the labor shortage, LD app developers need to carefully choose **tools** (programming language, platform, libraries, etc.) which enhance their **productivity**
 - As opposed to performance, ease of use, security, correctness, :::



Acting on the hypothesis

Hypothesis: to alleviate the labor shortage, LD app developers need to carefully choose **tools** (programming language, platform, libraries, etc.) which enhance their **productivity**

- 1 Prefer libraries over writing new code
 - There's probably a library for it
 - Recent (<5y ago) explosion in availability of high-quality libraries
 - NPM (JavaScript library index) has over **400,000** libraries



Acting on the hypothesis

Hypothesis: to alleviate the labor shortage, LD app developers need to carefully choose **tools** (programming language, platform, libraries, etc.) which enhance their **productivity**

- 1 Prefer libraries over writing new code
 - There's probably a library for it
 - Recent (<5y ago) explosion in availability of high-quality libraries
 - NPM (JavaScript library index) has over **400,000** libraries
- 2 Choose libraries that minimize programming (development, maintenance)
 - Time spent adjusting for breaking changes is “useless” to users
 - Choose stable languages and libraries



Acting on the hypothesis

Hypothesis: to alleviate the labor shortage, LD app developers need to carefully choose **tools** (programming language, platform, libraries, etc.) which enhance their **productivity**

- 1 Prefer libraries over writing new code
 - There's probably a library for it
 - Recent (<5y ago) explosion in availability of high-quality libraries
 - NPM (JavaScript library index) has over **400,000** libraries
- 2 Choose libraries that minimize programming (development, maintenance)
 - Time spent adjusting for breaking changes is “useless” to users
 - Choose stable languages and libraries
- 3 Choose libraries that are especially designed for the challenges of our domain (LD)
 - No internet
 - Multi-user collaboration
 - ...



Outline

- 1 Introduction
- 2 The State of LD Apps
- 3 Software Development Methods
- 4 Experiment: Cloning ELAN**
- 5 Conclusion



An informal experiment

- Goal: measure development time of an app, gauge whether careful tool choice helped
- Hypothesis has to do with *programming*, not design
! used an existing design



An informal experiment

- Goal: measure development time of an app, gauge whether careful tool choice helped
- Hypothesis has to do with *programming*, not design / used an existing design
- Chose to clone a **small subset** of ELAN
 - Time-aligned annotation
 - Used widely in LD and linguistics
 - Old, proven design
 - Moderately complex

ELAN 5.3 - elan-example3.eaf

File Edit Annotation Tier Type Search View Options Window Help

Grid Text Subtitles Lexicon Comments Recognizers Metadata Co

Volume: 100

0 50

elan-example1.mpg 0 25 50

Mute Solo 0

elan-example1.wav 0 25 50

Mute Solo 0

Rate: 00:00:06.822 Selection: 00:00:06.822 - 00:00:07.155 348

00:00:06.822 00:00:07.155 348

Selection Mode Loop Mode

elan-example1

00:00:06.000 00:00:06.000 00:00:07.000 00:00:08.000 00:00:09.000

K-SpCh
IT and then you go the other, Saint Anna Straat to this to the center of the town, to this big rotunde.

W-SpCh
IT and [then] you [go] [the] [other] Saint Ann[]Straat [to] [this] [to] [the] center [of] [th] town [to] [th] big

W-Words
IT con adv pro v art adj n n n prep dem prep art n pr art n pr de adj

W-POS
IT and den ju. gao dt otc sant ana stra to ois to da sente af da taun fu ois big rtunde

W-IPA
IT

W-RGU
IT preparation | stroke | hold | part retraction | hol prepar|stroke|preparation | stroke

W-RGph
IT

W-ROMe
IT Going along St. , Going , A roundat

K-RGU
IT

K-RGph
IT

K-ROMe
IT



The tools

- Platform: web browser
 - Solves portability, installation problems
 - Browser is stable after a decade of explosive growth



The tools

- Platform: web browser
 - Solves portability, installation problems
 - Browser is stable after a decade of explosive growth
- Programming language: ClojureScript
 - A Lisp-family language with elegant but practical abstractions
 - Very stable: code from 2009 runs unaltered
 - Compiles to JavaScript



The tools

- Platform: web browser
 - Solves portability, installation problems
 - Browser is stable after a decade of explosive growth
- Programming language: ClojureScript
 - A Lisp-family language with elegant but practical abstractions
 - Very stable: code from 2009 runs unaltered
 - Compiles to JavaScript
- Database: PouchDB
 - Allows the app to be used [offline](#) even though it's in a web browser



The tools

- Platform: web browser
 - Solves portability, installation problems
 - Browser is stable after a decade of explosive growth
- Programming language: ClojureScript
 - A Lisp-family language with elegant but practical abstractions
 - Very stable: code from 2009 runs unaltered
 - Compiles to JavaScript
- Database: PouchDB
 - Allows the app to be used [offline](#) even though it's in a web browser
- Interface: Material-UI (React)
 - Expansive, high-quality collection of off-the-shelf, [mobile-compatible](#) UI components
- See paper or talk to me for more details



Results

- Check it out!
<https://lgessler.com/ewan/>



Results

- Check it out!
<https://lgessler.com/ewan/>
- Created a **view-only subset** of ELAN (**not for real use!**)
 - Development completed in only 3 40-hour weeks
 - Offline support achieved
 - Sync with other users provided for free by the database
 - Real-time collaboration greatly facilitated by the database chosen



Results (contd.)

- Check it out! <https://lgessler.com/ewan/>
- Comparing performance with how it would have gone with “standard” tools:
 - Offline support took almost no work
 - No browser-based LD apps offer this, possible but difficult to implement in a traditional architecture



Results (contd.)

- Check it out! <https://lgessler.com/ewan/>
- Comparing performance with how it would have gone with “standard” tools:
 - Offline support took almost no work
 - No browser-based LD apps offer this, possible but difficult to implement in a traditional architecture
 - Data sync provided for free by the database
 - A few LD apps offer sync, users are wary of it, reported to be buggy



Results (contd.)

- Check it out! <https://lgessler.com/ewan/>
- Comparing performance with how it would have gone with “standard” tools:
 - Offline support took almost no work
 - No browser-based LD apps over this, possible but difficult to implement in a traditional architecture
 - Data sync provided for free by the database
 - A few LD apps over sync, users are wary of it, reported to be buggy
 - Real-time collaboration greatly facilitated by database
 - No LD apps over real-time collaboration



Results (contd.)

- Check it out! <https://lgessler.com/ewan/>
- Comparing performance with how it would have gone with “standard” tools:
 - Offline support took almost no work
 - No browser-based LD apps over this, possible but difficult to implement in a traditional architecture
 - Data sync provided for free by the database
 - A few LD apps over sync, users are wary of it, reported to be buggy
 - Real-time collaboration greatly facilitated by database
 - No LD apps over real-time collaboration
 - Little expected maintenance thanks to very stable PL



Results (contd.)

- Check it out! <https://lgessler.com/ewan/>
- Comparing performance with how it would have gone with “standard” tools:
 - Offline support took almost no work
 - No browser-based LD apps over this, possible but difficult to implement in a traditional architecture
 - Data sync provided for free by the database
 - A few LD apps over sync, users are wary of it, reported to be buggy
 - Real-time collaboration greatly facilitated by database
 - No LD apps over real-time collaboration
 - Little expected maintenance thanks to very stable PL
 - UI library provided almost all components needed, only a few needed to be created from scratch
 - Non-browser platforms often require much more work, even LD apps on the browser tend not to use as many libraries as they could
- “Experiment” is obviously subjective and limited



Results (contd.)

- Check it out! <https://lgessler.com/ewan/>
- Comparing performance with how it would have gone with “standard” tools:
 - Offline support took almost no work
 - No browser-based LD apps over this, possible but difficult to implement in a traditional architecture
 - Data sync provided for free by the database
 - A few LD apps over sync, users are wary of it, reported to be buggy
 - Real-time collaboration greatly facilitated by database
 - No LD apps over real-time collaboration
 - Little expected maintenance thanks to very stable PL
 - UI library provided almost all components needed, only a few needed to be created from scratch
 - Non-browser platforms often require much more work, even LD apps on the browser tend not to use as many libraries as they could
- “Experiment” is obviously subjective and limited
- That said, hypothesis seemed corroborated: providing similar features with standard tools would have taken much longer



Outline

- 1 Introduction
- 2 The State of LD Apps
- 3 Software Development Methods
- 4 Experiment: Cloning ELAN
- 5 Conclusion**



Conclusion

- Changing software development methods is key to progress in making comprehensive LD apps succeed because LD apps have so little labor available to them
- LD app developers need to optimize for **productivity** in their tools
- My three proposed commandments:
 - ① Prefer libraries over writing new code
 - ② Choose libraries that minimize programming
 - ③ Choose libraries that are especially designed for the challenges of LD