<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>Don't Believe The Hype! White Lies of Conversational User Interface Creation Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authors(s)</strong></td>
<td>Rough, Daniel; Cowan, Benjamin R.</td>
</tr>
<tr>
<td><strong>Publication date</strong></td>
<td>2020-07-24</td>
</tr>
<tr>
<td><strong>Conference details</strong></td>
<td>The 2nd International Conference on Conversational User Interfaces (CUI 2020), Bilbao, Spain, 23-24 July 2020</td>
</tr>
<tr>
<td><strong>Publisher</strong></td>
<td>ACM</td>
</tr>
<tr>
<td><strong>Link to online version</strong></td>
<td><a href="https://cui2020.com/">https://cui2020.com/</a></td>
</tr>
<tr>
<td><strong>Item record/more information</strong></td>
<td><a href="http://hdl.handle.net/10197/12031">http://hdl.handle.net/10197/12031</a></td>
</tr>
</tbody>
</table>
Don’t Believe The Hype! White Lies of Conversational User Interface Creation Tools

Daniel Rough
University College Dublin
Dublin, Ireland
daniel.rough@ucd.ie

Benjamin Cowan
University College Dublin
Dublin, Ireland
benjamin.cowan@ucd.ie

ABSTRACT
Following the initial hype and high expectations of conversational user interfaces (CUIs), a number of creation tools have emerged to simplify development of these complex systems. These have the potential to democratise and expand application development to those without programming skills. However, while such tools allow end-user developers to build language understanding and dialog management capability into a CUI application, actually fulfilling or executing an action still requires programmatic API integration. In this paper, we look at how CUI builders that claim to be “no code required” struggle to yield more than toy examples, with an aim to provoke the community to develop better tools for CUI creation.

CCS CONCEPTS
• Human-centered computing → Natural language interfaces; User interface design.

KEYWORDS
conversational user interfaces, chatbots, end-user programming

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM Reference Format:

1 INTRODUCTION
Conversational user interfaces (CUIs), either controlled through voice as Intelligent Personal Assistants (IPAs), or through text as chatbots, are regaining popularity after an initial slump caused by over-estimation of their potential capabilities [10]. Rather than relying on more social forms of conversation, which may not be desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and domestic contexts through simple, task-oriented adjacency-pair dialogues. For domestic use, these types of interactions are desirable [5], these assistants have found a niche in both commercial and dom
approaches to a CUI pipeline component require programming experience, or at least an understanding of semi-structured language scripting. Note that speech recognition/synthesis are not annotated in Figure 1, nor described in Table 1 as their use with any of the tools identified is limited to Google’s or Amazon’s provided services. These components would require significant speech technology expertise to tailor even if they were open-source.

Most commercial tools focus on supporting end-user developers in three key components of the CUI development pipeline; natural language understanding, dialog management, and natural language generation. The current state-of-the-art approach to tailoring language generation in the tools identified is a three-step process. One creates ‘intents’ - high-level actions a user would wish to perform, specifies example phrases that map to this intent, and annotates these phrases with ‘entities’ - parameters that customise the intent. Data-driven language models are trained on these phrases, such that no complex grammar creation is required. PandoraBots [14] still makes use of a grammar rule-based syntax, but all other commercial tools provide an intent-entity mapping user interface.

Tailoring the dialog management component often involves specifying transitions in the dialog state through a simple graphical user interface [2, 7, 9, 12], or less intuitive semi-structured scripting [3, 14, 15]. Major industry platforms (e.g., Google’s Dialogflow [7]; Amazon’s Lex [12]; IBM’s Watson Assistant [2]) take a straightforward frame-based approach to DM, managing context to ensure that all entities or ‘slots’ are filled prior to fulfilling an intent. This alleviates the need to implement any form of semi-structured language grammar.

In short, the tailoring of key CUI components is generally accessible to non-programmers in commercial tools. For instance, one can define a pizza ordering intent that recognises entities for size, crust type and toppings, a slot-filling dialog manager that prompts for missing values, and a parameterised confirmation response. Now our dialog is successfully completed, all we need is to actually place the order...and therein lies the trouble.

Figure 1: Traditional pipeline of CUIs and their implementation - adapted from [13]

3 NO CODE? NO! PROBLEM!

Despite bold claims of a no-code-required creation process, involving external services in a developed CUI application in any way - be it dimming a smart bulb or ordering pizza - requires programming. A CUI’s execution of a service is termed “fulfilment” and occurs when an intent has been matched and its requisite entities provided. DialogFlow and Lex currently offer integration functionality for deploying a CUI as a Google Action or Alexa Skill without the need for coding, but these still require programming experience to connect APIs together. From an end-user programming perspective, previous work has highlighted the difficulty that even enthusiastic end-users have in integrating new devices into their smart home setups [4, 6]. Without support for fulfilment, no-code-required tools can still be used to build FAQ answering CUIs. Yet for applications where some external service needs to be executed based on users’ commands, this lack of functionality is a major issue.

4 WHAT NEEDS TO BE DONE?

There are currently a plethora of CUI development tools available, but only really useful to those with the knowledge to implement back-end fulfilment and connect this with user utterances. The average end-user is instead limited to creating a front-end data-gathering dialog. This has the potential to be integrated with external services but, without programming, remains as a non-functional toy. We as a community need to drive the development of CUI creation tools with the following:

- Frank and honest descriptions of capabilities - unrealistic high expectations are a major cause of frustration and abandonment for these tools
- Straightforward integration with existing platforms for service integration such as IFTTT\(^1\). Such platforms abstract over code-based implementation for a wide (and growing) variety of web services
- No-code integration with speech recognition/synthesis services offered by Google or Amazon, so that developers who desire voice interaction are not locked into using the other components provided by these companies

Addressing these points will be a solid start to meeting the claims of “no code required” tools and consequently democratising CUI development by opening it to those without programming skills.

\(^1\)https://ifttt.com/
REFERENCES


