



# Educating Users to Formulate Questions in Q&A Platforms: A Scaffold for Google Sheets

Oscar Díaz and Jeremías P. Contell<sup>(✉)</sup>

ONEKIN Web Engineering Group,  
University of the Basque Country (UPV/EHU), San Sebastián, Spain  
{oscar.diaz, jeremias.perez}@ehu.eus

**Abstract.** Different studies point out that spreadsheets are easy to use but difficult to master. When difficulties arise, home users and small-and-medium organizations might not always resort to a help desk. Alternatively, Question&Answer platforms (e.g. Stack Overflow) come in handy. Unfortunately, we can not always expect home users to properly set good questions. However, examples can be a substitute for long explanations. This is particularly important for our target audience who might lack the skills to describe their needs in abstract terms, and hence, examples might be the easiest way to get the idea through. This sustains the effort to leverage existing spreadsheet tools with example-centric inline question posting. This paper describes such an effort for Google Sheets. The extension assists users in posing their example-based questions without leaving Google Sheets. These questions are next transparently channeled to Stack Overflow.

**Keywords:** Spreadsheets · Q&A platforms · Social computing  
StackOverflow

## 1 Introduction

*Home users* tend to learn “just enough” to keep going [9]. Learning as you go, ideally without consulting the manual, becomes a common practice. It comes as no surprise that these users are frequently faced with doubts about how to proceed. These doubts might be resolved through trial-and-error cycles or resorting to help desks. The former might be cumbersome and unsuccessful, while support for end users tends to be rather limited in most organizations. Indeed, different studies point to the lack of assistance as a major stumbling block for these users’ self-reliance [14, 24]. In this scenario, Question&Answer platforms (Q&A platforms) come in handy [10]. In Q&A sites, users (askers) post questions, and rely on other community members (answerers) to provide a suitable solution to their information needs [23]. Unfortunately, home users are not always aware of Q&A platforms or, more commonly, they are discouraged

by the upfront-cost of posing a question, and the risk of not getting a satisfying answer [11]. Findings confirm that (re)editing likelihood was larger for users with limited familiarity with either *Stack Overflow* or the topic at hand [25], both aspects characterizing our target audience.

The irony is that those who can benefit most from Q&A platforms might lack the skills (and support) to pose “good questions”. Studies found how successful inline posting correlates with affect, presentation quality or time [5]. This leads to our research question: *how can home users be assisted in successful question posting in Q&A platforms?*

Our main premise is that examples (and doubts) emerge while conducting tasks. Unfortunately, askers need to move away from the working environment (where questions arise) to the Q&A platform (where questions are posed). This requires askers to re-create the task scenario into the question. As an example, consider *Google Sheets* as the working environment, and *Stack Overflow* as the Q&A platform. When struggling with spreadsheet formulas, end users might resort to the *Stack Overflow* community. This involves moving to this Q&A platform and posting the question. This can be achieved in a rambling textual way, or conveniently formatted using *markdown* (i.e. a markup language for code snippets in *Stack Overflow*). More to the point, in accordance with *StackOverflow*’s own recommendations [18], screenshots or separated shared spreadsheet should be provided for respondents to more reliably reproduce the task scenario. This requires more involvement (and skills) on the asker’s side but, in return, it permits potential answerers not only to provide but also to validate their answers with the sample data. Indeed, the presence of code snippets in questions highly correlates with getting prompt and appropriate answers [5]. Hence, users need to balance the upfront investment (i.e. time/effort spent on writing the question) *versus* the risk of obtaining low-quality answers or no answer at all. The issue is that the upfront “attention capital” available for home users might be too limited.

To reduce this upfront investment, we resort to *inline question posting* through a question scaffold. Here, users can seamlessly channel questions to Q&A platforms without leaving their working environment. The aim: reducing the cost of posting questions while increasing the payoff, i.e. obtaining more accurate answers by making questions clearer through examples.

*QuestionSheet* is a research prototype we built to demonstrate the feasibility of this approach on top of *Google Sheets*. To use *QuestionSheet*, home users starts with an ordinary spreadsheet. Then, in cells where a formula is needed but unknown, the user enters a special `=QUESTION()` formula. This triggers an assisted process that helps users to pose their questions in *Stack Overflow* resorting to examples from the current spreadsheet to illustrate the desired output, all without leaving *Google Sheets*. Through this work, we pursue three main contributions:

- a scaffold design aimed at guiding users towards successful inline question posting (Sect. 3),

- a *Google Sheets* extension for inline question posting in *Stack Overflow* (Sect. 4),
- five use cases that provide insights about the benefits brought by inline question posting (Sect. 5).

We start by framing this work within the related bibliography.

## 2 Social Computing in Programming

While Personal Computing describes the behavior of isolated users, Social Computing highlights the social aspect when achieved through computers. Q&A platforms, microblogging or wikis can be framed within this term. The social aspect does not stop at collaboratively editing a wiki article or tweeting about a movie. Programming has also been the subject of socialization. A proliferation of approaches to provide this kind of capabilities *within* programming environments soon began. The vision is for users to seamlessly channel questions to Q&A platforms without leaving their working environment. By *channelling* is meant the interplay between *the Q&A platform* and *the working platform* during the question lifecycle. Questions are written from *the working platform*, and next, transparently posted into *the Q&A platform*. And the other way around: *the Q&A platform* is periodically pulled for answers that will eventually be available at *the working platform*. Differences among approaches stem from the working environment and the target social platform to tap into. Next, we cluster references based on two main working environments: *Eclipse* and spreadsheet programs (see Table 1).

**Table 1.** Approaches to social computing in programming.

Tool	Hosting platform	Social platform	Added value	Asker	Answerer
<i>Fishtail</i>	<i>Eclipse</i>	<i>Stack Overflow</i>	Discovering SO code examples	Programmer	Altruistic
<i>Seahawk</i>	<i>Eclipse</i>	<i>Stack Overflow</i>	Indexing into SO threads	Programmer	Altruistic
<i>Smartsheet</i>	Prop. spreadsheet framework	<i>Mechanical Turk</i>	Launching crowdsourcing tasks	Data analyst	Rewarded
<i>AskSheet</i>	Prop. spreadsheet framework	<i>Mechanical Turk</i>	Launching crowdsourcing tasks	Decision Maker	Rewarded
<b><i>QuestionSheet</i></b>	<i>Google App Sheet</i>	<i>Stack Overflow</i>	Launching SO threads	Home Users	Altruistic

*Eclipse* is an IDE for professional applications. The success of Q&A services soon led to devise mechanisms “for employees to appropriate status message Q&A as one possible source of stable peer support” [20]. For *Stack Overflow*, *Fishtail* is an *Eclipse* plugin that assists programmers in discovering code examples and documentation on the Web relevant to their current task [17]. Another plugin, *Seahawk*, turns Eclipse-hosted code snippets into hyperlinks which point to relevant *Stack Overflow* question threads [1]. *Seahawk* transparently channels all the communication between *Eclipse* and *Stack Overflow*.

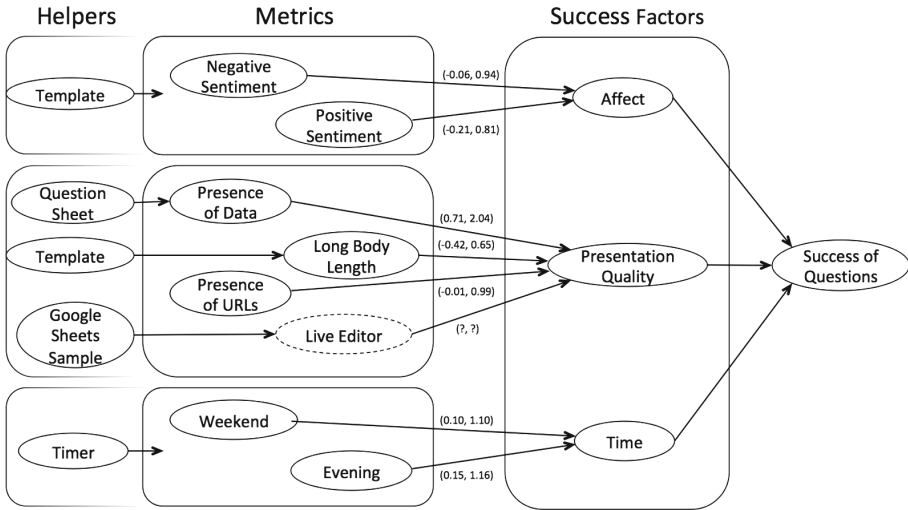
Spreadsheet frameworks also attempt to capitalize on social networks. *Smartsheet* [6] adds crowdsourcing capabilities to spreadsheets. Specifically, it permits to create tasks associated with spreadsheet cells so that cell values are obtained as a result of crowdsourcing tasks. *Smartsheet* transparently handles all the back-end process through *Amazon’s Mechanical Turk*. However, *Smartsheet* is not targeted to end users struggling with formulas but to professional programmers looking for data. In a similar vein, *AskSheet* [16] helps users to gather data for decision making in the form of spreadsheets. Decision making spreadsheets are data intensive, and include functions to aggregate data to help users to make a decision. To this end, *AskSheet* provides the *ASK()* function. *ASK()* takes as parameters a range of expected values (e.g. “1 to 10”), the item and attribute labels (e.g. “Galaxy S3”, “screen size”), the name of the task (“check basic specs”) and the target crowdsourcing platform (*Amazon’s Mechanical Turk*). The side-effect is the creation of a crowdsourcing task (a.k.a. HIT in the *Mechanical Turk* parlance). Main contribution rests on the optimization of the number of human tasks required to resolve all the cell dependencies in order to reduce the economic cost use as incentive to the human workers.

Our work is akin to these efforts insofar as inlining Q&A capabilities within the working environment. In so doing, questions are posted in context, facilitating focus and avoiding moving to the Q&A platform. However, previous approaches consider different target audiences. As for *askers*, previous works target professional programmers where coming up with good questions is not an issue but rather the smooth integration with the working environment. On the other hand, *answerers* might vary from rewarded ones *versus* altruistic ones. This introduces a remarkable difference: on money-rewarded systems (e.g. *Amazon’s Mechanical Turk*), the success is driven mainly by the amount of the incentive whereas in altruistic sites (e.g. *Stack Overflow*), success very much depends on how interesting the problem is and how well it is described.

Table 1 highlights how our work differs from previous approaches as for the combination (asker, answerer) being considered: (home users, altruistic respondents). If answerers are altruistic, then we should delve into what factors impact respondent engagement (Sect. 3). If askers are home users, then we should strive to provide some kind of scaffold that acts upon the engaging factors. If they are left on their own, studies confirm that newcomers tend to have lower chances of having their questions properly answered [4].

### 3 Scaffold Design for Successful Question Posting

This section gathers studies about how askers can increase the chances of eliciting a successful answer. Specifically, our approach pivots around the model for successful question posting on *Stack Overflow* presented in [5]. Calefato et al. focus on three main factors askers can act upon: affect (i.e. the positive or negative sentiment conveyed by text), presentation quality, and time (i.e. the moment at which the question is posted). Figure 1 depicts the main metrics introduced by Calefato et al. together with pairs (*coefficient estimate*, *odds ratio*) resulting from their experiment<sup>1</sup>. Next, we elaborate on means to assist users to increase these metrics, i.e. the scaffold.



**Fig. 1.** A model for successful question posting (adapted from [5]). Arrows are labeled by pairs (*coefficient estimate*, *odds ratio*) taken from [5]. Live editors are mentioned in this study but not investigated.

**Affect.** Successful questions adopt a neutral emotional style. This insight is aligned with previous investigations where expressing sentiment, either positive or negative, might decrease the chances of getting help [2].

Scaffold insight: Parse user text for sentiment analysis. Alternatively, a boilerplate template can be used.

<sup>1</sup> The sign of the coefficient estimate indicates the positive/negative association of the predictor with the success of a question. The odds ratio weighs the magnitude of this impact: the closer its value to 1, the smaller the impact of the parameter on the chance of success. A value lower than 1 corresponds to a negative association of the predictor with success (negative sign of the coefficient estimate), and the opposite for a value higher than 1.

**Presentation Quality.** Successful questions are “short, contain code snippets, and do not abuse with uppercase characters” [5]. The importance of attaching code snippets to questions was highlighted in a separate questionnaire where 95% of respondents strongly agreed with this view. An interesting twist was put forward by one of the respondents about the importance of using live editors, which would permit answerers to fiddle directly with the code samples provided by askers.

The literature backs the use of examples when posing questions [19]. Good examples can be a substitute for long explanations. This is particularly important for home users who might lack the skills to describe their needs in abstract terms, and hence, examples might be the easiest way to get the idea through. Good-practice manuals exist about how to write good questions [8, 10, 18]. Nasehi et al. advice for questions to have enough details (but not too many), enough depth (without drifting from the core subject), examples (if applicable) as well as including the avenues already investigated by the asker [15]. This effort pays off in terms of getting more high-quality answers, and hence, reducing the risks of obtaining inappropriate answers, or even no answer at all [13, 22].

Scaffold insight: Provide means for examples to be easily extracted from the working environment.

**Time Slice.** Experiments found that questions posted during the weekend are more likely to be answered than questions posted during the week [3]. In addition, the most successful time slices correspond to 3:00–6:00 PM of West Coast US time, that is, when most experts were available.

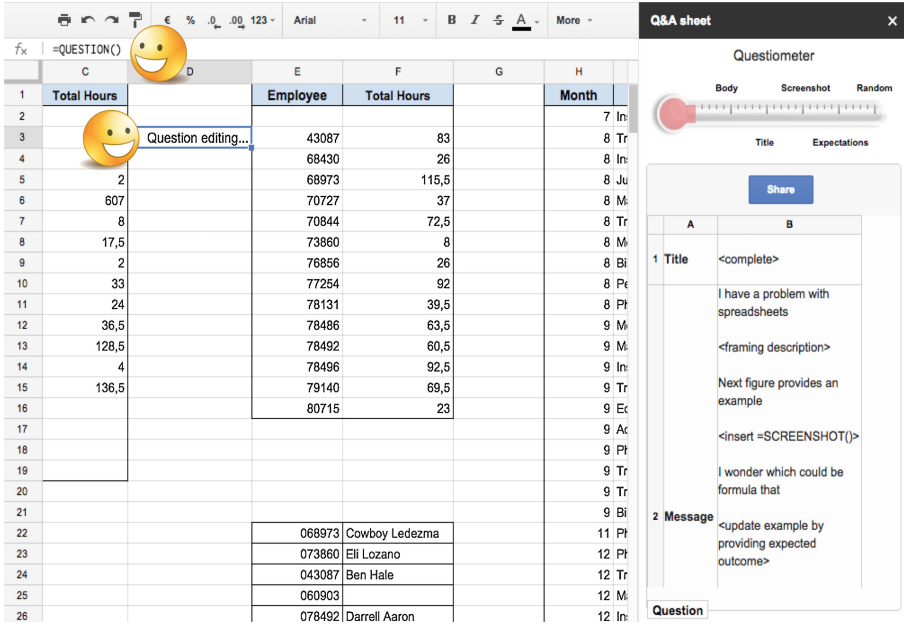
Scaffold insight: Provide a timer that decouples question specification from question posting. Users with a different time zone can work out their questions at the most appropriate time, and let the timer post them at the USA working hours or leave it till the weekend.

## 4 A scaffold for inline question posting for GoogleSheet

This Section tackles inline question posting using *Google Sheets* as the working environment, and *Stack Overflow (SO)* as the Q&A platform. The outcome is *QuestionSheet*, an extension for *Google Sheets* available for download at <https://goo.gl/RMD3p3>. Readers are encouraged to watch this video to see *QuestionSheet* at work: <https://goo.gl/wSy767>.

Functional requirements are those that derive from handling the question lifecycle. This includes: question description (i.e. title, message, answers, votes, respondent data, etc); question sharing (i.e. seamless propagation of the query from the working environment to the Q&A platform); answer awaiting (i.e. once the question is posted, mechanisms should be in place for the working environment to monitor the Q&A platform); answer resolution (i.e. once answers show up, mechanisms are required to handle answer resolution and integration into the working environment).

As for the non-functional requirements, we prevail “compatibility”, i.e. “the degree to which an innovation is perceived as being consistent with the existing



**Fig. 2.** *Question()* makes a *QuestionSheet* to be displayed as a side panel: the question and the working sheet are shown side by side. Smileys are overlaid to highlight points of interest.

values, needs, and past experiences of potential adopters” [21]. In our setting, we depart from a working platform to which the user is familiarized with (i.e. Google Sheet). Compatibility calls for the new functionality (i.e. inline question posting) to mimic as much as possible *the modus operandi* of Google Sheet. Targeting home users, consistency lowers the entry bar and eases user adoption [12].

The rest of this Section is structured along the stages of the question lifecycle. As the running example, consider the spreadsheet in Fig. 2. Employee data is scattered among two tables. One table holds the employee’s ID and hours worked. The table below keeps the ID and the employee’s name. The user wonders how the employee’s name could be replicated besides the ID column in the first table. Rather than re-typing this information, he looks for a function that achieves this.

### 4.1 Question Description

For the sake of compatibility, *QuestionSheet* is realized through two spreadsheet functions: *QUESTION()*, and *SCREENSHOT(Range)*.

**Initialization.** To use *QuestionSheet*, users start with an ordinary spreadsheet. Then, in cells where a formula is needed but unknown, the user enters a special *=QUESTION()* formula. Figure 2 shows the case for our running example (notice the expression *=QUESTION()* at the formula bar) From the perspective

of *Google Sheets*, *QUESTION()* is just another formula. Hence, *QUESTION()* can appear any place a formula is expected. The difference stems from the output. Traditional formulae output data. However, *QUESTION()* outputs a formula, and it has a side effect: popping up a *QuestionSheet* at the right-side. A *QuestionSheet* holds a cell for each of the question elements (e.g. Title, Message, etc). Function *QUESTION()* then obtains its arguments from this sheet<sup>2</sup>. As any other formula, *QUESTION()* output is re-evaluated as its companion *QuestionSheet* changes its content.

**Description.** At the onset, the *Message* cell contains a template that serves as a basic guideline for users. A hallmark of our approach is to enrich these messages with examples, i.e. providing input values and their expected outputs. For our running case, this entails adding the employee names by the IDs (red shaded in Fig. 3). The user is prompted to introduce the expected outputs that the sought formula would return. This not only facilitates understanding by potential respondents (and hence, the chances of getting a more accurate answer) but it also offers a way to validate the solution. Indeed, the blog of *Stack Overflow* itself advises to provide shared spreadsheet where potential respondents can more reliably reproduce the task scenario and validate their solutions [18]. Including expected outcomes in the working sheets might come in handy, but it also pollutes spreadsheets with spurious data. To avoid smudging the spreadsheet, the data introduced for illustration purposes is transient, i.e. they are kept from *QUESTION()* first enactment till the question is posted. Once the question is posted, no trace is left in the working sheet. Transient data is highlighted with a red shaded.

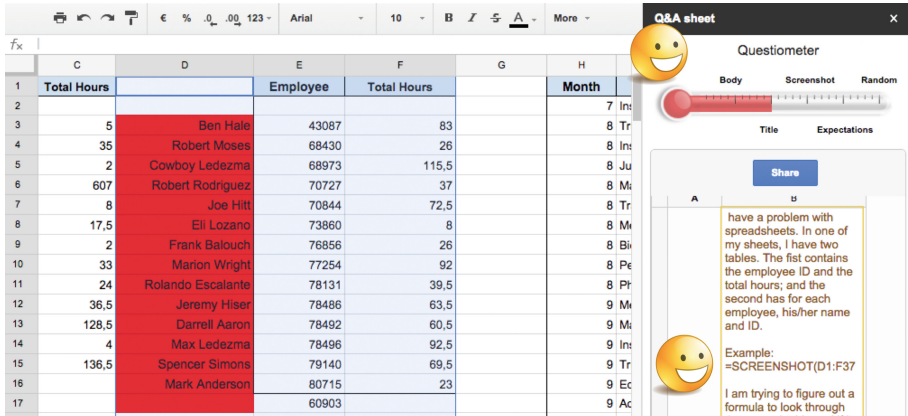


Fig. 3. Function *Screenshoot()* inlays sheet snippets into the message. (Color figure online)

<sup>2</sup> A spreadsheet can hold different *QUESTION()* formulae, each with its own *QuestionSheet*.




**Enriching Questions with Sample Data.** Sample data from the working sheet can be inlaid into the question’s message through *SCREENSHOT(Range)*. Figure 3 illustrates the use of this function. *SCREENSHOT()* obtains (1) a screenshot, and (2), a detached sample spreadsheet for the selected *Range*<sup>3</sup>. The screenshot helps message understanding while the detached spreadsheet allows respondents to easily try out their formulae. The actual thread can be found at <https://goo.gl/c9vbCF>.

## 4.2 Question Sharing

*QuestionSheets* can be posted immediately or handed to the timer. In this way, question specification is decoupled from question posting, letting users decide the most appropriate time slide for posting their questions. So far, the timer admits three values: *immediate*, *USA\_evening* and *USA\_weekend* based on the studies of [5]. When posted, *QuestionSheets* are transparently turned into their question-thread counterparts. Therefore, questions have a double representation: one in *SO* (as question threads) and one in *Google Sheets* (as *QuestionSheets*). These two representations need to be kept in sync, i.e. answers in *SO* are propagated to the *QuestionSheet* counterpart while changes in the *Q&A* spreadsheet might lead to re-edits of the question in *SO* (see later). Appropriate drivers map the *QuestionSheet* representation to the data structure expected by the Q&A platform at hand.

## 4.3 Answer Awaiting

Once the question is posted, drivers periodically polls *SO* for answers. This happens when either the spreadsheet is opened or in a polling frequency basis. By default, this polling frequency is set to 15’. In this way, answers in *SO* are propagated to the associated *QuestionSheet*. This in turn, causes *QUESTION()* to be re-evaluated. At this point, the user will notice how the *QUESTION()* display changes from “*Question posted...*” to “*Question answered (n) ...*” where “*n*” stands for the number of available answers. Figure 4 shows the case when two answers are available. It is not need for the *QuestionSheet* to be visible. Similar to the warnings about available app updates in mobile phones, the existence

	C	D	E	F
1	Total Hours		Employee	Total Hours
2				
3		<a href="#">Question answered (2)</a>	43087	83
4			68430	26
5			68973	115,5
		2		

**Fig. 4.** Adding answers makes function *QUESTION()* be re-evaluated showing the number of *SO* answers so far.

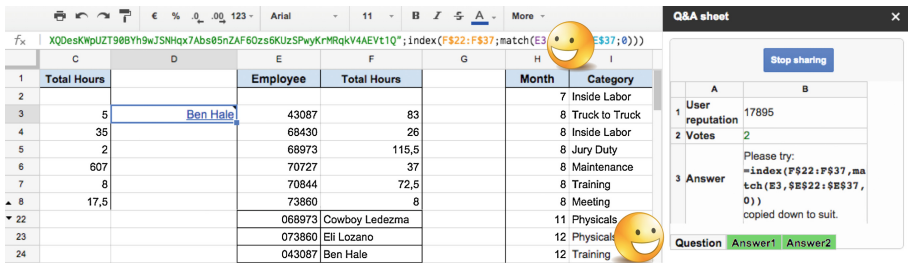
<sup>3</sup> This shared table is attached to the question’s message as a URL.

of answers do not force users to move to resolution right away. Based on the urgency, users might select the first answer that shows up or rather wait to see if additional answers come up.

#### 4.4 Answer Resolution

SO threads are mapped back into *QuestionSheets*. Information collected from SO includes the answer itself, the contributor’s reputation, and the number of votes (if any) (see Fig. 5). Users can move between answers by clicking on the tabs. This causes *QUESTION()* to be re-evaluated. Specifically, *QUESTION()* accesses the cell containing the answer, and attempts to extract the formula from the text. In most case, *QUESTION()* is successful since most contributors follow SO’s guidelines of using markdown for code. If so, *QUESTION()* returns the formula. This in turn, causes *Google Sheets* to evaluate the obtained formula, and return a value (e.g. *Ben Hale*). This value is compared with the expected outcome (should outcomes be provided at question time), shading the answer tab with green, yellow or red, depending on the number of expect outcomes the answer has matched. In this way, a single click on the question tab promptly permits users to see whether the select answer fits their expectations or not. No need to understand what the formula is about. Formulas are judged by their outputs. This highlights the importance of providing expected outcomes with proper coverage.

Users can wait till an answer produces the desired outputs. Once provided, they can make an answer final. To this effect, users click on the *stop-sharing* button (see top of Fig. 5). This ends the question lifecycle: the *QuestionSheet* is deleted together with the transient data. As for the SO counterpart, the query thread is finished along with SO good practices<sup>4</sup>. The only cue about a formula being obtained through SO, is by a comment left attached to the corresponding



**Fig. 5. Answer Resolution.** Answers are accessible via tabs (see Smiley at the bottom). Clicking on a tab has a three-fold effect: (1) the sought formula is obtained from the SO thread using pattern matching; this formula is displayed in the formula bar; and (3) the associated *QUESTION()* is re-evaluated to show what would be the formula output (e.g. *Ben Hale*). (Color figure online)

<sup>4</sup> <http://stackoverflow.com/help/someone-answers>.

cell. The comment holds the URL to the *SO* thread counterpart. This is the only trace left back about how the formula was obtained.

## 5 Evaluation

This section reports on a formative evaluation conducted through five case studies. For *QuestionSheet*, the experiment does not stop at posing the question but expand along the question lifecycle, including waiting for the *SO* community to answer. This potentially long lifespan is what sustains the use of use cases for formative evaluation.

**Table 2.** Task description. The *QuestionSheet*-generated *SO* thread is included for reference.

Case Study	Statement	Sample table	Sought Formula / Thread URL																																										
T1	Work out award nominee students, i.e. those whose marks are among the 25% top marks on his/her class	<table border="1"> <thead> <tr> <th></th> <th>A</th> <th>B</th> <th>C</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Student</td> <td>Mark</td> <td>Award Nominee</td> </tr> <tr> <td>2</td> <td>Mark Stenson</td> <td>9,2</td> <td>YES</td> </tr> <tr> <td>3</td> <td>Joshua McLarey</td> <td>5,9</td> <td>NO</td> </tr> <tr> <td>4</td> <td>Sandra Anderson</td> <td>6,4</td> <td>NO</td> </tr> <tr> <td>5</td> <td>John Tucker</td> <td>6,5</td> <td>NO</td> </tr> </tbody> </table>		A	B	C	1	Student	Mark	Award Nominee	2	Mark Stenson	9,2	YES	3	Joshua McLarey	5,9	NO	4	Sandra Anderson	6,4	NO	5	John Tucker	6,5	NO	$if(B2 \geq percentile(B\$2:B\$51;0,75); "YES"; "NO")$ <a href="https://goo.gl/KDNxms">https://goo.gl/KDNxms</a>																		
	A	B	C																																										
1	Student	Mark	Award Nominee																																										
2	Mark Stenson	9,2	YES																																										
3	Joshua McLarey	5,9	NO																																										
4	Sandra Anderson	6,4	NO																																										
5	John Tucker	6,5	NO																																										
T2	Count borrowed books by type (e.g. REF, AA) where type is embedded as part of the call-number ID	<table border="1"> <thead> <tr> <th></th> <th>A</th> <th>B</th> <th>C</th> <th>D</th> <th>E</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Name</td> <td>Call Number</td> <td></td> <td></td> <td></td> </tr> <tr> <td>2</td> <td>Patron A</td> <td>REF.01.2001</td> <td></td> <td></td> <td></td> </tr> <tr> <td>3</td> <td></td> <td>REF.04.2302</td> <td></td> <td></td> <td></td> </tr> <tr> <td>4</td> <td>Patron B</td> <td>REF.A2.260</td> <td>REF</td> <td></td> <td>3</td> </tr> <tr> <td>5</td> <td></td> <td>AA.26.7609</td> <td>FA</td> <td></td> <td>0</td> </tr> <tr> <td>6</td> <td></td> <td></td> <td>AA</td> <td></td> <td>1</td> </tr> </tbody> </table>		A	B	C	D	E	1	Name	Call Number				2	Patron A	REF.01.2001				3		REF.04.2302				4	Patron B	REF.A2.260	REF		3	5		AA.26.7609	FA		0	6			AA		1	$ArrayFormula(query(\{A2:Avegextract(B2:B; "\^[A-Z]*"); "Select Col2, count(Col1) where Col2 <>" group by Col2 label count(Col1) ""))$ <a href="https://goo.gl/CHy9b3">https://goo.gl/CHy9b3</a>
	A	B	C	D	E																																								
1	Name	Call Number																																											
2	Patron A	REF.01.2001																																											
3		REF.04.2302																																											
4	Patron B	REF.A2.260	REF		3																																								
5		AA.26.7609	FA		0																																								
6			AA		1																																								
T3	Place employees' names by the Employee columns	<table border="1"> <thead> <tr> <th></th> <th>A</th> <th>B</th> <th>C</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>Employee</td> <td>Total Hours</td> </tr> <tr> <td>2</td> <td>Ben Hale</td> <td>43087</td> <td>83</td> </tr> <tr> <td>3</td> <td>Robert Moses</td> <td>68430</td> <td>26</td> </tr> <tr> <td>4</td> <td>Joe Hitt</td> <td>70844</td> <td>72.5</td> </tr> <tr> <td>5</td> <td></td> <td>043087</td> <td>Ben Hale</td> </tr> <tr> <td>6</td> <td></td> <td>070844</td> <td>Joe Hitt</td> </tr> <tr> <td>7</td> <td></td> <td>068430</td> <td>Robert Moses</td> </tr> </tbody> </table>		A	B	C	1		Employee	Total Hours	2	Ben Hale	43087	83	3	Robert Moses	68430	26	4	Joe Hitt	70844	72.5	5		043087	Ben Hale	6		070844	Joe Hitt	7		068430	Robert Moses	$index(F\$22:F\$37, match(E3, \$E\$22:\$E\$37, 0))$ <a href="https://goo.gl/c9vbCF">https://goo.gl/c9vbCF</a>										
	A	B	C																																										
1		Employee	Total Hours																																										
2	Ben Hale	43087	83																																										
3	Robert Moses	68430	26																																										
4	Joe Hitt	70844	72.5																																										
5		043087	Ben Hale																																										
6		070844	Joe Hitt																																										
7		068430	Robert Moses																																										
T4	Calculate working time subtracting 30' lunch break, if applicable	<table border="1"> <thead> <tr> <th></th> <th>A</th> <th>B</th> <th>C</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Start time</td> <td>End time</td> <td>Total time</td> </tr> <tr> <td>2</td> <td>7:00:00</td> <td>13:30:00</td> <td>6:00</td> </tr> <tr> <td>3</td> <td>7:00:00</td> <td>10:00:00</td> <td>3:00</td> </tr> </tbody> </table>		A	B	C	1	Start time	End time	Total time	2	7:00:00	13:30:00	6:00	3	7:00:00	10:00:00	3:00	$mod(B2-A2; 1) - if(mod(B2-A2; 1) > time(4; 0; 0); time(0; 30; 0); 0)$ <a href="https://goo.gl/heqhr3">https://goo.gl/heqhr3</a>																										
	A	B	C																																										
1	Start time	End time	Total time																																										
2	7:00:00	13:30:00	6:00																																										
3	7:00:00	10:00:00	3:00																																										
T5	Calculate average calories for "moderate" food, with the formula being resilient to additional intakes being inlaid	<table border="1"> <thead> <tr> <th></th> <th>A</th> <th>B</th> <th>C</th> <th>D</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Intake</td> <td>Calories</td> <td></td> <td></td> </tr> <tr> <td>2</td> <td>negative</td> <td>2304</td> <td>Avg Net Carbs</td> <td>32</td> </tr> <tr> <td>3</td> <td>moderate</td> <td>890</td> <td></td> <td></td> </tr> <tr> <td>4</td> <td>small</td> <td>1617</td> <td>Avg Calories</td> <td>1360</td> </tr> <tr> <td>5</td> <td>small</td> <td>1370</td> <td></td> <td></td> </tr> <tr> <td>6</td> <td>negative</td> <td>1065</td> <td>Avg Moderate</td> <td>902</td> </tr> <tr> <td>7</td> <td>moderate</td> <td>914</td> <td></td> <td></td> </tr> </tbody> </table>		A	B	C	D	1	Intake	Calories			2	negative	2304	Avg Net Carbs	32	3	moderate	890			4	small	1617	Avg Calories	1360	5	small	1370			6	negative	1065	Avg Moderate	902	7	moderate	914			$averageif(A2:A; "moderate"; B2:B)$ <a href="https://goo.gl/WuR6me">https://goo.gl/WuR6me</a>		
	A	B	C	D																																									
1	Intake	Calories																																											
2	negative	2304	Avg Net Carbs	32																																									
3	moderate	890																																											
4	small	1617	Avg Calories	1360																																									
5	small	1370																																											
6	negative	1065	Avg Moderate	902																																									
7	moderate	914																																											

### 5.1 The Experiment

**Subjects.** Participants were recruited locally. For participants to qualify as "home users", they should have at least one year exposure to *Google Sheets* but

not having created more than six sheets in this period. Five students qualified. Participants were given a brief introduction to *QuestionSheet* where a running example was developed.

**Case Studies.** Scenarios were carefully selected so that questions should not be neither too trivial nor too complicated. For the validity of the experiment, it is most important to find the right balance. Too easy cases will not justify the effort of *QuestionSheet* as most users will know the answers without resorting to SO. In addition, easy questions will get prompt answers no matter whether the description of the query is enhanced with examples or not (one of the aspects to be tested out). On the other hand, too complicated cases will be also difficult for home users to stumble upon in their daily activities. Table 2 displays the selected cases inspired by real doubts risen in forums. The table also includes the sought formula for readers to ponder the scenarios' complexity. In addition, a link to the *QuestionSheet*-generated question in Stack Overflow is also included.

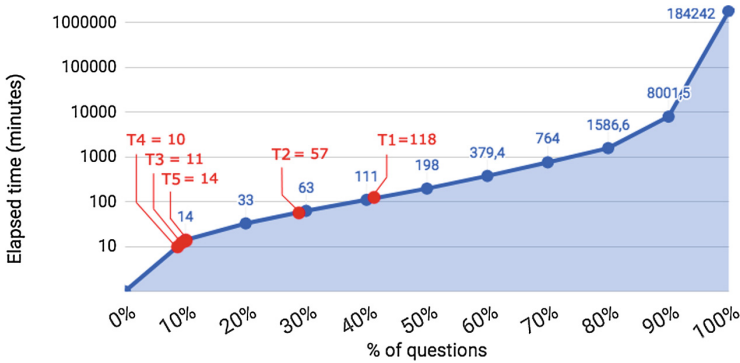
**Table 3.** Constructs of diffusion theory adapted for *QuestionSheet*.

Construct	Definition: The degree to which <i>QuestionSheet</i> innovation is perceived as
Relative advantage	...being better than typing the question directly on StackOverflow
Compatibility	...consistent with the existing values, needs, and past experiences of GoogleSheet users
Complexity	...being difficult to use

**Methodology.** Each subject addresses one of previous scenarios. To prevent scenario understanding from polluting the performance measures on the usage of *QuestionSheet*, the task was split into parts. First, we made sure subjects understand the scenario, i.e. they were asked to complete the sample spreadsheet with the values the expected formula should return (red shaded in Table 2). This helped to ensure scenarios were properly understood. Next, subjects were asked to post their questions using *QuestionSheet*. Once the testing session was over, participants were asked to fill in a questionnaire that rates different aspects of *QuestionSheet* through Likert scales (see Fig. 7). In addition, open comments were also welcome. The questionnaire builds upon a reduction of Roger's model of Diffusion of Innovations that includes only those constructs consistently related to technology adoption behavior: relative advantage, complexity and compatibility [21]. Table 3 elaborates these constructs for the *QuestionSheet* case.

## 5.2 Results

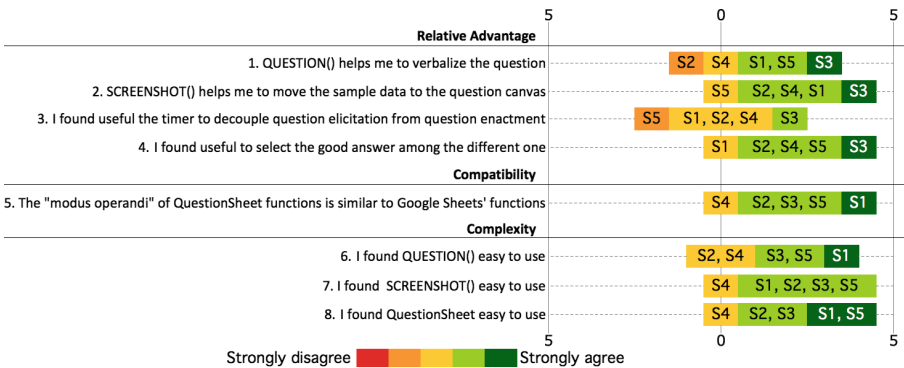
**Elapsed Time.** That is, the time it took for the SO community to answer the *QuestionSheet*-generated questions. We frame this outcome w.r.t. the distribution of time-to-answer in SO, more specifically, we obtained the elapsed-time first-answer distribution for “*google-spreadsheet*”-tagged questions in SO



**Fig. 6.** Elapsed-time first-answer distribution for “google-spreadsheet”-tagged questions at *Stack Overflow* as for September, 2017. Questions without answer were removed.

(see Fig. 6)<sup>5</sup>. We use this figure to frame our results: T3, T4 and T5 are among the top 10%, T2 in the 30% and T1 in the 50%. We believe these results to be rather good. More to the point, if we consider that questions were posted by SO newcomers with no reputation, where reputation is being indicated as a main correlator of response speed [5]. This anecdotal evidence supports the findings of Calefato et al. as for the use of examples as main answer attractors.

**Questionnaire.** In general, users perceive *QuestionSheet* as beneficial w.r.t. directly typing the question in SO (questions 1–4 in Fig. 7). Important enough, users ranked high *QuestionSheet* compatibility with *Google Sheets modus operandi* (question 5). This is a main incentive to user adoption.



**Fig. 7.** Diverging stacked bar chart for the satisfaction questionnaire using Likert scales.

<sup>5</sup> Studies indicate that how the question is being tagged is the best indicator to predict when an answer is to be expected [7]. Distribution on Fig. 6 was obtained by enacting the query at [goo.gl/RgGBE7](http://goo.gl/RgGBE7).

**Threats to Validity.** A larger number of participants are required to draw any conclusive remarks. As for external validity, *QuestionSheet* could post questions to Q&A platforms other than *StackOverflow* as long as appropriate API-based drivers were provided. On the other hand, our experience can be of interest for frameworks that target home users who can resort to examples to make themselves understood in Q&A forums.

### 5.3 Lessons Learned

This subsection discusses lessons learned along the model introduced by [5].

**Affect.** Calefato et al. highlight the importance of a neutral emotional style to promote answers. In a similar vein, we realized subjects were not so aware of the impact of posting questions repeatedly. Specifically, two users posted their questions (or small variations) more than once. *QuestionSheet* might turn both *StackOverflow* too transparent and question posting too easy, with the resulting risk of “question spam”. This makes true the famous quote: “*Everything Should Be Made as Simple as Possible, But Not Simpler*”. Users ignored the importance of reputation when interacting with social networks, and the eventuality of being banned, if a proper etiquette is not observed. Hence, *QuestionSheet* needs to be revised as for restricting the number of questions per sheet, or preventing questions which have already been posted at *StackOverflow*.

**Presentation Quality.** Calefato et al. underscore the importance of attaching code snippets and examples to questions. This remark was even more important in our setting: non-native English speaker. Two subjects noticed that without examples they would have not been able to describe their questions in a narrative way using English. One subject indicated that, though she knew *Stack Overflow*, she was afraid of posing question due to her limited use of English. Here, templates and examples lower the entry bar for home users.

**Time Slice.** Users were pleasantly surprised for the promptness of the answers. As a subject puts it “it is almost like Google!”. This promptness might be well due to the presentation quality of the questions (i.e. neutral language, example-based). But it might also be influenced by the medium complexity of the questions. Being home users, the expectations are for questions to be of medium complexity hence, easily answerable by more proficiency answerers. At this respect, delaying question posting till the evenings or weekends might make sense when there might be a need to attract a large number of answerers. However, medium-complexity questions might well be equally addressed by a smaller population without waiting till the weekend. If this were the case, it would certainly challenge the need of the built-in timer in *QuestionSheet*.

## 6 Conclusions

This work tackles how to spread the benefits of Q&A platforms to home users. To this end, we look into assisted inline question posting where users post their

questions without leaving their working environment. *QuestionSheet* tests this out for *Google Sheets*. Offering Q&A facilities from within *Google Sheets* not only avoids platform switches and facilitates focus, but also accounts for in-place example construction to be used for question clarification, which in turn helps to engage answerers. Informative evaluation is being conducted through five use cases. Though conclusive statements cannot yet be drawn, first insights seem to suggest that in-place scaffolds might help to combat the digital divide between home users and more technical savvy ones as for enjoying the benefits of Q&A platforms. Yet, the benefits of educating users to formulate questions goes beyond the askers themselves. Further investigation should look at how “good questions” enhance effective knowledge-sharing behavior that will eventually lead to the creation of long-lasting value pieces of knowledge in Q&A platforms. Along Israelmore Ayivor quote “*To get answers, ask questions; but to get good answers, ask good questions*”, we could conclude that for Q&A platforms to become a repository of good answers, first you need good questions.

**Acknowledgements.** This work is co-supported by the Spanish Ministry of Education, and the European Social Fund under contract TIN2014-58131-R. Contell enjoys a doctoral grant from the University of the Basque Country.

## References

1. Bacchelli, A., Ponzanelli, L., Lanza, M.: Harnessing Stack Overflow for the IDE. In: Proceedings of the 3rd International Workshop on Recommendation Systems for Software Engineering, pp. 26–30 (2012)
2. Bazelli, B., Hindle, A., Stroulia, E.: On the personality traits of StackOverflow users. In: Proceedings of the 29th IEEE International Conference on Software Maintenance (ICSM), pp. 460–463 (2013)
3. Bosu, A., Corley, C.S., Heaton, D., Chatterji, D., Carver, J.C., Kraft, N.A.: Building reputation in StackOverflow: an empirical investigation. In: Proceedings of the 10th Working Conference on Mining Software Repositories (MSR), pp. 89–92. IEEE Press (2013)
4. Calefato, F., Lanubile, F., Marasciulo, M.C., Novielli, N.: Mining successful answers in Stack Overflow. In: Proceedings of the 12th IEEE/ACM Working Conference on Mining Software Repositories (MSR), pp. 430–433 (2015)
5. Calefato, F., Lanubile, F., Novielli, N.: How to ask for technical help? Evidence-based guidelines for writing questions on stack overflow. *Inf. Soft. Technol.* **94**(C), 186–207 (2018)
6. Frei, B.: Paid crowdsourcing: current state & progress toward mainstream business use, report (2009). <https://www.smartsheet.com>
7. Goderie, J., Georgsson, B.M., van Graafeiland, B., Bacchelli, A.: Eta: estimated time of answer predicting response time in Stack Overflow. In: Proceedings of the 12th IEEE/ACM Working Conference on Mining Software Repositories (MSR), pp. 414–417 (2015)
8. Harper, F.M., Raban, D.R., Rafaeli, S., Konstan, J.A.: Predictors of answer quality in online Q&A sites. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 865–874 (2008)

9. Ko, A.J., Myers, B.A.: Human factors affecting dependability in end-user programming. In: Proceedings of the 1st Workshop on End-User Software Engineering (2005)
10. Korvela, H.: Plz urgent help needed!1!! - aspects of on-line knowledge sharing in end-user development support. In: Proceedings of the 19th Americas Conference on Information Systems (AMCIS) (2013)
11. Korvela, H., Back, B.: The impact of skills and demographics on end-user developers' use of support. In: Proceedings of the 18th Americas Conference on Information Systems (AMCIS) (2012)
12. Korvela, H., Packalén, K.: On-line support - a virtual treasure trove for end-user developers in small organisations? In: Proceedings of the 15th Americas Conference on Information Systems (AMCIS) (2009)
13. Mamykina, L., Manoim, B., Mittal, M., Hripcsak, G., Hartmann, B.: Design lessons from the fastest Q&A site in the west. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 2857–2866 (2011)
14. Moffitt, K.: Cases on the human side of information technology. In: End-User Computing at BRECI: The Ordeals of a One-Person IS Department, pp. 330–342. IGI Global (2006)
15. Nasehi, S.M., Sillito, J., Maurer, F., Burns, C.: What makes a good code example?: A study of programming Q&A in StackOverflow. In: Proceedings of the 28th IEEE International Conference on Software Maintenance (ICSM), pp. 25–34 (2012)
16. Quinn, A.J., Bederson, B.B.: AskSheet: efficient human computation for decision making with spreadsheets. In: Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW), pp. 1456–1466 (2014)
17. Sawadsky, N., Murphy, G.C.: Fishtail: from task context to source code examples. In: Proceedings of the 1st Workshop on Developing Tools and Plug-ins (TOPI), pp. 48–51 (2011)
18. StackOverflow: How do I ask a good question? <http://stackoverflow.com/help/how-to-ask>. Accessed 23 Mar 2018
19. Tagg, J.: Discovering ideas handbook: use specifics and examples (2003). <https://www2.palomar.edu/users/jtagg/handbook/specific.htm>. Accessed 23 Mar 2018
20. Thom-Santelli, J., Yuen, S., Matthews, T., Daly, E.M., Millen, D.R.: What are you working on? Status message Q&A in an enterprise SNS. In: Proceedings of the 12th European Conference on Computer Supported Cooperative Work (ECSCW), pp. 313–332 (2011)
21. Tornatzky, L.G., Klein, K.J.: Innovation characteristics and innovation adoption-implementation: a meta-analysis of findings. *IEEE Trans. Eng. Manage. EM* **29**(1), 28–45 (1982)
22. Treude, C., Barzilay, O., Storey, M.D.: How do programmers ask and answer questions on the web? In: Proceedings of the 33rd International Conference on Software Engineering (ICSE), pp. 804–807 (2011)
23. Wenger, E.: Communities of Practice and Social Learning Systems. The Systems Thinker (1998)
24. Xiao, L., Farooq, U.: Supporting end user development in community computing: requirements, opportunities, and challenges. *J. Commun. Inf.* **9**, 1 (2013)
25. Yang, J., Hauff, C., Bozzon, A., Houben, G.: Asking the right question in collaborative Q&A systems. In: Proceedings of the 25th ACM Conference on Hypertext and Social Media, pp. 179–189 (2014)