Receptive Users and Time-Insensitive Recommendations Improve Tool Discovery

Chris Brown
Department of Computer Science
North Carolina State University
Raleigh, NC
dcbrow10@ncsu.edu

ABSTRACT

Software contains many features and tools that provide a variety of functionality to help users save time and effort in completing tasks. However, users often have limited knowledge of these tools and features in applications. Peer interactions are an effective way for users to discover and integrate new tools into their normal software usage. This study investigates improving tool discoverability by creating a model to define a tool recommendation during peer interactions, analyzing peer characteristics and types of tools recommended during observed peer interactions between participants in a user study, and providing implications for designing and improving automated tool recommendation systems.

Keywords

Peer Interaction; Tool Recommendation; Tool Discovery

1. INTRODUCTION

Software applications contain numerous tools designed to make usage easier for users. We define a *tool* as any software command or feature that accomplishes a task. Software tools such as Flash Fill in Excel and Open Resource in Eclipse are designed to help users complete tasks efficiently, provide new functionality, and improve user experience. Many tools and features in software are rarely utilized and discovered by users [11]. This lack of awareness can lead to wasted time, which costs companies billions of dollars each year [24].

Software features are underused because tools have many barriers to entry that prevent users from adopting them [27]. These barriers hinder reliability, usability, interoperability, and discoverability. The discoverability barrier refers to when users are not aware of a tool and do not know how to find it or when to use it. Improving tool discoverability is important to help users learn and adopt useful tools as programs become increasingly "bloated" [26] with functionality and features.

There are several examples of attempts to improve tool discoverability that have been unsuccessful. Microsoft Clippy was a user interface designed to recommend helpful tools in Microsoft Office products. Theories suggest that Clippy was ineffective because users found it interruptive [10], impolite [40], and annoying [17]. Another technique software developers have implemented to try to enhance tool discoverability is "Help" menus. These menus appear in many applications and allow users to search for features and discover tools to help accomplish tasks. However, help menus are passive help systems which are ineffective and inefficient for users [8]. Menus require users to recognize their existence, navigate to items to seek help, and search through documentation or manuals to discover new functionality.

Previous research by Murphy-Hill and colleagues describes seven modes of discovery for learning new tools: Peer Observation, Peer Recommendation, Tool Encounter, Tutorials, Written Description, Twitter or RSS Feed, and Discussion Threads [30]. The first two discovery modes are examples of peer interaction, or the process of users discovering tools from their peers while completing normal work activities [29]. Murphy-Hill concluded that peer interactions were the most effective modes for discovering tools, but they are also less frequent modes of discovery. Our study focuses on analyzing what makes peer interactions effective modes of tool discovery.

We observed participants to research what makes peer interactions effective and seek to improve tool discoverability by studying the following research questions (RQs):

RQ1 What characteristics of peers make recommendations effective?

RQ2 What types of tools are most effectively recommended during peer interactions?

We analyzed personal characteristics of participants to determine what qualities of peers make peer interactions an effective method for tool discovery. Fischer suggests that recommendation systems should guide and advise users "similar to a knowledgeable colleague or assistant" [8]. Studying personal characteristics of peer interactions can provide insight for designing and improving recommendation systems for users. We also examined the tools recommended between participants in our study to find if certain types of features are more successfully discovered during peer interactions than others.

DOI: 10.1145/1235

2. RELATED WORK

Our study builds on prior research examining tool discovery and peer interactions.

2.1 Tool Discovery

Several researchers have examined reasons why users do not discover and adopt tools. McGrenere and Moore observed participants using Microsoft Word in Office 97 to examine their familiarity with features in the word processor and found that users are only familiar with about 50% of the functionality in Word [26]. Johnson and colleagues examined why software engineers do not use static analysis tools to improve code quality [15]. Similarly, Xiao and colleagues examined why software developers do not adopt security tools, even though they are helpful in finding security vulnerabilities and developers believe software security is important [42]. These studies focus on why users do not adopt useful software tools, and our research aims to help resolve this problem by analyzing what makes peer interactions effective modes of tool discovery.

Many researchers have proposed various techniques improve tool discovery in software. Findlater proposed customizing development environments to only contain features relevant to users' role in a company by analyzing the advantages and disadvantages of a role-based approach [7]. Murphy-Hill and colleagues examined different categories of tool recommendations in complex software and propose new algorithms based on comparisons to user's and community usage history to improve discovery of tools for software developers from colleagues [28]. Our research also intends to submit new techniques for tool discovery by studying peer characteristics and types of tools during peer interactions.

Previous research has presented various recommendation systems and tools to improve tool discoverability. One example is OWL, a tool implemented by Linton for organization-wide learning that observes commands used by colleagues and notifies users of popular tools in Microsoft Word based on knowledge from peers [22, 21]. Furthermore, Maltzahn presented a prototype help system called ToolBox to collect information from networked workstations and recommend Unix commands [25]. Lastly, Viriyakattiyaporn and Murphy introduced an active help system Spyglass, which improves awareness and use of navigation tools by observing a user actions and suggesting commands to help complete tasks more efficiently [39]. Contrary to these projects, our research aims to provide implications for creating and designing effective recommendation systems based on peer interactions.

2.2 Peer Interactions

Many researchers have shown that over-the-shoulder learning is effective in increasing knowledge. Damon evaluated the effectiveness of peer-based learning and education and discovered that collaboration improved learning and knowledge [5]. Cockburn and Williams studied pair programming, where two programmers develop software together on the same computer, and found that working with a peer saved time and money in addition to improving employee satisfaction, design quality, peer reviews of code, problem solving, learning, team building and communication, and project management of an organization [4]. Peer interaction is a form of over-the-shoulder learning, and our research builds on prior work and seeks to find what makes peer interaction effective for learning.

There is limited research on peer interactions, but previous research indicates that they are effective for tool discovery. Murphy-Hill presents *Peer Observation* and *Peer Recommendation* as types of peer interaction [30]. Peer Observation occurs when a peer observes a user utilizing an unfamiliar tool and Peer Recommendation occurs when a peer suggests a tool to another user. Murphy-Hill used interviews with computer programmers to determine that peers were the most effective way to learn tools but do not occur frequently in the workplace. We aim to observe Peer Observations and Peer Recommendations to determine what makes peer interactions an effective mode of tool discovery.

Similar to our work, several projects have examined the effectiveness of peer interactions. Murphy-Hill and colleagues extended their previous research to conduct interviews and diary studies with participants to understand what makes peer interactions effective compared to other modes of discovery [29]. Additionally, Maalej and colleagues examined the effectiveness of independent peer observations and peer debriefings, or a discussion between colleagues at the end of the day, in software developers' program comprehension [23]. We also aim to study what makes peer interactions effective by analyzing personal characteristics and types of tools during peer interactions.

Researchers have also proposed integrating qualities of peers into recommendation systems. Murphy-Hill proposed continuous social screencasting systems to simulate peer interactions for collocated colleagues through recorded videos of co-workers [27]. Snipes surveyed professional software engineers and proposed the gamification of the adoption of tools and practices to improve tool discovery by presenting results to peers [38]. Kalliamvakou and colleagues analyzed Github and noted how it allowed for open source-style collaboration between colleagues and can lead to peer interactions and tool adoption [16]. Our research also suggests incorporating features of peer interaction into recommendation systems, but it differs by focusing on analyzing peer characteristics and types of tools.

3. TOOL RECOMMENDATION MODEL

The goal of this study is to determine if characteristics of peers and types of tools impact the effectiveness of recommendations. This section presents a model we created to characterize tool recommendations during peer interactions. Our model was created based on prior work in peer recommendations and cognitive modeling in human-computer interaction. We used the following model to characterize instances of tool discovery during peer interactions:

Task Analysis → Task Execution → Dialogue → Reaction

The purpose of this model is to distinguish relevant actions during peer interactions. It helped focus our efforts for collecting data while analyzing recordings from pairs in our study. The remainder of this section describes each stage of our model and provides a detailed example. We borrowed terms from pair programming to describe the roles for each partner and to help identify recommendation patterns in peer interactions for our model, referring to the person operating the computer at the keyboard and mouse as the "driver" and the peer working with the driver as the "navigator" [4].

3.1 Task Analysis

The first stage of the model consists of users analyzing the task and defining their strategy to reach the goal. This stage utilizes Kieras' research on the GOMS model in *The Handbook of Task Analysis for Human-Computer Interaction* [1]. Task analysis consists of the driver and navigator mentally dividing the current task into Goals, Operators, Methods, and Selection rules. Methods are a series of operators that are used to accomplish specified goals, and selection rules are applied when more than one method exists. Many software has multiple ways for users to complete a task and reach a goal using different tools and features of a program.

For example, two peers Adam and Zach are using Excel to inspect test scores. Their goal is to calculate the average grade for 15 students in column A of a spreadsheet. Adam and Zach both understand the goal, but they define different operations to accomplish it during Task Analysis. Adam is a novice user and forms the operators in his method as:

1. Select an empty cell
2. Navigate to the "Formulas" menu at the top of the screen
3. Select "More Functions" in the menu
4. Expand the "Statistical" sub-menu
5. Select "AVERAGE" under statistical functions
6. Enter A1, A2, A3,..., A15 as individual number parameters in the Function Arguments pop-up box
7. Click "OK"

Meanwhile, Zach is an expert Excel user and his operators consist of:

1. Select an empty cell
2. Type "=AVERAGE(A1:A15)" in the cell
3. Press Enter

This illustrates how two users using the same software can apply different selection rules and choose contrasting methods for completing a task.

3.2 Task Execution

In the second stage, the driver applies their selection rule and begins executing their method defined during task analysis. During peer interactions, this leads to the navigator noticing a mismatch between their expectations and the driver's actions. The following sections provide examples of how Task Execution differs in Peer Recommendation and Peer Observation.

Peer Recommendation

During Peer Recommendation, the navigator observes that the driver is completing a task in an inefficient way. For instance, Adam is driving when Zach observes Adam navigating to the "Formulas" menu to find the AVERAGE function. Zach knows from previous experience that it is much more efficient to type the function in a cell using =AVERAGE(A1:A15) to calculate the mean.

Peer Observation

The driver begins executing their methods and their actions are unfamiliar to the navigator in Peer Observation. In the case where the roles are switched, Adam observes Zach driving and typing text into a cell. Adam notices the average was calculated by Zach without using the Excel menu is not sure of the methods that took place to reach the goal.

3.3 Dialogue

The next stage consists of a discussion between peers after the discrepancy in the Task Analysis between the driver and navigator. The Dialogue is an optional stage in actual tool discovery, but was crucial to our study and used in our analysis to determine when recommendations occurred between participants. We analyzed the dialogue between participants in our study to determine if peers exhibited specific characteristics during interactions. Explanations and examples of the dialogue between peers appear in the following sections, and we build on Murphy-Hill's definitions of Peer Recommendation and Peer Observation to include expected and unexpected categories. This was necessary to further define the dialogue between participants during peer interactions based on which peer initiates the conversation and which peer makes the recommendation.

Peer Recommendation Unexpected

Peer Recommendation Unexpected is the traditional definition of Peer Recommendation when the navigator observes the driver working and proposes a new tool or method [29]. In this case, the driver does not expect a recommendation from the navigator. Adam and Zach demonstrate then if Zach observes Adam driving and navigating to the Excel menu to find the function. Zach interrupts and says "Just typing the AVERAGE function would be a lot faster". This leads to a discussion between the peers and Adam discovering a new feature using =AVERAGE() to find the mean of values in a column.

Peer Recommendation Expected

For Peer Recommendation Expected, the driver explicitly asks the navigator for help and expects a recommendation. This categorization was necessary to account for instances where the driver initiates the dialogue, but the navigator still makes the recommendation. An example of this occurs when Adam does not remember where to find the AVERAGE function and asks, "Do you know where to find the average?". Zach explains to Adam that it's possible to compute the average by typing =AVERAGE(A1:A15) into a cell without using the menu.

Peer Observation Unexpected

The driver initiates the dialogue and causes the navigator to discover a new tool during Peer Observation Unexpected. We added this type of Peer Observation to capture interactions where the navigator unexpectedly learns about a new tool after the driver starts the conversation by explaining their actions or asking if a certain tool should be used. An example of this arises as Zach drives and mentions, "I'll go ahead and type it in here to see what happens". Adam confused with his limited knowledge of Excel functions and asks, "Type what?". This leads to a discussion on discovering new functionality by typing formulas in Excel.

Peer Observation Expected

Peer Observation Expected is the original Peer Observation where the navigator inquires about the driver's actions after observing an unfamiliar method. In this case, the navigator expects a recommendation from the driver after asking about their actions. For example, when Adam observes Zach calculate the mean in an unusual way by typing text into a cell, he asks "What did you do?". Then Zach explains how

Excel allows users to type the AVERAGE command into a cell to find the mean rather than always navigating to the menu.

3.4 Reaction

The final stage of our model consists of the reaction from the dialogue between peers. The reaction is also optional in terms of tool discovery, but was required for observing tool recommendations during peer interactions in our study. There are many possible reactions to a tool suggestion between peers, and we characterize them in two categories: *effective* or *ineffective*. An effective peer recommendation is one where the recommendee adopts the discovered tool or feature, while an ineffective peer recommendation is one where the recommendee ignores the dialogue between peers about the new tool.

In our peer interaction example, both types of reaction would be the same for Peer Recommendation or Peer Observation. An effective suggestion would lead to Adam incorporating the =AVERAGE command for future tasks when he needs to find the mean. If the recommendation was ineffective, then Adam will continue navigating to the Excel menu each time he needs to calculate the average and ignore typing the the function in a cell.

4. METHODOLOGY

The tool recommendation model in Section 3 helped us collect data to analyze for our study. We designed our methodology to discover if personal characteristics and types of tools influence tool discovery in peer interactions.

4.1 Study Design

4.1.1 Participants

We had a total 26 participants in our study form 13 pairs. We paired participants together based on schedule availability. The subjects arrived at the test location at the designated time, signed research consent forms, and began the study tasks. Our study was divided into two phases. The first phase was made up of 7 pairs of students and the second phase consisted of 6 pairs of data analysts from the Laboratory for Analytic Sciences[1] (LAS) to complete the second phase. The students provided a variety backgrounds and knowledge about different software and we recruited subjects from LAS because they have professional experience using tools to gather and analyze data as intelligence analysts.

4.1.2 Study Phases

The participants in the first phase of our study were graduate and undergraduate students at North Carolina State University studying Computer Science, Biochemistry, and Industrial Engineering. The first phase of the study required participants to complete six preliminary tasks and a final task. After the study concluded, the students were emailed a survey to fill out regarding the Peer Observations and Peer Recommendations that occurred during the study.

The second phase involved professional data analysts from LAS. We made general improvements to the study for the second phase by removing two preliminary study tasks and including a semi-structured interview focusing on one effective and one ineffective recommendation that occurred during the session and a survey to gather demographic information. These changes reduced the total number of tasks but allowed us to collect more data from participants concerning tool recommendations between peers for our study.

4.1.3 Tasks

The tasks involved analyzing data from the Titanic shipwreck from the Kaggle machine learning data science competition.[2] Each pair had to find relationships between different characteristics of individuals in the data, and then predict whether passengers survived. The study tasks for LAS and student participants can be found in Appendix A. The tasks were created to simulate partners working together on normal data analysis work and designed to elicit peer interactions based on research by Murphy-Hill [30]. The data for the tasks consisted of two separate comma separated values files, *train.csv* and *test.csv*.

The first tasks had participants examine data in *train.csv*, which contained Titanic passengers' identification number, whether or not they survived, seat class, name, sex, age, number of siblings and spouses on board, number of parents and children on board, ticket number, ticket fare, cabin, and the port they embarked from. We asked participants to find the relationship between different characteristics of passengers and rank the factors of survival. In the final task, the participants used *test.csv* which was similar to *train.csv* but has a different set of passengers and no data present on their survival. We asked partners to predict whether eight passengers survived based on their findings from the earlier tasks.

4.1.4 Experiment Setup

We required each pair to work together on the same computer for the study to observe peer interactions. One of the researchers moderated each experiment to record each session, and answer questions about the data and tasks. For each participant group we recorded the audio and screen while they worked on the tasks for the study. We provided an external mouse and keyboard for participants in addition to paper and writing utensils for taking notes.

Participants were allowed to choose any software to use for completing the tasks. The only restriction was they could not use the Internet. We prevented internet use to observe user knowledge during peer interactions without looking up information online. Pairs were only allowed to use the Internet to download software not installed on the test machine at the beginning of the session before starting the tasks. Participants used Windows 10 machine to complete the study with several data analysis programs installed including Microsoft Excel 2016 [6], JMP Pro 12 [14], MySQL Workbench 6.3 [31], Python (command line) [33], PyCharm [32], R (command line and GUI) [34], and RStudio [35]. Participants were allowed to request additional programs to use before the study if they were free and publicly available. Periscope[3] was the only software requested by a group that we were not able to provide because the software is proprietary.

4.2 Research Questions

This subsection explains how we collected data for each of our research questions during the experiment.

[1]https://ncsu-las.org/

[2]https://www.kaggle.com/c/titanic

[3]https://www.periscopedata.com/

4.2.1 Effectiveness

We created a scoring system to determine whether a peer interaction was effective or ineffective to help us answer our two research questions. The scoring rubric tracked the usage of the proposed tool during sessions in our study. We categorized each recommendation using a three-point Likert-type item scale based on the operations of recommendees when they had an opportunity to use the suggested tool:

- **3** Recommendee always or mostly uses recommended tool
- **2** There were no opportunities to use the tool later in the study after it was recommended
- **1** Recommendee mostly ignores or never uses recommended tool

An effective recommendation received a three, an ineffective recommendation received a one, and a recommendation where we could not determine the effectiveness received a two. We looked for instances where the recommendee selected a less efficient method to complete their actions to determine when a peer ignored an opportunity to use the recommended tool. Our results focus only on the effective and ineffective categories.

4.2.2 What characteristics of peers make recommendations effective?

We analyzed the Dialogue between participants to determine if peer characteristics play a role in recommendation effectiveness. The four characteristics we observed are politeness, persuasiveness, receptiveness, and time pressure. These characteristics were selected based on prior work regarding peer interactions as well as in psychology with a focus on human behavior and decision-making. We used a valence scale to calculate a score for the first three peer characteristics we studied:

- **+1** Participant obeyed a specific criteria
- **0** Participant neither obeyed nor violated a criteria
- **-1** Participant violated a specific criteria

This scale was used to categorize peer interactions by politeness (polite, neutral, impolite), persuasiveness (persuasive, unpersuasive), and receptiveness (receptive, neutral, unreceptive). Persuasiveness did not include a neutral category because, contrary to the other characteristics, omitting any of the criteria was a violation of the definition.

We present these three peer characteristics and related work regarding these traits in the sections below. Each characteristic consists of a table containing the definition as well as positive and negative examples of each criteria. The quotes in the examples use participant identifiers from our study, where L is the prefix depicting professional laboratory data analysts and the S prefix represents a student participant. The final set of criteria we agreed on to perform the data analysis for politeness, persuasiveness, and receptiveness can be found in Appendix C.

Politeness.

We hypothesize that politeness is more likely to improve tool adoption during peer interactions than impoliteness. Previous research on politeness and peer interactions supports this hypothesis. Whitworth examined Microsoft Clippy and suggested that politeness plays an important role in human interactions with computers [40]. Also, Murphy-Hill and colleagues previously studied peer interactions and interviewed participants to find that "respect" and "trust" between learners and teachers were important for the effectiveness of peer interactions [29]. Table 1 presents the criteria we used to classify interactions using Leech's six maxims for politeness: Tact, Generosity, Approbation, Modesty, Agreement, and Sympathy [20].

Persuasiveness.

We used previous research to hypothesize that persuasive interactons are more effective for tool discovery. Fogg outlined eight best practices for designing persuasive technology [9], and argues persuasiveness is important in convincing users to adopt desired behavior through software. Additionally, Murphy-Hill uncovered developer disinterest in learning new tools is a barrier to peer interactions [29]. Persuasiveness is necessary to convince users that a new feature should be adopted over existing methods to reach a goal. The criteria for persuasiveness is presented in Table 2 and uses the three features of persuasive messages described by Shen to measure persuasiveness of recommenders in our study: Content, Structure, and Style [36].

Receptiveness.

We hypothesize that receptiveness improves the success of tool discovery during peer interactions. The second step of Fogg's best practices for designing persuasive technology is to "Choose a receptive audience", and he provides two considerations for what makes a receptive audience: demonstrating a desire to adopt the target behavior and familiarity with the technology [9]. Receptiveness also played a role in Murphy-Hill's prior study on peer interactions as one participant stated "differences [in skill sets] make the collaboration interesting, but the similarities make the collaboration easier" [29]. He also found that unfamiliarity and use of different environments is a barrier to peer interaction. We used "Demonstrate Desire" and "Familiarity" to categorize interactions and define these criteria in Table 3.

Time Pressure.

Based on prior work, we hypothesize that time pressure will negatively impact the effectiveness of recommendations. Andrews and Smith assert time constraints affect decision-making in marketing by stifiling creativity, reducing exploratory thinking, and forcing a dependence on familiar approaches [2]. Additionally, Murphy-Hill's study on peer interactions identified time pressure through project release deadlines as a barrier to peer interactions [29]. During our study we did not strictly enforce time constraints for completing tasks, but we recommended each group to spend approximately 7-8 minutes on each one. We measured time pressure by searching for statements mentioning time from participants or the moderator before or during a recommendation. If we determined either participant made a statement regarding time during or before a peer interaction, then we categorized the recommendation as being under time pressure.

An example of time pressure influencing recommendation effectiveness in our study occurred between L13 and L14. The pair spent approximately 30 minutes working on the first task, and while L13 was driving she noted "I think we have like, four minutes left" for completing the preliminary tasks. She moved on to the next task and seconds later L14 recommended using the IF function in Excel for the second

Politeness Criteria		
Tact	Definition	Minimize cost and maximize benefit to peer
	Polite	“We can do all of it together, just sort by level.” - S9
	Impolite	“We can do a histogram...which is always sort of a pain in the butt to do in Excel.” - L14
Generosity	Definition	Minimize benefit and maximize cost to self
	Polite	“CONCATENATE you can do. I can do this for you, very easily.” - S10
	Impolite	“Maybe you should write a python script for this.” - L6
Approbation	Definition	Minimize dispraise and maximize praise of peer
	Polite	“I’m not as good at the Excel stuff as you are.” - L5
	Impolite	“This is useless.” - S14
Modesty	Definition	Minimize praise and maximize dispraise of self
	Polite	“From whatever limited knowledge of data analysis I have, I think you need to create a linear regression model...” - S14
	Impolite	“I’m very good at Paint.” - S10
Agreement	Definition	Minimize disagreement and maximize agreement between peers
	Polite	“Do you want to use Python?” - S8
	Impolite	“No, no, no...Don’t you want it comma separated? That’s what I’m doing.” - S14
Sympathy	Definition	Minimize antipathy and maximize sympathy between peers
	Polite	“We can try JMP...” [“I haven’t done anything in JMP.”] “Neither have I!” - L14
	Impolite	“It doesn’t matter how you do it.” - L16

Table 1: Definition of politeness criteria and examples from the user study

Persuasiveness Criteria		
Content	Definition	Recommender provides credible sources to verify use of the tool
	Persuasive	“Go here, go to Data. Highlight that...Data, Sort, and it lets you pick two.” - L8
	Unpersuasive	“Let’s try to text filter, right?” - S5
Structure	Definition	Messages are organized by climax-anticlimax order of arguments and conclusion explicitness
	Persuasive	“I know that SUMIF is a type of function that allows you to combine the capabilities of SUM over a range with a condition that needs to be met.” - S3
	Unpersuasive	“There’s a thing on Excel where you can do that, where you can say if it is this value, include, if it is not, exclude...Yeah, IF.” - S11
Style	Definition	Messages should avoid hedging, hesitating, questioning intonations, and powerless language
	Persuasive	“Control-Shift-End” - S1
	Unpersuasive	“I guess we’re going to have to use some math calculations here, or a pivot table.” - L9

Table 2: Definition of persuasiveness criteria and examples from the user study

Receptiveness Criteria		
Demonstrate Desire	Definition	User showed interest in discovering, using, or learning more information about the suggested tool
	Receptive	“That was cool, how [the column] just populated.” - S4
	Unreceptive	“No, don’t do a sort. Use a filter.” - S10
Familiarity	Definition	User explicitly expresses familiarity with the environment
	Receptive	“Control shift...how do I select it completely?” - S2
	Unreceptive	“I’ve never done anything in JMP.” - L10

Table 3: Definition of receptiveness criteria and examples from the user study

task. However, L13 ignored the navigator's recommendation and used her own methods. In this case time pressure played a role in the ineffective Peer Recommendation by influencing the driver to apply a selection rule that limited exploratory thinking and tool discovery to accomplish the goal.

4.2.3 *What types of tools are most effectively recommended during peer interactions?*

We hypothesize that percievable tools are more effective in recommendations than imperceptible ones. Both types of tools provide additional functionality to users, but Murphy-Hill noted that the type of tool can impact peer interations and argues recommendation systems should have noticeable causes and effects [29]. We further examined this by gathering the tools recommended between participants in our study. After collecting the tools suggested in our study, we categorized them into two different types of tools: observable and non-observable.

Observable refers to tools and features that are visible through a user interface. Examples of observable tools recommended by participants during our study include software programs such as Excel, Python, and R in addition to built-in software features such as =AVERAGE(), Sort, Histograms, Text to Columns, and pivot tables in Excel. Non-observable tools are features that do not have a user interface, such as keyboard shortcuts. Examples we observed include Control-Space in PyCharm for code completion, dragging the corner of a cell to automatically copy a formula in Excel, Control-V to paste, and Control-S to save.

4.3 Data Analysis

During each study session, the moderator took note of possible occurrences of peer observation or recommendation. The notes were used to discuss recommendations that occurred during the study for the post-interview. Two independent researchers reviewed and coded each recording to find and verify instances of peer interactions and categorize the peer characteristics for each tool recommendation. We coded a peer interaction if it fit our model from Section 3. The specific information collected for each tool recommendation occurrence found is listed in Appendix B.

The coders watched each recording to manually collect data for our results. We iteratively coded the recordings and modified our criteria to clearly define what we were looking for during peer interactions. After finalizing the criteria, we divided the coding process into two parts. The first part involved going through each of the peer interactions recorded by each coder to determine if it was an actual tool recommendation. Then, we went through the confirmed list of recommendations and compared each instance based on the criteria for politeness, persuasiveness, and receptiveness.

When we disagreed on the existence a recommendation or any of the characteristics, the two researchers watched the clip of the instance in question together, explained the reasoning behind their individual rating, debated the reasoning behind their decision, and came to an agreement after the discussion. We calculated our interrater agreement for politeness ($\kappa = 0.50$), persuasiveness ($\kappa = 0.28$), and receptiveness ($\kappa = 0.51$) using Cohen's Kappa. According to Landis' measurement of observer agreement, which has been used by studies in this field [37], our agreement for politeness and persuasiveness had a moderate strength of agreement while persuasiveness had a fair strength of agreement [18].

Politeness	**Effective**	**Ineffective**	**Unknown**
Polite	52% ($n = 16$)	19% ($n = 6$)	29% ($n = 9$)
Neutral	51% ($n = 55$)	26% ($n = 28$)	23% ($n = 25$)
Impolite	50% ($n = 6$)	25% ($n = 3$)	25% ($n = 3$)

Table 4: Rate of effectiveness for politeness

Persuasiveness	**Effective**	**Ineffective**	**Unknown**
Persuasive	33% ($n = 5$)	33% ($n = 5$)	33% ($n = 5$)
Unpersuasive	53% ($n = 72$)	24% ($n = 32$)	24% ($n = 32$)

Table 5: Rate of effectiveness for persuasiveness

5. RESULTS

We analyzed 151 total recommendations between participants in our study. We categorized 77 as effective, 37 as ineffective, and 37 as unknown. Each group averaged approximately 12 recommendations with a maximum of 28 and minimum of 5. The first phase of the study contributed 104 interactions with 50 effective, 23 ineffective, and 30 unknown between students. LAS participants in the second phase added 47 interactions consisting of 26 effective, 14 ineffective, and 7 unknown. We used the Wilcoxon rank sum test for evaluating ordinal data with an alpha level of $\alpha = 0.05$ to statistically analyze effective and ineffective interactions unless otherwise specified. Odds ratios (OR) were used to calculate the effect size for statistically significant results.

5.1 Peer Characteristics

We categorized each recommendation based on politeness, persuasiveness, receptiveness, and time pressure. Figure 1 presents the classifications of peer interactions for each of the characteristics we observed.

Politeness

We identified 108 neutral, 31 polite, and 12 impolite recommendations. Table 4 presents the effectiveness for each polite, neutral, and impolite interactions. Most were categorized as neutral because many participants suggested tools without explicitly obeying the criteria we were looking for in politeness. Polite recommendations had a higher rate of effectiveness than impolite and neutral categories on average, but we were unable to identify politeness as not a significant factor in the effectiveness of tool recommendations ($p = 0.6244$).

Persuasiveness

Participants in our study were rarely persuasive during interactions according to our criteria. There were only 15 persuasive recommendations out of 151 total peer interactions. A breakdown of persuasiveness and effectiveness are shown in Table 5. Contrary to prior work, unpersuasive recommendations had a higher rate of effectiveness than persuasive ones. The results from our user study were unable to identify persuasiveness as a significant factor in recommendation effectiveness ($p = 0.2191$).

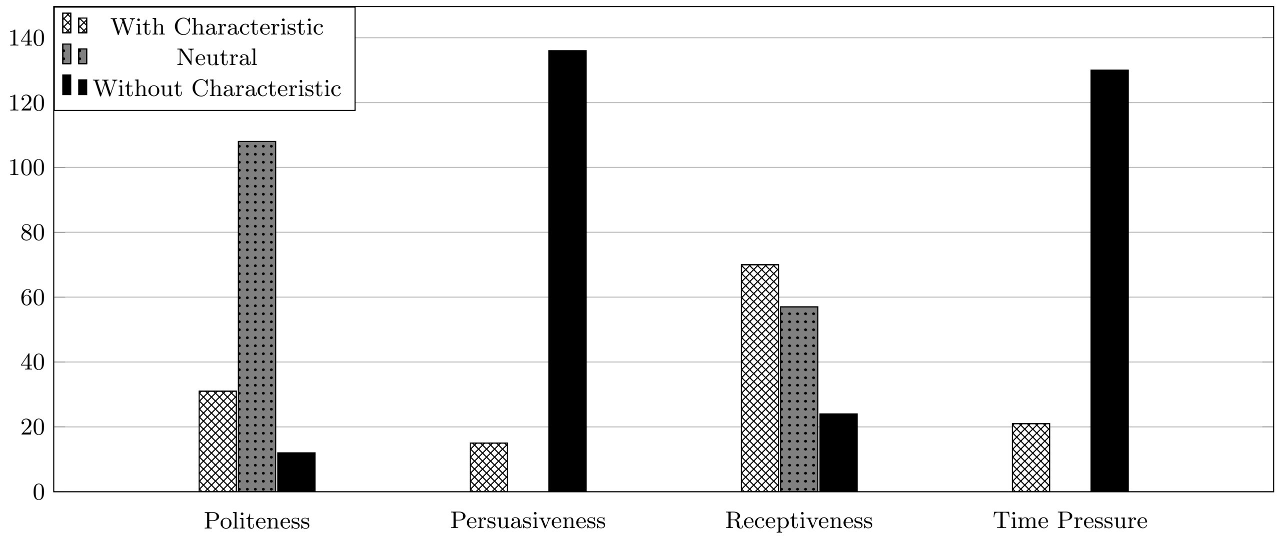

Figure 1: Number of recommendations for each peer characteristic

Receptiveness	Effective	Ineffective	Unknown
Receptive	61% ($n = 43$)	13% ($n = 10$)	24% ($n = 17$)
Neutral	49% ($n = 28$)	25% ($n = 14$)	26% ($n = 15$)
Unreceptive	25% ($n = 6$)	54% ($n = 13$)	21% ($n = 5$)

Table 6: Rate of effectiveness for receptiveness

Time Pressure?	Effective	Ineffective	Unknown
Yes	33% ($n = 7$)	43% ($n = 9$)	24% ($n = 5$)
No	54% ($n = 70$)	22% ($n = 28$)	25% ($n = 32$)

Table 7: Rate of effectiveness for time pressure

Receptiveness

For receptiveness, we categorized 70 receptive interactions, 57 neutral interactions, and 24 unreceptive interactions in our study. The effectiveness rate for this category is displayed in Table 6. Receptive interactions had the highest rate of effectiveness out of all the characteristics we studied with approximately 61% of recommendations classified as receptive also categorized as effective. We also identified receptiveness as a significant factor in the effectiveness of tool recommendations between peers ($p < 0.0003, OR = 0.1073$).

Time Pressure

We categorized 21 out of 151 tool recommendations as being under time pressure. Table 7 shows that time constraints limited tool discovery negatively impacted the effectiveness of recommendations. Interactions that were categorized as being under time pressure were less effective than those that weren't. We also discovered time pressure plays a significant role in the outcome of peer interactions using Pearson's chi-squared test for categorical data ($\alpha = 0.05, p = 0.0283, OR = 3.2142$).

Tool Type	Effective	Ineffective	Unknown
Observable	50% ($n = 62$)	26% ($n = 32$)	24% ($n = 29$)
Non-Observable	54% ($n = 15$)	18% ($n = 5$)	29% ($n = 8$)

Table 8: Rate of effectiveness for type of tool

Type	Effective	Ineffective	Unknown
POE	17% ($n = 1$)	0% ($n = 0$)	83% ($n = 5$)
PRE	67% ($n = 6$)	22% ($n = 2$)	11% ($n = 1$)
POU	32% ($n = 15$)	11% ($n = 5$)	57% ($n = 27$)
PRU	62% ($n = 55$)	34% ($n = 30$)	5% ($n = 4$)

Table 9: Rate of effectiveness for Peer Observation Expected (POE), Peer Recommendation Expected (PRE), Peer Observation Unexpected (POU), and Peer Recommendation Unexpected (PRU)

5.2 Tools

We also classified types of tools recommended in our study as observable and non-observable. Observable tools were by far the most recommended types of features with 123 recommendations compared to only 28 non-observable tools. Although observable tools were suggested more often, Table 8 shows non-observable features were more effectively recommended. However, we were unable to determine that tool observability played a significant role in the effectiveness of recommendations using Pearson's chi-squared test ($\alpha = 0.05, p = 0.4329$).

5.3 Recommendations and Expectations

This section presents additional results we analyzed from data collected in our study. We evaluated the types of peer interactions based on the effectiveness categories using Pearson's chi-squared test with an alpha level of $\alpha = 0.05$.

Observation vs. Recommendation

We evaluated the two types of peer interactions to determine if they played a role in the effectiveness of tool discovery. We detected 53 Peer Observations and 98 Peer Recommendations in our study. Peer Recommendations appeared more often in our study and had the higher rate of effectiveness at 62% while about 30% of Peer Observations were categorized as effective while 60% were identified as unknown. We were unable to conclude that the type of peer interaction was a significant factor in the effectiveness of tool recommendations ($p = 0.2597$).

Expected vs. Unexpected

We also analyzed if expectation of a recommendation had an affect on effectiveness. For *Peer Observations* and Peer Recommendations, we observed 15 expected tool recommendations and 136 unexpected. The effectiveness rates and counts of recommendations based on expected and unexpected types of peer interactions are presented in Table 9. Unexpected recommendations overall had a higher rate of effectiveness at 51% compared to 47% for expected, but we were unable to determine that users expecting a recommendation was a significant factor in effectiveness ($p = 0.4235$).

5.4 Qualitative Results

Our interviews and surveys suggest most participants expected peer characteristics to impact effectiveness, however they did not take these qualities into account when making recommendations. When participants gave explanations for why they decided to make recommendations, 69% used "I" statements noting their own knowledge and experience. S7 embodies this attitude by stating he suggested using Find in Excel because "This was a better way to solve the problem at hand and I have used it in similar situations". This suggests peers are often motivated to offer suggestions based on their own expertise instead of to benefit their partner.

Additionally, we asked subjects why they phrased their recommendations the way they did, and 74% mentioned using language that was easier or shorter for themselves. S2 demonstrates this after responding he recommended Control-Shift-End in Excel to help his partner but phrased it in the "simplest way I could phrase it". We found that participants preferred brevity when recommending tools, which could explain why few recommendations were categorized as polite and persuasive.

6. DISCUSSION

This section presents a summary of our results, implications for designing recommendation systems, threats to the validity of our experiment, and future work.

6.1 Summary

Our results were unable to show that politeness, persuasiveness, and tool observability are significant factors for tool recommendation effectiveness between peers. This counters previous research that suggests these qualities influence human behavior, decision-making, and tool discovery. Furthermore, contrary to our original hypotheses, we found that unpersuasive interactions and non-observable tools were more effective in tool discovery. Alternate theories may explain these observations in our study.

Our criteria for persuasiveness expected lengthened peer interactions from users providing context and explaining their recommendations, but our qualitative results show that participants phrased suggestions to be brief and concise. Additionally, we noticed that peers often used poor style and weak language during interactions in our study. This complies with previous research Wood and colleagues who note that the quality of the argument impacts effectiveness while message length has little impact on persuasiveness [41].

We also expected observable tools to be more effectively recommended than non-observable features. Many shortcuts are non-observable tools without a graphical user interface, and previous research by Lane points out that keyboard shortcuts are underused but are more efficient than observable tools such as menus and icon toolbars [19]. Appert and Zhai also outline the benefits of using stroke shortcuts for cognitive learning and recall [3].

6.2 Implications

Our results indicate receptiveness and time pressure are characteristics that significantly impact the outcome of peer interactions. This section provides insight into improving automated recommendation systems using these qualities to help increase the effectiveness of recommendations and improve tool discovery.

6.2.1 Increase Receptivity

Receptiveness may be difficult to implement in systems because it is based on how users respond to recommendations, which recommenders cannot control. The criteria we used for measuring receptiveness in our study involve users demonstrating desire to use a tool and expressing familiarity with the technology. Incorporating these qualities into automated recommendation systems can help create more effective recommendations and improve tool discoverability.

Demonstrate Desire

We looked for instances where recommendees explicitly communicated a desire to use recommended tools as a criteria for receptiveness. During one interaction, L11 was driving when L12 recommended using multi-level sort in Excel to sort by multiple columns. L11 was unfamiliar with that functionality, but demonstrated a desire to use the new tool by saying "Oh! Add level! Yes, awesome!". The multi-level sort was then adopted by the recommendee and used for the remaining tasks after she demonstrated a desire to use it.

Predicting user desire can be valuable in improving tool discoverability. Our results propose that developers should prioritize building usable and approachable tools to increase users' receptivity to recommendations. One possible way to collect this data is to analyze users' search history to gather their most popular queries in the software to determine what functionality and tools a user is interested in learning about. History-based recommendations systems also propose tools based on previous actions of the user and can help determine what users desire to use. Additionally, Fischer notes that help systems should not just respond but notice and actively make suggestions to users while completing tasks [8]. Goal-recognition techniques, such as *CIGAR* by Hu and colleagues [12], can be useful for creating models and predicting user goals based on intermediate actions.

Foster Familiarity

We also searched for statements where participants consciously expressed familiarity or unfamiliarity with suggested tools and technologies to measure receptiveness. Previous research suggests users are more likely to adopt target behaviors if they have familiarity. Unfamiliarity led to many ineffective recommendations during our study, such as an interaction between S9 and S10. They were discussing the best way to display relationships between data when S10 makes an unexpected peer recommendation to use R to create a plot. He also notes a benefit of using R, saying it will only take about two lines of code. However, the driver S9 responds by saying "I don't know R". S9's unfamiliarity with the R statistical computing language and it's environment led to an ineffective recommendation of a tool that could have been useful, but it was never used or mentioned again for the remainder of the study session.

Tool recommendation systems can foster familiarity for users by assessing the user's current knowledge. According to our results, systems should avoid recommending random and obscure tools may be more effective because participants were more likely to ignore unfamiliar features. History-based systems offer are also helpful in making sure recommendees have familiarity with proposed tools. Another possibility is to rank or group the features in software based on similarity and usage. Murphy-Hill and colleagues have explored this by ranking tools using collaborative filtering based on command patterns from colleagues [28].

6.2.2 *Minimize Time Pressure*

We identified time pressure as a significant factor for recommendation effectiveness. Previous research shows that increased time pressure negatively impacts exploratory thinking [2] and prevents peer interaction [29]. These time constraints are primarily enforced through external factors such as peers or project deadlines, but automated recommendation systems can minimize time pressure by improving algorithms to recognize user actions and anticipate the goal early. This will allow systems to recommend tools sooner and provide users with more time to decide whether or not to adopt the proposed feature and gives them an opportunity to learn the new functionality.

6.3 Threats to Validity

There are several threats to the internal validity of this study. First, although the data was contained in two comma-separated values files, Microsoft Excel was the default program to open the specific file type. A majority of groups used Excel to complete the tasks and this may have influenced participants' decisions to use it over another program. Second, participants were allowed to request the data analysis software of their choice for the study, however they did not know the tasks beforehand. Participants could not make informed recommendations about programs to use without knowledge of the tasks, such as one participant (L13) who noted during the study she should have requested Tableau[4].

Threats to external validity include limitations to our criteria and study design. One external threat was that we only observed politeness, persuasiveness, receptiveness, time pressure, and types of tools, but other character traits can also possibly influence effectiveness. Next, we only measured the effectiveness of recommendations within the duration of the study and did not track long term use to determine if recommendees adopted discovered features in their normal software usage. Another external threat was our scoring for recommendations. We treated every compliance or violation of a characteristic's criteria as an equal action. For example, our scoring system allowed for polite and impolite statements to cancel each other out but that is not the case in a real-world social situation.

Futhermore, we only categorized recommendations based on what was explicitly mentioned during the dialogue of a recommendation, but did not account for implicit actions or context including peer relationships, social cues, and behavior. Finally, different cultures have varying criteria and cultural norms to describe the peer characteristics we studied. For example, Huang used Leech's politeness maxims to compare and contrast differences in western and Chinese concepts of politeness [13].

[4]http://www.tableau.com/trial/data-analysis-software

6.4 Future Work

Future research for this project includes examining the effectiveness of additional characteristics on recommendations between peers. Other participant qualities can potentially influence the effectiveness of recommendations are previous experience, the relationship between partners, the success of previous recommendations made in the study, and more. Other factors can also be analyzed to determine if they play a role in tool discovery and adoption such as the length of the recommendation dialogue, difficulty of the task, and how long it disrupts users from their current task.

The study completed for this research focused on data analysis, but future work could replicate this experiment using different study tasks for populations to examine recommendation effectiveness. Examining different populations and tasks could impact the peer characteristics and types of tools that make recommendations effective, for example observing computer programmers completing code refactoring tasks in integrated development environments. Based on the results of this experiment, researchers and toolsmiths should develop new tools that integrate receptiveness by focusing on the desire and familiarity of users to improve tool discovery and recommend useful software features.

7. CONCLUSION

This research examines what makes peer interactions effective for tool discovery by introducing a model to define tool recommendations and analyzing peer characteristics and types of tools recommended during peer interactions between participants in our study. Our results show that the receptiveness of recommendees and absence of time pressure have a significant impact on the effectiveness of peer interactions. This suggests that automated recommendation systems should clearly recommend tools early in users' actions based on their desire and knowledge in order to increase tool discoverability and adoption in software.

8. ACKNOWLEDGMENTS

Thanks to all of the student and LAS participants who volunteered for this study. Also special thanks to Emerson Murphy-Hill for his advising and guidance, Justin Middleton for helping with data analysis, and Esha Sharma for designing and implementing the first phase of this project.

9. REFERENCES

[1] *The Handbook of Task Analysis for Human-Computer Interaction*, chapter GOMS Models for Task Analysis. Lawrence Erlbaum Associates, 2004.

[2] J. Andrews and D. C. Smith. In search of the marketing imagination: Factors affecting the creativity of marketing programs for mature products. *Journal of Marketing Research*, pages 174–187, 1996.

[3] C. Appert and S. Zhai. Using strokes as command shortcuts: Cognitive benefits and toolkit support. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 2289–2298, New York, NY, USA, 2009. ACM.

[4] A. Cockburn and L. Williams. Extreme programming examined. chapter The Costs and Benefits of Pair Programming, pages 223–243. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[5] W. Damon. Peer education: The untapped potential. *Journal of Applied Developmental Psychology*, 5(4):331 – 343, 1984.

[6] Microsoftexcel. https://products.office.com/en-us/excel.

[7] L. Findlater, J. McGrenere, and D. Modjeska. Evaluation of a role-based approach for customizing a complex development environment. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1267–1270, New York, NY, USA, 2008. ACM.

[8] G. Fischer, A. Lemke, and T. Schwab. *Active help systems*, pages 115–131. Springer Berlin Heidelberg, Berlin, Heidelberg, 1984.

[9] B. Fogg. Creating persuasive technologies: An eight-step design process. In *Proceedings of the 4th International Conference on Persuasive Technology*, Persuasive '09, pages 44:1–44:6, New York, NY, USA, 2009. ACM.

[10] J. L. Franke, J. J. Daniels, and D. C. McFarlane. Recovering context after interruption. In *Proceedings 24th Annual Meeting of the Cognitive Science Society (CogSci 2002)*, pages 310–315, 2002.

[11] T. Grossman, G. Fitzmaurice, and R. Attar. A survey of software learnability: Metrics, methodologies and guidelines. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 649–658, New York, NY, USA, 2009. ACM.

[12] D. H. Hu and Q. Yang. Cigar: Concurrent and interleaving goal and activity recognition. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, AAAI'08, pages 1363–1368. AAAI Press, 2008.

[13] Y. Huang. Politeness principle in cross-culture communication. *English Language Teaching*, 1(1):96, 2008.

[14] Jmp pro. http://www.jmp.com/en_us/home.html.

[15] B. Johnson, Y. Song, E. Murphy-Hill, and R. Bowdidge. Why don't software developers use static analysis tools to find bugs? In *Proceedings of the 2013 International Conference on Software Engineering*, ICSE '13, pages 672–681, Piscataway, NJ, USA, 2013. IEEE Press.

[16] E. Kalliamvakou, D. Damian, K. Blincoe, L. Singer, and D. M. German. Open source-style collaborative development practices in commercial projects using github. In *Proceedings of the 37th International Conference on Software Engineering - Volume 1*, ICSE '15, pages 574–585, Piscataway, NJ, USA, 2015. IEEE Press.

[17] J. A. Krisch. Why you hated clippy, that annoying microsoft paperclip, 2016.

[18] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, 1977.

[19] D. M. Lane, H. A. Napier, S. C. Peres, and A. Sándor. Hidden costs of graphical user interfaces: Failure to make the transition from menus and icon toolbars to keyboard shortcuts. *International Journal of Human-Computer Interaction*, 18(2):133–144, 2005.

[20] G. Leech. *Principles of Pragmatics*. Longman linguistics library ; title no. 30. Longman, 1983.

[21] F. Linton, D. Joy, H. peter Schaefer, and A. Charron. Owl: A recommender system for organization-wide learning, 2000.

[22] F. Linton, D. Joy, and H.-P. Schaefer. Building user and expert models by long-term observation of application usage. In *Proceedings of the Seventh International Conference on User Modeling*, UM '99, pages 129–138, Secaucus, NJ, USA, 1999. Springer-Verlag New York, Inc.

[23] W. Maalej, R. Tiarks, T. Roehm, and R. Koschke. On the comprehension of program comprehension. *ACM Trans. Softw. Eng. Methodol.*, 23(4):31:1–31:37, Sept. 2014.

[24] D. Malachowski. Wasted time at work costing companies billions. *San Francisco Chronicle*, 11, 2005.

[25] C. Maltzahn. Community help: Discovering tools and locating experts in a dynamic environment. In *Conference Companion on Human Factors in Computing Systems*, CHI '95, pages 260–261, New York, NY, USA, 1995. ACM.

[26] J. McGrenere and G. Moore. Are we all in the same "bloat"? In *Proceedings of the Graphics Interface 2000 Conference, May 15-17, 2000, Montr'eal, Qu'ebec, Canada*, pages 187–196, May 2000.

[27] E. Murphy-Hill. Continuous social screencasting to facilitate software tool discovery. In *Proceedings of the 34th International Conference on Software Engineering*, ICSE '12, pages 1317–1320, Piscataway, NJ, USA, 2012. IEEE Press.

[28] E. Murphy-Hill, R. Jiresal, and G. C. Murphy. Improving software developers' fluency by recommending development environment commands. In *Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering*, FSE '12, pages 42:1–42:11, New York, NY, USA, 2012. ACM.

[29] E. Murphy-Hill, D. Y. Lee, G. C. Murphy, and J. McGrenere. How do users discover new tools in software development and beyond? *Computer Supported Cooperative Work (CSCW)*, 24(5):389–422, 2015.

[30] E. Murphy-Hill and G. C. Murphy. Peer interaction effectively, yet infrequently, enables programmers to discover new tools. In *Proceedings of the ACM 2011*

Conference on Computer Supported Cooperative Work, CSCW '11, pages 405–414, New York, NY, USA, 2011. ACM.
[31] Mysql workbench. http://www.mysql.com/products/workbench/.
[32] Pycharm. https://www.jetbrains.com/pycharm/.
[33] Python. https://www.python.org/.
[34] R. https://www.r-project.org/.
[35] Rstudio. https://www.rstudio.com/.
[36] L. Shen and E. Bigsby. The effects of message features: content, structure and style. *The SAGE handbook of persuasion developments in theory and practice*, 2012.
[37] J. Smith, B. Johnson, E. Murphy-Hill, B. Chu, and H. R. Lipford. Questions developers ask while diagnosing potential security vulnerabilities with static analysis. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, ESEC/FSE 2015, pages 248–259, New York, NY, USA, 2015. ACM.
[38] W. Snipes, A. R. Nair, and E. Murphy-Hill. Experiences gamifying developer adoption of practices and tools. In *Companion Proceedings of the 36th International Conference on Software Engineering*, ICSE Companion 2014, pages 105–114, New York, NY, USA, 2014. ACM.
[39] P. Viriyakattiyaporn and G. C. Murphy. Improving program navigation with an active help system. In *Proceedings of the 2010 Conference of the Center for Advanced Studies on Collaborative Research*, CASCON '10, pages 27–41, Riverton, NJ, USA, 2010. IBM Corp.
[40] B. Whitworth. Polite computing. *Behaviour & Information Technology*, 24(5):353–363, 2005.
[41] W. Wood, C. A. Kallgren, and R. M. Preisler. Access to attitude-relevant information in memory as a determinant of persuasion: The role of message attributes. *Journal of Experimental Social Psychology*, 21(1):73–85, 1985.
[42] S. Xiao, J. Witschey, and E. Murphy-Hill. Social influences on secure development tool adoption: Why security tools spread. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing*, CSCW '14, pages 1095–1106, New York, NY, USA, 2014. ACM.

APPENDIX

A. TASKS

A.1 Student Tasks

Please do not use the internet to answer the questions.

Please work on the tasks together in pairs.

The time to spend on the task is 60 mins, which comes to 7 to 8 mins on each task. However, this is just a recommendation.

Please use any tool you are comfortable with to answer these questions. This computer has Excel, Rstudio, Python, SAS JMP Pro 12, and MySQL Workbench. If you need anything else, we can download that as well.

The total time for the task is 60 mins. Please do not spend more than 45 mins in the training tasks which comes to around 7 to 8 mins on each question.

Task: For a to e, please to describe the relationship. For f the factors should be ranked from the most significant to least significant. You can use mean, mode, etc. to explain the ranking.

Using training data: (45 mins)

a What is the relationship between the (gender, age) and number of sibling/spouse (SibSp) traveling?

b What is the relationship between the Title(you can find this in the name) and the number of children/parents (Parch) traveling?

c What is the relationship between the Title(you can find this in the name) and the age and gender?

d What is the relationship between (class, fair) and age?

e What is the relationship between (the fare and class) and the city embarked?

f Please rank all the factors (a-e) for their contribution to survival. The factors should be ranked from the most significant to least significant. You can use mean, mode, etc. to explain the ranking.

On the testing data: (10 mins)

Please find whether these people survived.

a. Chaffee, Mrs. Herbert Fuller (Carrie Constance Toogood)
b. Robins, Mr. Alexander A
c. Peltomaki, Mr. Nikolai Johannes
d. Abelseth, Mr. Olaus Jorgensen
e. Mulvihill, Miss. Bertha E
f. Thomas, Mr. John
g. Daniels, Miss. Sarah
h. Delalic, Mr. Redjo

A.2 LAS Tasks

Do not use the internet to answer the questions.

Please work on the tasks together in pairs.

Please use any tool you are comfortable with to answer these questions. This computer has Excel, R, Rstudio, Python (command-line), SAS JMP Pro 12, and MySQL Workbench. If you prefer to use another software we can download that as well. We will also provide paper and writing utensils.

The total time for the study is approximately 60 mins. Please do not spend more than 45 mins on the tasks below.

Task: For a to c, please describe the relationship between the categories. For d the factors should be ranked from the most significant to least significant. You can use mean, mode, etc. to explain the ranking.

Using train.csv: (35 mins)

a. What is the relationship between the gender (Sex), age, and the number of siblings/spouse traveling (SibSp)?

b. What is the relationship between the Title (you can find this in the name- Mr., Mrs., Ms., Miss., Master., Dr., etc. There may be more than this) and the number of children/parents (Parch) traveling?

c. What is the relationship between the fare, class (Pclass), and age?

d. Rank the factors for their contribution to survival. The factors should be ranked from the most significant to least significant. You can use any methods to explain the ranking. (1 = survived, 0 = died)

When you are comfortable with your answers to the tasks above or time is running out, please move on to the final task. Again, you must work together in pairs and you may not use the internet to answer the questions.

Using test.csv and your results from the previous task: (10 min.)

Predict whether the following passengers survived:

a. Chaffee, Mrs. Herbert Fuller (Carrie Constance Toogood)

b. Robins, Mr. Alexander A

c. Peltomaki, Mr. Nikolai Johannes

d. Abelseth, Mr. Olaus Jorgensen

e. Mulvihill, Miss. Bertha E

f. Thomas, Mr. John

g. Daniels, Miss. Sarah

h. Delalic, Mr. Redjo

B. DATA COLLECTED

- The type of peer interaction (Peer Observation Expected, Peer Observation Unexpected, Peer Recommendation Expected, or Peer Recommendation Unexpected),
- the approximate time in the video the recommendation took place,
- which participants are the driver and navigator,
- the study task,
- the method of the driver and navigator (if possible),
- the name and type of the recommended feature,
- a transcript of the dialogue concerning the new tool,
- the reaction of the recommendee,
- instances in the study where the tool was re-used,
- instances where the tool was ignored for a less efficient method,
- the effectiveness, politeness, persuasiveness, and receptiveness scores,
- whether the recommendations was under time pressure, and
- if the recommendation was discussed during the interview and time of discussion in the video.

C. PEER CHARACTERISTICS

This section of the appendix presents the final set of criteria that the two independent coders agreed upon for Politeness, Persuasiveness, and Receptiveness. We specifically searched for the following when analyzing the study videos and scoring the peer characteristics.

C.1 Politeness

C.1.1 Tact

+1 Recommender provides beneficial reason for using tool

0 No statement on advantages or disadvantages of tool

-1 Recommender notes weakness of using suggested tool

C.1.2 Generosity

+1 Recommender offers to do the work for the recommendee

0 No statement on either peer doing work

-1 Recommender makes partner complete all the work

C.1.3 Approbation

+1 Recommender praises or compliments partner

0 No statements of praise or insults

-1 Recommender insults or offends partner

C.1.4 Modesty

+1 Recommender expresses humility in knowledge or abilities

0 No statements of humility or arrogance

-1 Recommender praises their own knowledge or abilities

C.1.5 Agreement

+1 Recommender agrees with statements made by partner or uses inclusive language

0 No statements of agreement or disagreement

-1 Recommender disagrees or argues with partner

C.1.6 Sympathy

+1 Recommender expresses congratulations, commiseration, or expresses condolences

0 No statements regarding sympathy or apathy

-1 Recommender incites conflict, expresses dismissiveness, or enjoys pain of partner

C.2 Persuasiveness

C.2.1 Content

+1 Recommender explicitly explains why the tool suits the purpose by citing a source, relating to previous experience, explaining how it works, or presenting why it's useful

-1 Recommender does not provide any information explaining why to use the suggested tool

C.2.2 Structure

+1 Recommender presents the tool before explaining why it should be used

-1 Recommender explains why a tool should be used before saying the tool or does not provide content

C.2.3 Style

+1 Recommender avoids hedging, hesitating, recommending multiple features simultaneously, asking if a tool should be used, tag questions, and passive and powerless language (i.e. "I think", "I guess", "sort of", excessive number of "Uh...", etc.)

-1 Recommender uses the statements above in their recommendation

C.3 Receptiveness

C.3.1 Demonstrate Desire

+1 Recommendee explicitly expresses interest or asks questions to learn more information about tool

0 No statements demonstrating desire

-1 Recommendee explicitly expresses disinterest in using tool

C.3.2 Familiarity

+1 Recommendee explicitly expresses familiarity with tool and environment or compares to a familiar tool

0 No statements on familiarity

-1 Recommendee explicitly states they are unfamiliar with the tool or environment