ABSTRACT

The Mythical Man-Month: Essays on Software Engineering [4] by Dr. Fred Brooks is often required reading in many software engineering classes and recommended by practitioners throughout industry as a text with which computer scientists should be familiar. Yet, while many of the conceptual ideas that were first presented decades ago are still applicable today, aspects of the presentation of the material are lacking in inclusivity, posing additional challenges for faculty who are trying to increase diversity within the discipline. Thus, we propose a student-developed portrayal of some of the concepts in a modernized fashion, presented as a menu and incorporating ideas from the food-service industry, a model that more college students are directly familiar with than the surgical team.

INTRODUCTION

As a still relatively young and rapidly changing discipline, books in computer science often become dated and show their age. Despite that challenge, some have become classics that are deemed worth reading by academics and industry practitioners alike. The Mythical Man-Month: Essays on Software Engineering [4] by Dr. Frederick P. Brooks is one such example, appearing in whole or in part on many required reading lists for software engineering courses, listed in recent curricular guidelines [10], and included in compilations of recommended reading for computer science students [5] and developers [1, 3]. The original 1975 essays were based on Brooks’ experiences at IBM while managing the development of OS/360; later publications of the collection refined these essays and added some additional essays, including some which addressed critiques of the earlier essays. It is deemed influential enough by many that there is even work studying the citations and influence of The Mythical Man-Month to disciplines beyond software engineering and computer science [11].

Despite the additions of later chapters/essays and multiple publications, The Mythical Man-Month is not immune from criticism, due not only to some concepts aging poorly, but also because of its noted lack of inclusivity. Yet academic assessments of the book focus on changes in software engineering and project management [14], and conversations about the implications on diversity in the field are largely left to blogs and personal conversations. Even a course that, starting in Spring of 1998, used The Mythical Man-Month as the basis for discussions about software engineering for an entire semester focused on how students “expressed surprise that a text over 20 years old
could still be relevant today” [13], but did not raise concerns about its presentation. Similarly, a more recent work which provides “a more active approach to incorporating MMM into the classroom” [12] including lesson plans, frameworks for discussion, and criteria for evaluation likewise fails to encourage any discussion about limitations related to diversity. Yet, the gendered nature of the title is immediately obvious to people attentive to such details, and some readers note that male pronouns are used exclusively throughout the text for computer scientists, managers, and clients [7]. Other readers raise concerns over potentially discriminatory religious statements that are included: in the point about “on using two core programmers on a team the author makes a footnote of ‘Note God’s plan for marriage’” [15].

It is well-known that software engineering, and computer science as a whole, suffers from a lack of gender and racial diversity, both in the classroom and in industry [8]. Appropriately framing the classroom climate [2] has seemingly played a role in Harvey Mudd College’s success in recruiting and retaining women in the computer science major. While that work focused on the status of students among their peers, a supportive climate is also one in which students feel included, not marginalized or unwelcome; this may be particularly important when the work students are reading is that of an A.M. Turing Award recipient. Furthermore, psychology research has shown that even though confronting other people’s biases can be uncomfortable for people to do, it may in fact be effective tool for reducing occurrences of such bias in the future [6].

A REVISED PRESENTATION

In the software engineering course at Southwestern University students learn about best practices in software engineering while also working on a major group project. Along with a more traditional software engineering textbook, each student is required to read a book relevant to the field that would not generally be considered a textbook. They give a presentation to the class about the material, provide a handout which highlights some of the details they hope their peers will remember long-term, and are considered the go-to person for that material during the semester. Students are provided with descriptions of the books and given the opportunity to thumb through them before indicating their preferences. The number of available titles is more than the number of students (up to twice as large), and while the instructor makes assignments to further the course and program goals, students are frequently given their first or second choice, and never assigned a book they requested to not read.

The student author of this paper picked The Mythical Man-Month: Essays on Software Engineering as one of the five books that he would prefer to read. His optional statement about the reason for his preferences indicated that he chose books because of the valuable knowledge about industry that they purported to convey. Yet, as a young Latino at a liberal arts college, where at least one social justice course is required of all students, he was attentive to the language and examples used in the book. He also knew from his years in the department that the computer science faculty, including the course instructor (and paper co-author), actively promoted diversity in computing and attention to inequity. (Another book presented in the course by a different student
was Programmed Inequality: How Britain Discarded Women Technologists and Lost its Edge in Computing [9].) Thus, while not all students appreciated the importance of diversity to the extent he did, the student had the background knowledge and confidence to point out some of the limitations of the essays, while conveying the still-relevant software-engineering principles.

He titled his presentation The Mythical [Hu]man-Month, highlighting concerns about the names from the start, but leaving the focus on the lasting contributions in software engineering. The handout for students was presented in the form of a menu, complete with illustrations (including the tar pit image from the text), stylizing, and even some headings in French to fit with the chosen theme. While the menu theme aligns with the Restaurant Antoine menu provided at the beginning of Chapter 2 in The Mythical Man-Month: Essays on Software Engineering, which contains the translated quote “Good cooking takes time. If you are made to wait, it is to serve you better, and to please you.” [quoted in 4] and mentions the dangers of a cook turning up the heat, the student’s presentation takes it a step further. The two-page menu included some of the ‘offerings’ of the restaurant, a section devoted to meeting the chefs (a reprise on the familiar surgical team from the book), and a manager’s note explaining the student’s choice of an additional excerpt provided to students as well as his motivations for doing so.

The student’s in-class presentation likewise adapted other components of the text to the cooking theme. The provided rule of thumb for scheduling, presented in the original essay as a simple list of “⅓ planning, ⅙ coding, ¼ component test and early system test, ¼ system test and all components in hand” [4] can easily be presented as ingredients in a recipe for a productive project, with the recipe instructions providing additional guidance about the value of the components, the dangers of substitutions, and the importance of the proper order for assembling the final result. Though recipes are often adapted based upon the available ingredients and individual preferences, college students often have a sense, whether from cooking or chemistry, that deviating too far from the original can have undesirable consequences.

Some of the menu options included perfectly partitionable tasks and unpartitionable tasks. Just as menus often contain pictures of selected items, the graphs from the text displaying how Month and [Hu]Man depend upon each other are provided. The idea that these graphs are impacted by the amount of communication required can be conveyed through a pairing analogous to a wine pairing, indicating which tasks are well-suited for certain variants or adaptations. Since current menus frequently contain calorie counts, labels indicating that certain items are heart healthy, local favorites, and various other designations, students find these indicators and editorialization appropriate. However, it is important to also be mindful that, just like a menu, the recommendations are merely that, and the advice need not be heeded. It may be unwise to order more calories in a meal than are recommended for daily consumption, but restaurants will not prevent you from doing so. Similarly, it may be unwise to add additional workers to a late project, but that also happens, and may likewise seem like a good idea in the short term, but in the end be regretted.
Figure 1: The Meet the Chefs portion of the menu provides the roles for the software engineering team.

While Brooks’ consideration of the parallels between software engineering projects and restaurants are limited solely to the second essay, for many students, the participants in a surgical team are less familiar than the roles in a restaurant. While impressions of surgery come largely from the media for most college students, many are currently working in restaurants and have grown up interacting with some of the employees. Thus, the student’s presentation of Chapter 3 from [4] introduced the roles in a Meet the Chefs section, as shown in Figure 1. The head chef and sous chef are understood to work together, and be on the same team, but the head chef (or chief programmer, or surgeon) is the one on whom final responsibility for the project rests. Having two bussers instead of two secretaries still conveys the idea of multiple people working to allow other people to be focused on perhaps what they can do best, while ensuring that the clients have a good overall experience. And, just like bussers may be assigned to particular tables at a restaurant, they are often expected to pitch in where needed to help out their coworkers, as secretaries are frequently asked to do as well. The saucier (the toolsmith in the surgical model) is often responsible for working with other chefs to develop flavors that will then compliment the restaurant’s entrees, appetizers, and desserts, which aligns nicely with the responsibilities for constructing specialized utilities and macro libraries. Finally, bartenders are often known for their creative drinks made from a limited set of ingredients already in use, which parallels the `language lawyer’ who finds “neat and efficient ways to do difficult, obscure, and tricky things” [4].

DISCUSSION

Admittedly, roles within a restaurant have their limitation, just as the surgical team does, in determining the roles for a team for a large software engineering. However, they have some distinctions and potential advantages. The hierarchical nature is still present, as in the other categorizations, but that is in fact an inherent part of a large team project. Yet, while there are
default cultural assumptions in the United States about the genders of some members of the surgical team (wrong as they may be), there may be fewer gender assumptions about roles in restaurants. Additionally, there may not be the same assumptions about the ethnicity, socioeconomic status, or lack of upward mobility related to the roles within the restaurant. Furthermore, the restaurant model emphasizes the creation of a product, in this case a dish, as opposed to fixing or reconstructing something, as conveyed by the surgical model. Arguably, this draws more of a parallel to creating software with a team, where a tangible final product is the outcome of the collective efforts. Yet it still retains the notion that most large software engineering products are not created fully from scratch, relying on libraries and off-the-shelf code. Similarly, chefs make choices about what menu items are made from ingredients straight from the farm, and which incorporate packaged ingredients, such as dried pasta. More importantly, even if there are flaws in the parallels, presenting it as a potential model that is open to critique and refinement allows students to learn the software engineering lessons while also discussing limitations, assumptions, and restrictions.

While students may be hesitant to critique the ideas about roles in the industry that they plan to pursue yet may have limited experience, they may be more willing to discuss industries with which they are familiar. For example, the analog to the bartender in [4] can serve multiple head chefs. Does a bartender typically work in multiple establishments, or is there a breakdown in the analogy? If so, what are the limitations, and why? Students could then be asked to work, perhaps in groups, to convert the surgical team model to another structure with which they are familiar, perhaps even from the movies or a television show, and to highlight assumptions, stereotypes, and misconceptions.

The student’s presentation was well-received, and he clearly enjoyed sharing the information with his classmates. Time limitations prevented the class from doing any of the possible extensions discussed here, but multiple students would bring up Brooks’ law and other ideas throughout the course of the semester, suggesting the information was effectively presented and retained. Students likewise seemed optimistic that the gendered phrasing exists in part because it was prevalent at the time, and hopefully they can help change it. In fact, students commented on how attentiveness to diversity is a component of computer science courses at the college, but that there is always more to do in that regard.

It is expected that Brooks’ law and other ideas presented and developed in his The Mythical Man-Month: Essays on Software Engineering will continue to be influential in software engineering for years to come. We do not need to discard the valuable concepts or ignore the prudent advice about effectively conducting large projects, but we can also acknowledge and address the limitations of the presentation in terms of diversity and inclusivity. By allowing students to also re-cast the surgical team as organizations with which they are familiar or those that are representative to them of the diversity that they would like to see in their careers, we enable discussions about the challenges that are currently faced, increase awareness among students, and hopefully contribute to lasting change in the industry, academia, and beyond.
REFERENCES