Staying Consistent: Discipline-Specific Language Used in a Well-known AP CS A Curriculum

Jason L Weber
School of EECS
Oregon State University
Corvallis, USA
webejaso@oregonstate.edu

Jennifer Parham-Mocello
School of EECS
Oregon State University
Corvallis, USA
parhammj@oregonstate.edu

2023 IEEE Frontiers in Education Conference (FIE) | 979-8-3503-3642-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/FIE58773.2023.10343026

***Abstract*—The understanding of Discipline-Specific Language is an important competency for students of any field to begin mastering early in their studies, since it serves as a prerequisite for both the analysis of expert text and precise communication. Therefore, an introductory curriculum should pay careful attention to how it incorporates, defines, and uses Discipline-Specific Language.**

While close examination of the language used in instruction has been studied in applied linguistics for more than a decade, this idea has not yet been extensively applied to the language used in Computer Science (CS) instruction. A handful of authors in CS literature have suggested that Discipline-Specific Language be examined in CS education or software engineering education, but little published work has been done to justify the value of further research in this area.

In this exploratory research paper, we examine how a well-known Advanced Placement (AP) CS curriculum uses CS-specific language. We posed the following questions to guide our research:

1) **Is discipline-specific language identified and defined in the example curricula?**
2) **Is there temporal contiguity between:**
   a) **Occurrences of the same CS-specific word?**
   b) **The first occurrence of a CS-specific word and it's definition?**

We look at what words are used, when they are introduced, how they are defined, and the consistency of their application. To establish an initial set of CS-specific words, we analyzed the free-response questions of past AP CS-A exams and the official AP CS-A course and exam description document. We then recorded every location of each of these words within the curriculum so that we could examine the context around their usage.

While we found that technical CS-specific words were regularly defined and applied consistently, common (everyday) words with CS-specific context and connotations were either absent, ill-defined, or not applied consistently.

Our results suggest that the example AP CS-A curriculum could improve its introduction of Discipline-Specific Language, and we believe that this preliminary review of a well-known AP CS-A curriculum supports further investigation of other CS curricula and the impacts language has on students' learning and academic success.

***Index Terms*—Discipline-Specific Language; Computing Literacy; K-12 Education**

I. Introduction

In any field, it is important for both students and teachers to work towards or have mastery of writing, speaking, and understanding language that is a part of their discipline. In any discipline, there are words that are unique to the discipline or have different meanings when applied in a discipline-specific context. For example, in computer science (CS), a word like "byte" is unique to the discipline, but the word "translator" has a different connotation when used in CS. The precise use of these words is vital for students to be able to communicate with each other and experts.

While the need for general language competency and communication skills in CS has been recognized for some time [1], recent research questions posed by Diethelm and Goschler [2], [3] have opened up a new area of study focused on CS-specific language. Based on our experience teaching introduction to programming and CS, we believe that *introductory CS and programming learning material does not adequately define CS-specific language for students*, but we need to first establish what CS-specific language is and what it means to be adequately defined.

In this paper, we present the findings of our exploratory study focused on the usage of CS-specific language within a popular high school CS-A Advanced Placement (AP) curriculum named CSAwesome [4]. We established a list of CS-specific words and analyzed the incorporation, definition, and consistency of CS-specific language throughout the curriculum, shedding light on areas that may require improvement. By doing so, we aim to contribute to the ongoing discourse surrounding language and its role in optimizing students' learning experiences and academic achievements in the field of CS.

The subsequent sections of this paper provide a description of our research methods, an analysis of our research findings, discussion of their possible implications, and proposed recommendations for enhancing the usage of discipline-specific language in the CSAwesome curriculum. Through this research endeavor, we seek to provide valuable insights that can inform future efforts in developing language-sensitive didactics and improving students' language proficiency within the discipline of Computer Science.

II. Motivation and Related Work

In engineering education literature, the desire to create a list of discipline-specific words has gained popularity in recent years. Mudraya created the Student Engineering Corpus

with 1260 word families with words that were conjugates or compounds grouped together [5], and Ward created the intentionally-short Basic Engineering list of 299 words [6]. Since these kinds of lists help teachers "to set goals for their students' vocabulary learning" [7] and to modify students' reading materials [8], Ward and Todd believe that shorter lists are more valuable to teachers [6], [9].

Within the different approaches to list-making, authors defined different categorization schemes for words. Some researchers divided words into one of three categories: technical, sub-technical, or non-technical. Mudraya characterized technical words as having "the absence of exact synonyms, resistance to semantic change, and a very narrow range" [5], and other researchers [10]–[13] defined sub-technical words as having the same meaning across several disciplines or being neither non-technical (a general word used in everyday speech) nor technical. Similar to this definition of sub-technical words is Todd's definition of 'opaque words' as "words where the meaning required is not the usual meaning" [9].

In these exploratory studies, we used a similar, but unique categorization scheme that combined the above schemes. We defined *technical* CS-specific words as those which students would not be expected to have seen or heard outside of a computing/programming context and *common* CS-specific words as those which students probably have seen or heard outside of a computing/programming context that also have computing/programming-specific meaning(s). While our definition for technical CS-specific words is similar to Mudraya's [5], we note that our definition for common CS-specific words is more in line with Todd's definition for opaque words [9].

Although Beaubouef established the need for language competency in CS students [1], only recently have Diethelm and Goschler posed research questions to open up the research area of CS-specific language [2]. Since then, Diethelm and Goschler expanded upon their initial set of questions, now posing questions from three unique perspectives: the scientific perspective, the classroom perspective, and the curriculum perspective [3]. From the scientific perspective, an important question was posed: "Which terms are used in scientific publications and textbooks, and which of them could hinder or support learning due to narrow meaning of the word or parts of it?" [3]. Questions like this spurred others to adapt models of language learning to CS in an effort to begin development of "language-sensitive CS education" [14]. Additionally, those in adjacent fields like Software-Engineering Education have posed similar questions about discipline-specific language in their own disciplines [15].

To answer questions about what terms should be considered CS-specific language, one might incorrectly assume that we can look to dictionaries and glossaries from well known computing associations like the IEEE. However, as Gold-Veerkamp, Abke, and Diethelm point out, these terminology collections are both far too few and unsuitable to use as a baseline set of discipline-specific language for use in CS curriculum [15]. Additionally, even though new language has emerged in CS, the IEEE Standard Glossary of Software Engineering Terminology was last revised over 30 years ago [16]. Möhlmann and Sybre approached this problem by collecting an analysis corpus of German CSE web-pages and textbooks and a reference corpus of German newspaper articles (meant to represent general, non-CS language) to do difference analysis to extract high-frequency words unique to computer science [17]. Unfortunately, 41% of the words identified were not CS-specific, and further work needs to be done before a full dictionary can be created for CS education.

While a gap in research regarding discipline-specific language in CS has been clearly identified by researchers, little has been done to assess how a lack of language-sensitive didactics in CS affects student learning and academic success. To begin understanding how discipline-specific language manifests in CS curricula, we decided to quantitatively analyze a single curriculum in depth. Since Möhlmann and Sybre established the complexity of automating CS-specific language extraction [17], we instead started with a qualitative analysis of a small corpus of AP CS A exams, as well as the official AP CS A Course Description document [18]. Since these cumulative exams are meant to assess learning in introduction to programming, CS1-equivalent courses, they should use most or all of the CS-specific language covered in a CS1 course, and a course for the CS A AP exam should use the same the vocabulary on the exam.

III. Research Method

To address our hypothesis that introductory CS and programming learning material does not adequately define CS-specific language for students, we wanted to first establish what CS-specific language is and what it means to be adequately defined. Based on prior research, we believe that CS-specific language are divided into common and technical words, and a word is adequately defined when it includes a definition and an example when it is introduced, it is repeatedly used for re-enforced learning, and students feel they understand the word. Therefore, we conducted an exploratory study to answer the following research questions.

1) Is discipline-specific language identified and defined in the example curricula?
2) Is there temporal contiguity between:
    a) Occurrences of the same CS-specific word?
    b) The first occurrence of a CS-specific word and it's definition?

A. Selection of Curriculum

Since the high school CS-A AP exam [18], [19] is the only standard assessment for introductory CS/programming in the United States that we know about, we decided to use CS-A AP material to establish a list of some common and technical words and determine a baseline for how CS-specific language is defined in introductory to CS/programming. We selected a well-known high school CS-A AP curriculum called CSAwesome [20]. We selected CSAwesome for a few reasons. First, CSAwesome is officially endorsed by the College Board [21]. We believe that an officially endorsed curriculum is

more likely than non-endorsed curricula to use the same CS-specific language as the official CS-A AP exams. Additionally, CSAwesome has reported over 20,000 users as well as promising results as to its effectiveness [20]. Lastly, we had a K-12 teacher we have worked with who was currently using it for his CS-A AP course and spoke highly of it.

B. Collection of a Baseline Set of Discipline-Specific Words

In order to analyze CS-specific language use in the CSAwesome curriculum, we first established a set of CS-Specific words. To establish the list, we collected Free-Response Questions (FRQs) from past official AP CS A exams. We parsed through the FRQs from the 2022 exam [22] and compiled two lists of words. One list has words we categorize as *technical* CS-specific words, which we would not expect students to be familiar with prior to taking a CS course, and another list with words we categorize as *common* CS-specific words that would likely be familiar to students but in a different context or meaning outside of CS.

After making the two lists of words, we repeated this process for the FRQs from the 2021 and 2019 CS-A AP exams [18], [23], adding a small number of words to our lists. The 2020 exam was not publicly available and was skipped. Lastly, we reviewed the 2018 and 2017 CS-A AP exams [24], [25] and found no additional words to add. Because our identification of additional words seemed to plateau, we concluded our gathering of Discipline-Specific Language from the AP CS A exams. The collection of CS-specific words and their category (technical or common) were recorded in Tables I and II.

Note that while some words, like "row" and "column", may not seem like they should be considered CS-specific, since we might assume that many students encounter these in their math education; anecdotally, one author encountered first-year college students who were unfamiliar with the words "row" and "column". Therefore, we cannot make the assumption that students have the expected or desired mathematical preparation [26]. The proportion of technical words to common words can be found in Figure 1.

TABLE I
FREQUENCY OF TECHNICAL CS-SPECIFIC WORD USAGE BY UNIT

Word	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10
array	0	0	0	6	1	627	550	406	47	40
ArrayList	0	0	0	2	0	1	386	0	33	9
boolean	27	15	140	17	17	8	10	5	9	2
compose	0	0	0	0	0	0	0	0	0	0
constructor	0	101	0	3	180	10	19	56	80	0
implement	0	0	0	5	2	9	11	13	14	0
implementation	0	6	0	3	40	16	31	4	6	0
index	0	60	36	64	0	206	188	64	4	23
initialize	10	20	6	19	29	34	10	17	10	0
int	164	148	6	12	5	29	184	16	0	1
integer	45	67	6	12	5	29	184	16	0	1
Java	92	127	36	47	53	51	103	170	47	19
null	0	28	16	0	3	15	19	20	6	0
parameter	0	93	1	5	134	17	20	8	34	6
postcondition	0	0	0	2	20	4	2	7	0	0
precondition	0	0	0	11	55	14	11	2	0	1
static	62	78	52	88	104	70	97	57	41	54
subclass	0	5	0	0	0	0	0	0	102	0
substring	0	72	10	50	12	4	6	10	0	22
two-dimensional	0	0	0	0	0	0	1	49	0	0
variable	161	81	27	63	396	77	22	27	68	6

TABLE II
FREQUENCY OF COMMON CS-SPECIFIC WORD USAGE BY UNIT

Word	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10
accessible	0	0	0	0	6	0	0	0	0	0
call	0	70	7	15	78	24	45	13	74	82
class	113	490	85	79	640	146	149	169	686	6
code	265	256	289	287	275	261	340	241	175	55
column	2	5	3	4	9	1	0	170	0	0
declaration	6	0	0	1	18	13	12	6	21	6
definition	2	14	0	0	32	0	1	0	3	1
double	146	108	4	3	60	25	15	6	10	0
element	0	1	0	0	0	153	179	37	1	27
execution	3	2	2	6	10	2	7	0	3	4
helper	0	0	0	3	1	0	0	0	0	6
import	0	60	5	17	2	6	100	143	7	1
instance	0	34	2	1	255	18	12	7	58	0
method	33	324	58	103	579	175	180	141	323	146
new	0	179	32	11	92	60	257	32	108	1
object	8	261	43	7	214	69	118	27	153	2
private	0	25	5	4	208	34	41	35	65	2
program (noun)	22	19	20	9	30	16	14	13	10	2
public	150	208	129	153	397	198	226	171	297	66
represent (verb)	13	33	1	9	18	13	1	40	7	0
return	8	126	23	100	169	136	152	55	95	221
row	1	0	1	51	0	0	0	467	0	0
sequence	3	10	0	2	1	4	0	0	0	1
set (verb)	14	37	20	8	90	13	45	59	34	0
statement	17	10	210	41	11	6	13	14	5	1
string	124	487	204	236	260	139	289	131	206	61
value	130	182	79	83	144	243	144	207	12	28
void	72	101	52	58	112	61	75	44	71	10

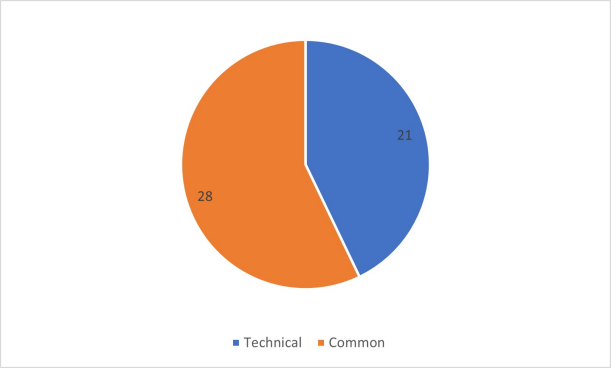

Fig. 1. Number of Technical vs Common Words

IV. RESULTS

After we established our lists of common and technical words found in the CS-A AP exam, we wanted to find out where and how many of the words appeared in the CSAwesome curriculum, if the words we identified were defined, and what the distance was between when words were introduced and when they were defined.

A. Identification of Discipline-Specific Language in CSAwesome

While CSAwesome provided an ebook and lesson plans associated with each chapter [4], we only looked at the ebook. Lesson plans probably contained the words too, but since it is up to individual teachers to decide whether or not to utilize the lesson plans in the classroom, we did not include the lesson plans in our analysis.

With the list of words collected in Tables I and II, we accessed all of the traditional "text" pages of the first ten units of the ebook and recorded the number of times each word was used. We did not record word usage from non-text pages, specifically those consisting solely of interactive multiple-choice questions or programming tasks. Although the table represents most words as purely lowercase, their associated counts include capitalized words. Additionally, common words were only counted if they were used in a discipline-specific context. The CSAwesome ebook contains sixteen different units, but only the first ten introduce and discuss computing concepts. The last six units serve as appendices with sample AP exam problems and stories from computing professionals. Since the last units were a collection of resources and did not discuss new computing topics, we excluded them from our analysis. Likewise, we excluded the first section of Unit 1 from our analysis because the first lesson starts at the second section, and we excluded the table of contents pages of each unit. The frequency of words used were recorded in Tables I and II.

Existing research has been unclear about how much exposure to new vocabulary is optimal for learning, with estimates ranging from 12 exposures [27] to 36 exposures [28]. Therefore, we identify words that appear under this range as *low*-frequency, words inclusively inside as *moderate*-frequency, and words exclusively above as *high*-frequency. In Tables I and II, rows with low-frequency and moderate-frequency words are colored red and yellow, respectively.

B. Definition of Discipline-Specific Language in CSAwesome

We also wanted to examine how language was introduced. Specifically, we recorded whether or not each CS-specific word we identified was defined, as a lack of proper introduction may suggest the need for greater precision in the introduction of CS-specific language. As shown in Figure 2, we found that while technical and common words had a similar number of definitions with 18 defined technical words and 19 defined common words, common words were left undefined far more often with 3 undefined technical words and 9 undefined common words. Interestingly, "postcondition" was the only moderate-frequency word that was defined, and no low-frequency word was defined.

This gap between common and technical definitions may suggest that there is a tendency to assume familiarity with words that appear outside of CS contexts, even when those words have unique connotations inside of CS. For example, "helper", while certainly a word English-speakers will be familiar with, was not defined even though it has unique

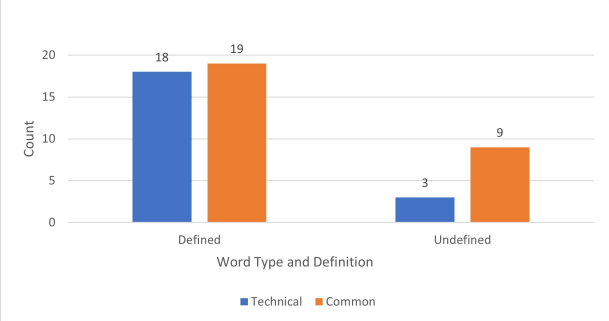

Fig. 2. Number of Undefined Words Versus Defined Words by Type

implications in a CS context. Mentioning a "helper" method usually refers to a private utility method which "helps" another method, often by repeating some repetitive task; this context may not be readily-apparent to a novice CS learner. Additionally, only a single moderate-frequency word and none of the low-frequency words being defined suggests a tendency to only define high-frequency words. This could be problematic though, as a low-frequency word like "helper" may still be used to explain and understand specific concepts or processes.

C. Temporal Contiguity of Word Usage

Next, we examined the temporal contiguity of CS-specific word usage by comparing the first occurrence of a word to the location of its definition and whether or not there are large gaps between occurrences of the word. To examine temporal contiguity in the CSAwesome ebook, we assumed a linear path from the start of the ebook to the end.

1) First Usage Versus First Definition: We chose to examine the distance between a CS-specific word's first usage and its definition. We wanted to see if words were being introduced and used to explain other topics before they were properly defined. Such occurrences could signal a need for curriculum designers and instructors to ensure that they are using language that is understood by their their students.

We found that 20 of the defined words were used before their definitions were provided, compared to 17 words which were defined in the same unit where they were first used. We subtracted the unit number of the first usage from the unit number of the definition for each defined word, giving us distances ranging from 0 (introduced in the same unit) to 7 (introduced 7 units prior to its definition). Figure 3 shows the distributions of distances by word type.

It is worth noting that 6 of the 20 words that were used before they were defined were done so in code examples. While some were out of necessity, like the Java keyword "public", we opted to include them since no special attention was given to their meaning until they were later defined and discussed.

2) Consistent Usage: We next wanted to look at whether or not CS-specific words were introduced and neglected for long periods of time, or if words were consistently repeated after initial use. Studies in both English and foreign language

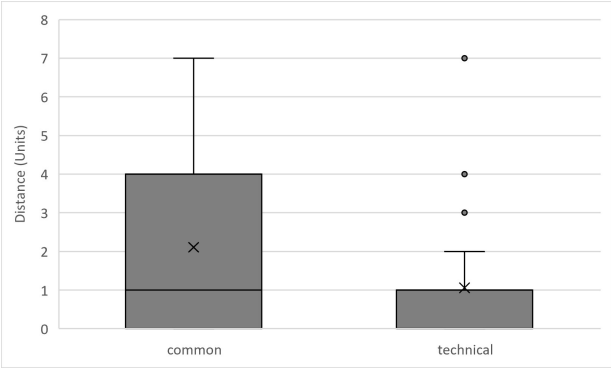

Fig. 3. Distances Between First Usage and Definition

acquisition have suggested that following the "Spacing Effect" (spreading learning out over time) is an effective way of supporting long-term retention [29], [30]. Inconsistent word usage may result in students forgetting a word that they are assumed to know in later units.

To analyze this, we selected all of the words which contained at least one unit with zero mentions between the units where the word was mentioned (see Table III). The results in Table III and Figure 4 show that 11 different words had 1 whole unit between uses (colored yellow) and 7 words had 2 or more units between occurrences (colored red).

V. Conclusions

From this exploratory research, we gained a few valuable insights. First, the CS-specific language used, regardless of its type: technical or common, might pose issues for student learning and academic success. Our observations of word frequency, definition, and temporal contiguity in Study 1 identified some interesting potential deficiencies in CS curriculum design. Common CS-specific words appeared to be utilized less effectively than technical words in all of our observations, particularly with respect to definitions.

However, there were some notable limitations with our results. While we explicitly defined and outlined our criteria for classifying words as CS-specific in Study 1, a person's familiarity with CS will likely produce lists that are slightly different. Additionally, there may be some disagreement with which CS-specific words are considered as technical or common based on a person's exposure to specific words outside of CS and their interpretation of the definitions for what categorizes a word as technical or common. For students with the expected mathematical background, words like "integer" and "variable" would already be familiar and therefore not considered technical, as opposed to students without this mathematics preparation who might be hearing these words for the first time in a programming/CS context. Despite these limitations however, we believe that our list and accompanying justifications are meaningful enough to support our initial exploration into the AP CSAwesome curriculum.

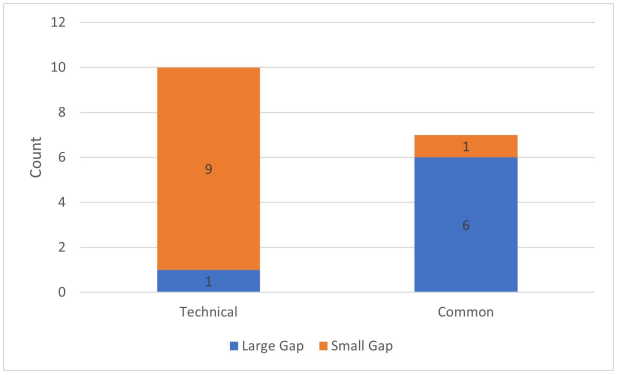

Fig. 4. Number of Gaps in Word Usage by Type

Despite the limitations of our study, we believe our findings from the CSAwesome curriculum are still valuable and provide insight into how the curriculum could improve its support of discipline-specific language competency for students. Based on our results, we provide the following recommendations:

1) Identifying the common words with CS-specific meanings or connotations within the curriculum and explicitly offering a definition,
2) Checking to make sure a CS-specific word has been thoroughly defined or discussed before using it to explain a new topic, and
3) Utilizing previously discussed words in code examples or review exercises so that there are no large gaps between usage of a discipline-specific word.

This preliminary study supports further investigation into other CS curricula, in addition to a deeper look into the effects that language-sensitive CS education or lack of language-sensitive CS education may have on students' learning and academic success. Based on our findings, we propose the following courses of further research:

1) How do different curricula define words? Do certain approaches lead to better student success?
2) How and why do different people classify CS-specific words differently?
3) Do students in courses that intentionally support discipline-specific language competency learn differently or have differing academic success than those in other courses?
4) The scope of discipline-specific language includes larger strokes of speech than single words. What larger, unique patterns of speech exist in a CS/programming context?

While this study is exploratory, we believe this initial research paves the way for more meaningful, rigorous research on discipline-specific language in CS education.

Acknowledgment

This work was partially supported by the National Science Foundation under the grant DRL-1923628.

TABLE III
INCONSISTENT FREQUENCY OF CS-SPECIFIC WORD USAGE

Word	Category	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10
definition	common	2	14	0	0	32	0	1	0	3	1
element	common	0	1	0	0	0	153	179	37	1	27
row	common	1	0	1	51	0	0	0	467	0	0
declaration	common	6	0	0	1	18	13	12	6	21	6
execution	common	3	2	2	6	10	2	7	0	3	4
helper	common	0	0	0	3	1	0	0	0	0	6
sequence	common	3	10	0	2	1	4	0	0	0	1
int	technical	164	148	6	12	5	29	184	16	0	1
integer	technical	45	67	6	12	5	29	184	16	0	1
ArrayList	technical	0	0	0	2	0	1	386	0	33	9
constructor	technical	0	101	0	3	180	10	19	56	80	0
implementation	technical	0	6	0	3	40	16	31	4	6	0
index	technical	0	60	36	64	0	206	188	64	4	23
null	technical	0	28	16	0	3	15	19	20	6	0
precondition	technical	0	0	0	11	55	14	11	2	0	1
subclass	technical	0	5	0	0	0	0	0	0	102	0
substring	technical	0	72	10	50	12	4	6	10	0	22

REFERENCES

[1] T. Beaubouef, "Why computer science students need language," *ACM SIGCSE Bulletin*, vol. 35, no. 4, pp. 51–54, Dec. 2003.

[2] I. Diethelm and J. Goschler, "On human language and terminology used for teaching and learning CS/informatics," in *Proceedings of the 9th Workshop in Primary and Secondary Computing Education*. Berlin Germany: ACM, Nov. 2014, pp. 122–123.

[3] ——, "Questions on Spoken Language and Terminology for Teaching Computer Science," in *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education*, ser. ITiCSE '15. New York, NY, USA: Association for Computing Machinery, Jun. 2015, pp. 21–26.

[4] B. Ericson, "AP CS A Java Course — AP CSAwesome," 2015. [Online]. Available: https://runestone.academy/ns/books/published/csawesome/index.html

[5] O. Mudraya, "Engineering English: A lexical frequency instructional model," *English for Specific Purposes*, vol. 25, no. 2, pp. 235–256, Jan. 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0889490605000360

[6] J. Ward, "A basic engineering English word list for less proficient foundation engineering undergraduates," *English for Specific Purposes*, vol. 28, no. 3, pp. 170–182, Jul. 2009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0889490609000246

[7] A. Coxhead, "The Academic Word List 10 Years On: Research and Teaching Implications," *TESOL Quarterly*, vol. 45, no. 2, pp. 355–362, 2011, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.5054/tq.2011.254528. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.5054/tq.2011.254528

[8] D. Gardner and M. Davies, "A New Academic Vocabulary List," *Applied Linguistics*, vol. 35, no. 3, pp. 305–327, Jul. 2014. [Online]. Available: https://doi.org/10.1093/applin/amt015

[9] R. Watson Todd, "An opaque engineering word list: Which words should a teacher focus on?" *English for specific purposes (New York, N.Y.)*, vol. 45, pp. 31–39, 2017, publisher: Elsevier Ltd.

[10] J. R. Cowan, "Lexical and Syntactic Research for the Design of EFL Reading Materials," *TESOL Quarterly*, vol. 8, no. 4, pp. 389–399, 1974, publisher: [Wiley, Teachers of English to Speakers of Other Languages, Inc. (TESOL)]. [Online]. Available: https://www.jstor.org/stable/3585470

[11] L. Trimble, *English for science and technology: a discourse approach*, ser. Cambridge language teaching library. Cambridge ; New York: Cambridge University Press, 1985.

[12] M. Baker, "Sub-Technical Vocabulary and the ESP Teacher: An Analysis of Some Rhetorical Items in Medical Journal Articles," *Reading in a Foreign Language*, vol. 4, no. 2, pp. 91–, 1988.

[13] J. Flowerdew, "Concordancing as a tool in course design," *System*, vol. 21, no. 2, pp. 231–244, May 1993. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0346251X9390044H

[14] F. Batur and J. Strobl, "Discipline-Specific Language Learning in a Mainstream Computer Science Classroom: Using a Genre-Based Approach," in *Proceedings of the 14th Workshop in Primary and Secondary Computing Education*, ser. WiPSCE'19. New York, NY, USA: Association for Computing Machinery, Oct. 2019, pp. 1–4.

[15] C. Gold-Veerkamp, J. Abke, and I. Diethelm, "A research approach to analyse and foster discipline-specific language competency in software engineering education," in *2016 IEEE Global Engineering Education Conference (EDUCON)*, Apr. 2016, pp. 652–659, iSSN: 2165-9567.

[16] "IEEE Standard Glossary of Software Engineering Terminology," *IEEE Std 610.12-1990*, pp. 1–84, Dec. 1990, conference Name: IEEE Std 610.12-1990.

[17] K. Möhlmann and J. Syrbe, "Term Extraction from German Computer Science Textbooks," in *Data Mining and Big Data*, ser. Lecture Notes in Computer Science, Y. Tan and Y. Shi, Eds. Cham: Springer International Publishing, 2016, pp. 219–226.

[18] "AP Computer Science A 2021 Free-Response Questions," 2021. [Online]. Available: https://apcentral.collegeboard.org/media/pdf/ap21-frq-computer-science-a.pdf

[19] C. Board, "AP Computer Science A Course and Exam Description, Effective Fall 2020."

[20] B. Ericson and B. Hoffman, "CSAwesome Java Curriculum," in *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 2*, ser. SIGCSE 2022. New York, NY, USA: Association for Computing Machinery, Mar. 2022, p. 1026.

[21] C. Board, "Adopt Ready-to-Use Curricula – AP Central | College Board," 2023. [Online]. Available: https://apcentral.collegeboard.org/courses/ap-computer-science-a/classroom-resources/curricula-pedagogical-support

[22] ——, "AP Computer Science A 2022 Free-Response Questions," 2022. [Online]. Available: https://apcentral.collegeboard.org/media/pdf/ap22-frq-computer-science-a.pdf

[23] ——, "AP Computer Science A 2019 Free-Response Questions," 2019. [Online]. Available: https://apcentral.collegeboard.org/media/pdf/ap19-frq-computer-science-a.pdf

[24] ——, "AP Computer Science A 2018 Free-Response Questions," 2018. [Online]. Available: https://secure-media.collegeboard.org/apc/ap18-frq-computer-science-a.pdf

[25] ——, "AP Computer Science A 2017 Free-Response Questions," 2017. [Online]. Available: https://apcentral.collegeboard.org/media/pdf/ap-computer-science-a-frq-2017.pdf

[26] K. Morati, "Examining the Mathematical Preparedness of First Year College Students Entering College Directly from Traditional High Schools in Connecticut," *Research & Teaching in Developmental Education*, vol. 25, no. 1, pp. 30–44, 2008.

[27] M. G. McKeown, I. L. Beck, R. C. Omanson, and M. T. Pople, "Some Effects of the Nature and Frequency of Vocabulary Instruction on the Knowledge and Use of Words," *Reading Research Quarterly*, vol. 20,

no. 5, pp. 522–535, 1985, publisher: [Wiley, International Reading Association].

[28] H. L. Storkel, K. Voelmle, V. Fierro, K. Flake, K. K. Fleming, and R. S. Romine, "Interactive Book Reading to Accelerate Word Learning by Kindergarten Children With Specific Language Impairment: Identifying an Adequate Intensity and Variation in Treatment Response," *Language, Speech, and Hearing Services in Schools*, vol. 48, no. 1, pp. 16–30, Jan. 2017, publisher: American Speech-Language-Hearing Association.

[29] P. I. Pavlik Jr. and J. R. Anderson, "Practice and Forgetting Effects on Vocabulary Memory: An Activation-Based Model of the Spacing Effect," *Cognitive Science*, vol. 29, no. 4, pp. 559–586, 2005.

[30] H. S. Sobel, N. J. Cepeda, and I. V. Kapler, "Spacing effects in real-world classroom vocabulary learning," *Applied Cognitive Psychology*, vol. 25, no. 5, pp. 763–767, 2011.