Bull, G. (2020). Integrating language arts and computational thinking: A reflection on the importance of gossip. *Contemporary Issues in Technology and Teacher Education, 20*(3), 555-561.

Integrating Language Arts and Computational Thinking: A Reflection on the Importance of Gossip

Glen Bull
University of Virginia

Read the Goldenberg chapter and related articles:
Gossip and Other Life Sentences: A Simple Grammar
Reflections on "Exploring Language With Logo"
Exploring Language: The Journey from Logo to Snap!
Commentary: Thoughts on "Exploring Language: The Journey from Logo to Snap!"

According to Cynthia Solomon, Seymour Papert went to Europe in the summer of 1966 and returned with the initial specifications for the computing language Logo (Personal Communication, 2018). Logo is notable as the first computing language designed specifically for children. Solomon and Papert collaborated with Wally Feurzeig to develop an initial implementation of the computing language, which they began testing with children in local schools the following year.

At the time, Papert was codirector of the MIT Artificial Intelligence Laboratory. Artificial intelligence (AI) is the science of creating computer programs that can replicate the performance of humans. An important aspect of AI is interpretation of human language. Therefore, not surprisingly, an important aspect of Logo is its capability to process words and sentences. Feurzeig originated the name of the language, Logo, the Greek word for *word*.

Papert summarized this work in *Mindstorms: Children, Computers, and Powerful Ideas,* published in 1980. Papert introduced the term "computational thinking" in *Mindstorms*. He also described the use of Logo by middle school students to generate "computer poetry." The students created syntactic structures that generated sentences in the process of exploring the way in which language works.

Background

Like many others, I was influenced by the vision that Papert outlined in *Mindstorms*. Texas Instruments donated 10 TI 99/4 microcomputers with Logo that I used to offer an initial course, Teaching With Logo, at the University of Virginia in fall 1980. The TI microcomputer had a fraction of the power of today's personal computers. Programs were saved on audio cassette tapes. However, some of the features such as sprites foreshadowed similar features found in contemporary block programming languages such as Snap! and Scratch.

A physics student, Tom Lough, took the course as an elective. Tom was so impressed by the potential of Logo that he entered the doctoral program as my advisee. He also founded and published the *National Logo Exchange* (NLX) newsletter. A colleague, Steve Tipps, and I supported Tom in this endeavor by serving as associate editors. In 1983 Tom proposed a national Logo conference. Seymour Papert learned about our plans and was kind enough to propose that we jointly sponsor the conference.

We flew to MIT to participate in the conference planning process and were excited to meet a number of the pioneers in Logo in person. Paul Goldenberg was one of those individuals. His work has continued to influence my work with Logo and with its successors such as Snap! and Scratch. He was a scientist at Bolt, Beranek, and Newman in the late 1970s, where he met Wally Feurzeig. He then served as director of the computer science department at Lincoln-Sudbury High School in the early 1980s.

Language and Computers

Goldenberg and Feurzeig published *Exploring Language With Logo* in 1987. This work has held a prominent place above my desk since its publication. Therefore, we are very pleased to be able to make its introductory chapter, "Gossip and Other Life Sentences – A Simple Grammar," available as a seminal work to readers of the *CITE Journal*.

Exploring Language With Logo explores the structure, function, and history of language using functions made possible by computers. It lies at the intersection of linguistic thinking and computational thinking.

Goldenberg observes that for computers to be useful to students of language they must "enable the expression of linguistic structures in a fairly natural way," noting that Logo (as its name implies) was designed with the needs of language manipulation in mind.

One of the crucial elements in a computing language designed for this purpose is the ability to process lists of words. Logo is based on a language, LISP, developed to support work in artificial intelligence. In order for digital assistants like Siri and Alexa to respond to our commands, they must process the lists of words that comprise our sentences. The programming language LISP was designed for this purpose. LISP is an acronym that stands for a "List Processing" language. Logo builds on the foundation of LISP reconfigured with a notation that is more accessible to children.

Mitch Resnick studied with Papert in the 1980s, developing extended versions of Logo that controlled motors, gears, and sensors. This work led to today's LEGO Mindstorms robotics systems. Resnick subsequently became the Papert Professor of Learning Research at MIT. In that role, he developed a modern-day programming language, Scratch, that incorporated lessons learned from work with Logo.

Scratch incorporates graphical user interface elements that did not exist when Logo was conceived. A key goal was to make Scratch accessible to young children. By design, Scratch did not incorporate some of the more advanced features of Logo, such as recursion. Scratch proved to be successful and influential, with more than 50 million users.

The educational programming language Snap! was subsequently developed at the University of California at Berkeley by Brian Harvey and Jens Moenig. The acknowledgements page for Snap! notes that it was inspired and influenced by Scratch. However, Snap! also includes advanced features not available in Scratch.

Snap! is tightly integrated with an undergraduate computing course at Berkeley, the Beauty and Joy of Computing(BJC). The version of this course developed at EDC (https://bjc.edc.org/) is approved for Advanced Placement credit by the College Board, and has been adopted by some of the largest school systems in the United States (Education Development Center, 2018).

Snap! encompasses the list processing features of Logo described in *Exploring Language With Logo*. It can be used to explore language in the same way as Logo. However, because it is a block-based coding system with a web-based graphical user interface, students can devote more of their time to exploration of language and less time dealing with programming trivia that does not directly advance language learning.

The companion piece, "Exploring Language: The Journey from Logo to Snap!", also published in this issue, provides a roadmap for implementing these activities today. It provides a guide for the way in which the original Logo activities can be implemented today for those who wish to explore these activities.

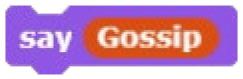
Gossip Today

In Logo the output of the Gossip procedure could be printed on the screen using the Print command:

```
? Print Gossip
  Dale talks a mile a minute
```

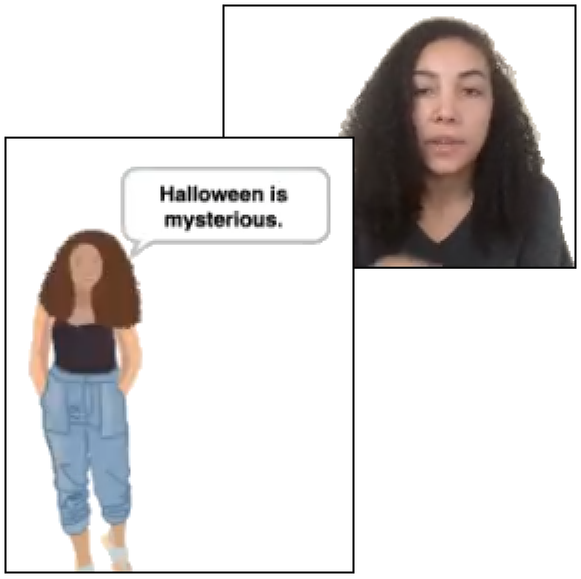
The graphical capabilities of Snap!make it possible to create avatars that gossip and interact in other ways. The Say code block in Snap! can be combined with the Gossip code block to enable avatars to say gossip (Figure 1).

Figure 1
Say code block combined with Gossip code block in Snap!



In Figure 2, Alexis (on the right) has created an avatar (on the left) in Snap! and used the Say code block to enable the avatar to express themselves.

Figure 2
Avatars in Snap!



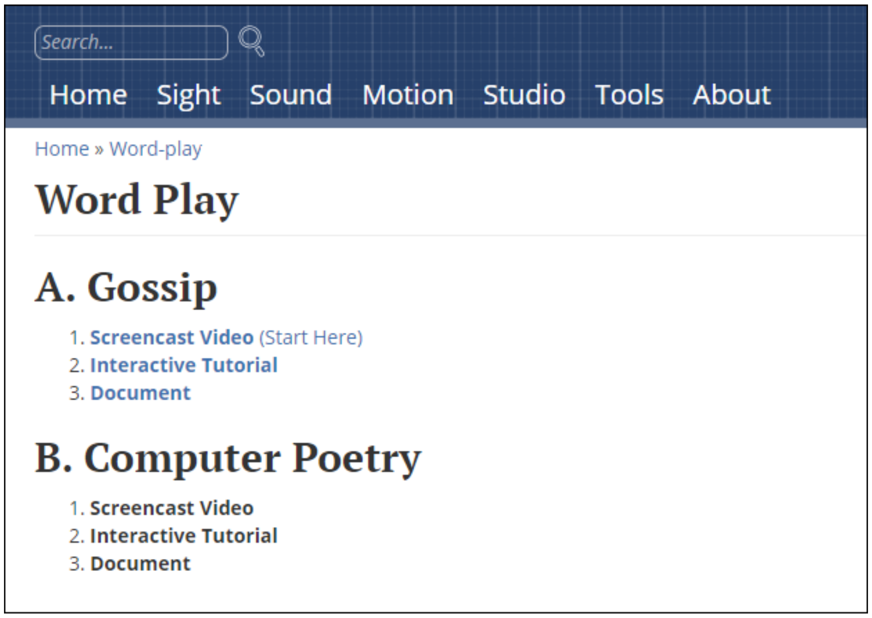
The capabilities for bringing characters on the screen to life in this manner offer a rich universe of possibilities.

Technology and Teacher Education

The *CITE Journal* is jointly sponsored by five teacher educator associations, including the Society for Information Technology and Teacher Education (SITE) and the English Language Arts Teacher Educators (ELATE) of the National Council of Teachers of English (NCTE). The preface to the English Language Arts section of the journal notes, Because ELATE members are engaged in the preparation, support and continuing education of teachers of the English language arts and literacy, they understand the need to explore English education with and through technology.

In order to support and advance this endeavor, the *CITE Journal* has also developed and sponsored an illustration of the activities described here. These resources include an interactive tutorial and an instructional video illustrating the way in which these activities might be implemented (Figure 3).

Figure 3
Screenshot of Word Play site.



Word Play

A. Gossip

1. Screencast Video (Start Here)
2. Interactive Tutorial
3. Document

B. Computer Poetry

1. Screencast Video
2. Interactive Tutorial
3. Document

These illustrative examples are available at www.maketolearn.org/word-play.

Time in the teacher education curriculum is always constrained. The initial illustration is designed for implementation in a single class period. It can be implemented either as a class activity or as a supplementary assignment using the online resources.

Summary

The capabilities of computers have changed dramatically in the three decades since *Exploring Language With Logo* was published. The grammatical structure of language that is investigated in this work, however, has not changed at all.

Foundational concepts regarding synergies between linguistic thinking and computational thinking were established in this work. Modern educational programming languages such as Snap! enable the concepts developed in this work to be implemented in an even more natural way.

We are, therefore, pleased to have the opportunity to republish the opening chapter of *Learning Language With Logo* as a seminal work in the *CITE Journal*. This work is important from a historical perspective, because it provides insight into the origins of related modern-day work that it inspired. However, it is also highly relevant as a road map to ways in which we might incorporate computers into instruction in a humane and effective way.

Acknowledgements

This material is based in part upon work supported by National Science Foundation STEM-C Grant No. 159457. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors.

References

Bull, G., Garofalo, J. & Nguyen, N.R. (2020) Thinking about computational thinking, *Journal of Digital Learning in Teacher Education*, *36*(1), 6-18. https://doi.org/10.1080/21532974.2019.1694381

Education Development Center. (2018). *Bringing a rigorous computer science principles course to the largest school system in the United States*. Annual Project Report submitted to the National Science Foundation.

Papert, S. (1970). Teaching children thinking. *Proceedings of the World Conference on Computer Education*. International Federation for Information Processing.

Papert, S. (1971). *Teaching children thinking* (MIT Artificial Intelligence Laboratory Memo No. 2247). http://hdl.handle.net/1721.1/5835.

Papert, S. (1980). *Mindstorms, children, computers, and powerful ideas*. Basic Books.

Papert, S. (2005). You can't think about thinking without thinking about thinking about something. *Contemporary Issues in Technology and Teacher Education*, 5(3/4). https://www.citejournal.org/volume-5/issue-3-05/seminal-articles/you-cant-think-about-thinking-without-thinking-about-thinking-Q7 about-something

Resnick, M., Silverman, B., Kafai, Y., Maloney, J., Monroy-Hernandez, A., Rusk, N., & Silver, J. (2009). Scratch: Programming for all. *Communications of the ACM, 52*(11), 60–67. https://dl.acm.org/doi/10.1145/1592761.1592779

Weintrop, D., & Wilensky, U. (2017, October). Comparing block-based and text-based programming in high school computer science classrooms. *ACM Transactions on Computer Education, 18*(1), Article 3.

Contemporary Issues in Technology and Teacher Education is an online journal. All text, tables, and figures in the print version of this article are exact representations of the original. However, the original article may also include video and audio files, which can be accessed online at http://www.citejournal.org