Duane Buck

a. Professional Preparation.

Institution	Major	Degree	Year
Lawrence Technical University, Southfield, MI	Physics, Math	B.S.	1975
Wayne State University, Detroit, MI	Computer Science	M.A.	1977
Ohio State University, Columbus, OH	Computer Science	Ph.D.	1993

b. Appointments.

2012-	Professor, Department of Mathematical Sciences, Otterbein University, Westerville, OH
2002-2012	Associate Professor, Department of Mathematical Sciences, Otterbein University, Westerville, OH
1993-2002	Assistant Professor, Department of Mathematical Sciences, Otterbein University, Westerville, OH
1991-1993	Lecturer, Department of Mathematical Sciences, Otterbein University, Westerville, OH
1987-1991	Graduate Teaching Associate, Computer and Information Science, Ohio State University, Columbus Ohio
1985-1987	Programmer/Analyst, Ford Motor Company, Dearborn, Michigan
1980-1985	Lecturer, adjunct faculty, Lawrence Technical University, Southfield, Michigan
1979-1985	Systems Analyst, General Motors, Warren, Michigan
1977-1979	Senior Systems Analyst, American Natural Service Company, Detroit, Michigan

c. Publications. PUBLICATIONS RELATED TO EDUCATION

[1] Duane Buck and David J. Stucki. Design early considered harmful: Graduated exposure to complexity and structure based on levels of cognitive development. In *Proceedings of the thirty-first SIGCSE technical symposium on Computer science education*, SIGCSE '00, pages 75–79, New York, NY, USA, 2000. ACM. ISBN 1-58113-213-1. doi: 10.1145/330908.331817. URL http://doi.acm.org/10.1145/330908.331817.

[2] Duane Buck and David J. Stucki. JKarelRobot: A case study in supporting levels of cognitive development in the computer science curriculum. In *Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education*, SIGCSE '01, pages 16–20, New York, NY, USA, 2001. ACM. ISBN 1-58113-329-4. doi: 10.1145/364447.364529. URL http://doi.acm.org/10.1145/364447.364529.

[3] Duane Buck and David J. Stucki. The hidden injuries of overloading 'ADT'. In *Proceedings of the 40th ACM technical symposium on Computer science education*, SIGCSE '09, pages 256–259, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-183-5. doi: 10.1145/1508865.1508958. URL http://doi.acm.org/10.1145/1508865.1508958.

[4] Duane Buck. GUIGraph: Editing live object diagrams for GUI generation enables new pedagogy in CS1/2. In *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education*, ITiCSE '11, pages 193–197, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0697-3. doi: 10.1145/1999747.1999803. URL http://-doi.acm.org/10.1145/1999747.1999803.

[5] Duane Buck and Ira Diethelm. 2013. Authentic object modeling in the early computer science curriculum using Objektgraph. In Proceedings of the 18th ACM conference on Innovation and technology in computer science education (ITiCSE '13). ACM, New York, NY, USA, 363-363. DOI=10.1145/2462476.2466532 http://doi.acm.org/10.1145/2462476.2466532

[6] Duane Buck. 2013. First, do no harm: a curricular approach to reliability. In Proceedings of the 18th ACM conference on Innovation and technology in computer science education (ITiCSE '13). ACM, New York, NY, USA, 319-319. DOI=10.1145/2462476.2466515 http://doi.acm.org/10.1145/2462476.2466515

[7] Duane Buck, Ira Diethelm, and Stephen Sheneman. 2013. Objektgraph: why code when MVC applications can be generated with UML-based diagrams?. In Proceedings of the 2013 companion publication for conference on Systems, programming, & applications: software for humanity (SPLASH '13). ACM, New York, NY, USA, 25-26. DOI=10.1145/2508075.2514576 http://doi.acm.org/10.1145/2508075.2514576

OTHER PUBLICATIONS

[8] Duane Buck and Mukesh Singhal. An analytic study of caching in computer systems. *J. Parallel Distrib. Comput.*, 32 (2): 205–214, February 1996. ISSN 0743-7315. doi: 10.1006/jpdc.1996.0014. URL http://dx.doi.org/10.1006/jpdc.1996.0014.

d. Activities related to pedagogical software.

1. My papers report on only two of the pedagogical software tools I have developed. During the autumn of 2001 I visited the BlueJ group at Monash University in Melborne, Australia for several weeks and worked with them to enhance their plug-in architecture, then implemented a novel object visualization plugin which for a time was in the standard BlueJ distribution.
2. I ported the object visualization plugin to JGrasp and enhanced it (again I worked with the group to enhance their plug-in interface). They have since also developed visualization tools for specific data structures. The object visualization is an advance that I will publish about in the future.
3. I have interacted frequently with the JGrasp group and have had a role in shaping many of its pedagogical features. One idea of mine they specifically acknowledged in an ITiCSE'08 paper is the capability of switching the context of visualizing an object between private, package, protected, and public access.
4. I developed a web-based curriculum for the Otterbein computer science introductory course in Pascal in the late nineties. It was praised as an outstanding course by our Education chair, after her young teenaged son took it independently with a friend. She placed a letter of commendation in my promotion portfolio. Although I did not actively promote and distribute the course, through anecdotal reports and Internet searches, I have found it is still used by many schools.
5. An early version of Karel The Robot was developed for the above course and written in Object Pascal. A surprising number of users continue to use the Windows only version which is used in the course, even though it has not been distributed by me for about 12 years.
6. I later developed a web-based course for the Java language when we moved the curriculum to that language. That course became the basis for our inside/out curriculum approach, which was presented in the paper "Design Early Considered Harmful." The new web-based course was required in our curriculum for more than ten years, until we switched to semesters. We were among the first to advocate the approach we used in our curriculum, which later became known as the methods-early approach, and is still used widely.
7. The JKarelRobot pedagogical software tool was developed for the above course. It is programmable in Pascal, Java, and Lisp syntaxes. (Lisp was not a good choice; today I would choose Python). The languages share a common runtime system because each compiles an executable object structure. In addition to the three programming languages, the interface and language keywords are switchable between three natural languages; the software is used worldwide. I wrote the new version in Java so that it could run on more platforms, and added many pedagogical features. It has several built-in exercises, one of which is unique. The student constructs a flowchart for an already compiled program. They are given all of the nodes, and it is their assignment to add the flow-lines. Their accuracy rate is monitored, and students willingly repeat the exercise to improve their score. It has proven to be a good assignment for a review in later courses also. Writing the flowchart for a piece of code requires a deep level of understanding, and the exercise helps the student acquire it. During his

keynote address at SIGCSE'06, Rich Pattis showed a slide of me with a screenshot of JKarelRobot and said it was "by far" the best current implementation of his original Karel, among the large number developed over the years. An ITiCSE'07 working group surveyed the available literature on the teaching of introductory programming. In their published report, they say that of the 99 tools nominated, they reviewed 60, and they selected JKarelRobot as one of the 14 best for their "Guide to the Literature."

8. My GUIGraph pedagogical tool enables the student to build "live" object structures with a graph editor, and has proven to build and reinforce object-oriented thinking, even though its original goal was to support methods-early curricula by supplying a context in which to write methods. The JGrasp group has agreed to work with me on integrating GUIGraph into JGrasp. The pedagogical advantages of building "live" object structures that are actually graphs that can be saved and edited led me to this line of research and the current proposal.

e. Collaborators & other affiliations.

1. **Collaborators.**
   David J. Stucki Otterbein University
2. **Graduate advisor.**
   Mukesh Singhal University of Kentucky
3. **Thesis advising.**
   None.