Ritter, F. E., Qin, M., MacDougall, K., & Chae, C. (2023). Lessons from a broad survey of tutoring tools: It's a big world out there. *Interactive Learning Environments, 31*(4), 2444-2451.

For resubmission to: *Interactive Learning Environments.* NILE-2019-0363.R1

Lessons from Surveying Tutoring Tools: It's a Big World out There

Frank E. Ritter[1]
(frank.ritter @psu.edu)

Michael "Q" Qin[2]
(michael.qin@navy.mil)

Korey MacDougall[1]
(korey@koreymacdougall.com)

Chungil Chae[1]
(chadchae@gmail.com)

[1] College of IST, Pennsylvania State University

[2] ONR

Author Note

Word count: ~4,670, inclusive of tables, references, figure captions, footnotes, and endnotes.
4 Feb 2020 — surveying tutoring toolsV28.doc

Keywords: tutoring systems, design space, stakeholders, ontology of system features

Acknowledgements

We thank internal reviewers for criticisms of an earlier draft, which helped greatly in improving it, as well as Chris Garrison, Laura Kurland, Mark J. W. Lee, Ashley McDermott, Jacob Oury, David Schwartz, Yvette Tenney, and Kurt VanLehn (who provided valuable counsel), and three anonymous reviewers. Preparation of this paper was supported by ONR N00014-15-1-2275, N00014-18-C-7015, and AFRL FA8650-16- C-6680.

Abstract

We created a list of more than 140 tools that can be used to create tutoring systems. This includes a wide range of systems, from high-level tools providing complete tutoring systems to low-level tools for preparing instructional materials for inclusion in tutors, such as video editors. Based on this list, we present a preliminary ontology of system dimensions that can serve as a base for a comprehensive review or in building a system to navigate this problem space more effectively. From this survey we also pull out lessons that we think any similar survey of tutoring systems would find, that indicate: (a) such broad surveys or reviews of tutoring systems will always be incorrect because they are incomplete, (b) such reviews are not worth pursing indefinitely (because they will never be complete), and, however, (c) comprehensive reviews, or a "living platform" like a wiki, may be immensely useful for stakeholders. The present work, as the result of surveying the variety of available tools, provides several insights about tutoring systems: (a) the scope of digital tutoring software includes a great range of dissimilar tutoring architectures, as well as an even wider range of supplementary tools; (b) tutoring systems vary on many dimensions (we provide an initial set); and (c) any individual tutoring system is unlikely to gain widespread adoption for all use cases because of the variety of needs of learners and other stakeholders. This suggests that a wide variety of tutoring architectures are needed. We argue that a clear sense of usage and the wide scope we found will be help tailor development, funding, and acquisition decisions.

Introduction

It would be useful to understand the range of tutoring tools. A review in terms of scope of application and the variability of features that these systems exhibit could help ground future work in tutoring. There have been previous, more formal reviews that have been helpful (Grubisic, Stankov, & Zitko, 2015; Murray, 1999; Psotka, Massey, & Mutter, 1988; Sleeman & Brown, 1982; VanLehn, 2011; Wenger, 1987). We have also undertaken a shorter review, internally and informally, as a step related to creating a tutoring architecture (Ritter et al., 2013). This paper arises in part to address the difficulties involved in conducting that review. Keeping a review current can also be challenging in part due to the rapidity with which new systems are produced and released, and in part to the wide range of use-cases to which these systems are applied.

As tutoring systems are used more broadly and regularly, the number of dimensions upon which tutoring systems vary is increasing as well. For example, one key dimension on which tutoring systems vary is the context of use: teaching and training are important in a wide range of settings, from primary through tertiary education, to military and sport training, to specialized vocational training. Each setting imposes a different set of demands on teachers and learners. There are thus many dimensions along which tutoring systems differ, several of which we discuss below, and this is reflected by a complex landscape of tutoring tools.

This report is a preliminary attempt to classify the dimensions on which tutoring systems cohere or differ and to make explicit the value and difficulty of assembling such an ontology. We identify and refer to four primary user groups for tutoring systems: learners, who learn from tutoring architectures; instructional designers, who build tutors (using modules, lessons, courses, etc.) within tutoring architectures; architects, who build tutoring architectures; and researchers

who build and use tutors to study how to build systems or build the systems to gather learning data. The latter three are potential readers of this paper.

An Informal, Broad Survey of Tutoring Systems

To understand the space of tools related to tutoring, one of us (MQQ) created an informal survey of all the tutoring tools and their types that he could find in several months using a sample of convenience. These tools are shown in Figure 1. This figure has 135 tools and is 4.3 m (14 feet) wide when printed at 10-point font. That is to say, it is a large figure, and there are several types of tools and numerous tutoring systems. The types (subconcepts) appear merely as triangles at this resolution. The categories were developed using grounded theory (Adams, Lunt, & Cairns, 2008; Muller, 2014) by examining the systems. The figure is included to give the reader a visual representation of the complexity of the space; we cannot reproduce it completely or clearly here due to its size, but include the figure to show the general shape and size of the taxonomy. A scrollable and zoomable version is available online (http://acs.ist.psu.edu/papers/Authoring_Tool_Tree_2.0.pdf). Table 1 presents the types of systems and their structures in a more readable form.

While this set is not a complete list of available tutoring tools our conclusions would be supported by any similar list because the set of systems is so broad and the details and even the recency of the systems do not quite matter—it does not matter to the conclusions if a particular tutor (such as Debuggy, VanLehn, 1982) or systems such as the Plato system (Dear, 2017; Smith & Sherwood, 1976), ASSISTments (Singh et al., 2011), or GiFT (Sottilare, Graesser, Lester, & Baker, 2017) are included or not. Readers familiar with these topics can likely think of numerous systems to add to each category.

The tools in Table 1 cover a wide range of system types and sizes. The highest-level systems in the ontology are self-contained authoring tools and content management systems (CMSs) that provide turnkey tutoring support for universities and are capable of providing tutoring solutions for a wide range of tutoring needs. Middle-level systems include those used to create specific types of tutors (i.e., those targeted at a more restricted set of use cases) or research versions of intelligent tutoring systems that can be applied more widely. Lower-level systems provide support for creating and modifying instructional material, such as text or video editing software. Note that some of the lower-level systems here may be large or complex software packages in their own right, such as Adobe Acrobat for editing PDFs, or drawing tools such as Adobe Photoshop and OmniGraffle for preparing diagrams for inclusion in tutors. We refer to them here as “lower-level” in the sense that they do not provide complex tutoring functionality, though they are often used as part of the tool-chain in larger or more comprehensive tutoring systems.

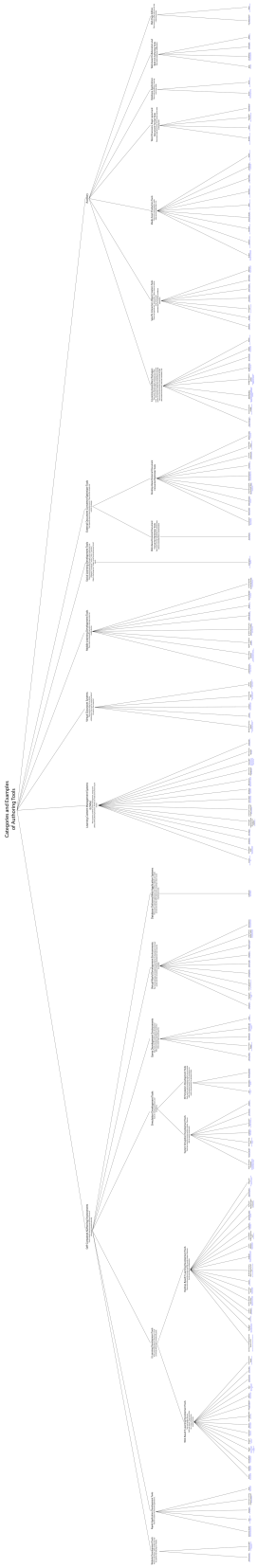

Figure 1. Task analysis-based survey of tools related to tutoring.

Full figure available at http://acs.ist.psu.edu/papers/Authoring_Tool_Tree_2.0.pdf

Table 1. Ontology of tutoring tools used in Figure 1. The number and names of tools found are in (). References for tools in online figure.

1. Self-contained authoring environments
   a. Website development tools (2: Dreamweaver, Expression Web)
   b. Rapid application development tools (5: Expression Studio, Flex, Flypaper, Impression Learning Content Framework, Flash)
   c. E-Learning Development Tools
      i. Web-Based E-Learning Development Tools (14: Unison, RapidL, SmartBuilder, ROCCE, MyUdutu, Luminosity, Podium, Ilias SCORM Editor, Mzinga Publisher, Mohive, Kreus, Content Point, Force Ten, Course Avenue Studio)
      ii. Desktop-Based E-Learning Development Tools (16: Learning Content Development System (LCDS), GLO Maker, eXe, Content Publisher, Desktop Development Suite, Xerte, Trainer, Multimedia Learning Object Authoring Tool, Toolbook, Lectora Publisher, CourseMaker Studio Authoring Suite, Captivate, E-Learning Suite, Impression Learning Content Framework, Expert Author, MOS Solo)
   d. Simulation Development Tools
      i. System Simulation Development Tools (7: Firefly Simulation Developer, Camtasia Studio, Assima Training Suite, Captivate, Capture Point, STT Trainer, SoftSim)
      ii. 3D Simulation Development Tools (3: ESP, Flex Builder, Thinking Worlds)
   e. Game Development Environments (5: Game Studio, Torque Game Engine, Visual3D.net, Truevision 3D, Unity)
   f. Virtual World Development Environments (9: 3Dxplorer, OpenQwaq, Project Wonderland, Protosphere, Second Life, Teleplace, Vastpark Creator, Vizard Virtual Reality Toolkit, World Visions)
   g. Database-Delivered Web Application Systems (1, ColdFusion)
2. Learning Content Management Systems (13: A Tutor, dominKnow LCMS, Evolution, First Align, GreeLight Learning Content Management System, Impression Learning Content Framework, Learn eXact, Mindflash, MOS Chorus, Saba Content Management, Sana LCMS, SAP Enterprise Learning, TotalLCMS)
3. Virtual Classroom Systems (5: Blackboard Learning System, Centra, Connect, Social Learning Suite, Wimba Classroom)
4. Mobile Learning Development Tools (8: Chalk Pushcast, eXact Learning Solutions, iQpakk (LCMS for mobile), On Point Learning and Performance System, Mobile Study, Mobl 21, Hot Lava Mobile, SumTotal Mobile e-Learning Solution)
5. Social Learning Development Tools (1: Scate Ignite)
6. External Document Converter/Optimizer Tools
   a. Web-Based External Document Converter/Optimizer Tools (1: AuthorPoint)
   b. Desktop-Based External Document Converter/Optimizer Tools (9: Content Point, Elicitus Suite, Helius Presenter, Learning Essentials for Microsoft Office, Metamorphosis, Presenter, PPT Flash Professional, SmartBuilder Author, Wimba Create)
7. Auxiliary
   a. E-Learning Assemblers/Packagers (8: eXact Packager, Framework for Scorm, RELOAD Editor, SCORM Developer's Toolkit, SCORM Driver, Scormworks, THESIS, Trident)
   b. Specific Interaction Object Creation Tools (7: Diploma, Engage, Hot Potatoes, Perception, Quiz Creator, Quizmaker, Rapivity)
   c. Media Asset Production Tools (9: 3DMax, Audition, Camtasia, Final Cut Studio, Flash, Illustrator, Logic Studio, Photoshop, Veodia)
   d. Word Processors, Page Layout, and Document Format Tools (4: Acrobat, Office, OpenOffice, QuarkXPress)
   e. Database Applications (2: Access, Oracle)
   f. Web-Based Collaboration and Web 2.0 Authoring Tools (4: ELGG, Live Meeting, Participate, WebX)
   g. Web Page Editors (2: Easy WebContent, Editor)

Note. Top-level items, left to right. Number of tools noted in (for each final category). Most names copyright/registered by their owners.

Our intent, therefore, is not to provide a comprehensive overview of the potential dimensions of tutoring systems or a list of all such systems currently available, the latter of which is almost certainly impossible, and the former of which, if tenable at all, is so only in the very short term. The state of the art of computerized tutoring systems as well as computer technology more generally is evolving so quickly that any such listing is fated for prompt obsolescence. We aim instead to pull lessons from this review, noting dimensions that tutoring tools can vary upon, and insights that are possible from this review.

Table 2 shows the principal dimensions along which these systems used in tutoring vary, including their technology, what they teach, their developers, their users, and their instructional approach and purpose. These dimensions were based on a grounded theory analysis (Muller, 2014) of extracting meaning and building our taxonomy as we found related systems. We also took some dimensions from Murray (1999). The dimensions can be organized into groups. The technological dimensions include how they work, and what hardware, software, and network connectivity they need to work. The content that they teach can also vary quite widely, and some content, such as simulations, will also bring additional differentiation on hardware, software, and network connectivity that further specialize such systems. There are several dimensions related to instructional designers and how they use the tutoring system. These differences are similar to factors that differentiate the learners. Finally, the instructional approach used, and the purpose of the tool also differentiate the tutors and tutoring tools.

Some of these dimensions (i.e., features) can be classified with binary inclusion/exclusion (e.g., requires Internet connectivity or not, supports user-generated content or not), while other dimensions are multi-valued (e.g., level of education of targeted students, programming language used) or continuous (e.g., interface difficulty). Taken together, the combinatorial space of these

dimensions is substantial; we estimate from this ontology that there are on the order of $2^{\text{number-of-dimensions}}$ tutoring systems and related tool types, or about 2^{40} different types of systems. Granted, for some situations some dimensions will be don't-care values. But even if half of the dimensions are relevant, the resulting space is still a very large number of different tutoring environments.

Table 2. Dimensions influencing the adoption, use, and development of tutoring systems, a partial list.

1. Technology supporting the tutor, instructional designers, and learners
   - 1.1. Interface difficulty/complexity
   - 1.2. Internet connectivity—required or not, available (or not)
   - 1.3. Runs on mobile devices (or not)
   - 1.4. Runs with or requires programming language: Java, C++, Flash, etc
   - 1.5. Free or paid (cost, and cost per learner)
   - 1.6. Source code available
   - 1.7. Commercially available, or available only to researchers
   - 1.8. Demo or production
   - 1.9. Hardware/OS required of end user (Mac, PC, Unix)
   - 1.10. Adaptive/intelligent (or not)
   - 1.11. Other software used (e.g., browser).
   - 1.12. Required software download
   - 1.13 Ability to print, record, or export tutor content
2. Types of tutor content
   - 2.1. Type of knowledge taught: declarative, procedural knowledge, recognition, recall, perceptual-motor
   - 2.2. Uses/needs as instructional material: movies, audio, pictures, text, simulations (these features alone yield 2^5 combinations), including technical requirements on video, audio, pictures representations such as .mp4 and .avi, mp3 and jpg.
   - 2.3. Supports user-generated content (or not)
   - 2.4. Domain-specific, or domain-general
   - 2.5. Theoretical foundation—ACT-R, Eriksson/TOT, KRK
   - 2.6. Fidelity of instructional material
3. Types of teachers/instructional designers
   - 3.1. Level of education required of Instructional Designers
   - 3.2. Level of training on the system allowed for Instructional Designers (e.g., needs to be usable immediately or some/extensive training required)
   - 3.3. Assessment/analytic tools (dashboards, skill trees, problem-area notifications, etc.) provided to teachers
   - 3.4. Tools to communicate with students for individual training/coaching
   - 3.5. Tools for students to communicate with each other, including subgroups
   - 3.6. Provides discussion/collaboration forum for students
4. Types of learners
   - 4.1. Educational level of learner (primary, secondary, tertiary, post-grad, commercial government, military)
   - 4.2. Number of learners supported (demo, one class, one university, 1000s, 10,000s, 1M, more)
   - 4.3. Motivation of learners
   - 4.4. Time allowed to learn the tutoring system (minutes/hours/blocks or just short times)
5. Instructional approach and purpose
   - 5.1. Competition across students
   - 5.2. Publicly certifiable knowledge and formally documentable (e.g., surgery boards, RN certificates)
   - 5.3. Learners can see their progress and grade
   - 5.4. Results need to be shared beyond the tutor itself (e.g., part of a university course)
   - 5.5. Competency based on reaction time, percent correct, or strategy choice, the range of behavior that is aggregated (trial, series of trials, section), time of learning, and/or forgetting
   - 5.6. The range of behavior that is recorded and how it is aggregated (trial, series of trials, section, time of learning, and/or forgetting)
   - 5.7. Tutor for teaching or for research
   - 5.8. For instruction or preparation of instructional materials

Variation along these dimensions can lead to radically different tutoring systems. We can illustrate this by examining a few examples from Table 2. Consider first the level of interface complexity. There exist tutors whose operation requires the instructional designer to have extensive knowledge and training in the tutor, perhaps on the order of an advanced graduate degree in that specific area, before they can implement a basic tutoring module (e.g., the Carnegie Mellon Tutors)(Anderson, Corbett, Koedinger, & Pelletier, 1995). On the other end of the spectrum, there are systems that require the instructional designers to simply type text into forms on web pages. More complex or difficult architectures require more effort and time investment from instructional designers but may offer more targeted interventions, better data collection capabilities, or the ability to conduct learning experiments.

As another example, an important technological dimension along which tutoring systems differ is their reliance on a network connection. Some architectures and tools may require users to be connected to the Internet if data, models, or applications are hosted on a remote machine. Such systems often have the advantage of providing ready-made infrastructure to the end-user and do not require them to set up tutoring environments of their own. On the other hand, there are use-cases in which a network connection may not be possible or desirable, such as during military field operations. In such environments, it may be useful to have off-line capable tutors, perhaps for brushing up on knowledge of vehicle maintenance or a foreign language while traveling.

For researchers and instructional designers, a major dimension of tutoring systems is the theoretical or scientific foundation upon which they rest, particularly whether a specific theory of learning has driven the development of the system. Different models of learning will lead to different beliefs about how best to teach learners, and thus to different implementation decisions.

Being explicit about this dimension would be particularly useful for scientists looking to compare and contrast different tutoring systems as a way of testing different theories of learning.

One of the major advantages of a computerized tutoring system over other methods, such as classroom instruction or mass-distributed learning materials like books and videos, is that they allow for data collection. This can be useful for making a tutoring system adaptive, for testing theories of learning schedules, or for refining models of pedagogy. The amount and types of data that is collected varies widely between systems but may include time on task (and sub-tasks, depending on the skill model implemented in the system), time to mastery, ease of use of the system, use of social tools like forums, path of progression through tutor, or progress across multiple tutors/subjects.

Lessons from the Survey

With the tutoring tools survey in hand, we were able to pull out at least 35 dimensions that tutoring tools can vary on. Considering these two results together we can pull out several lessons.

Our taxonomy and summary of tutoring systems is incomplete

The classificatory scheme presented in Figure 1 and Table 1 is incomplete in that it does not include all the systems that exist, and thus the dimensions that may be used differentiate use contexts may be incomplete. Any similar effort will suffer from the same shortcoming—every reader will know of further systems and further types that belong in the tree, based on their particular environment and concerns. In addition, the survey will be overcome by time—the problem of completeness is compounded by the speed at which tools are developed, distributed, and become obsolete, meaning that even the most comprehensive review of all such tools (if this is even possible in the first place) will not remain exhaustive for long.

The ontology may not be worth pursuing further

Given that this ontology and listing are incomplete, a related insight is that it is may not be worth pursing it to create a complete ontology. Because of its breadth this would be an endless task, that will always remain incomplete, and it will remain difficult to define the boundaries.

However, it may be helpful to create summaries from time to time, particularly more focused reviews, but attempting to maintain such a broad list will likely be frustrating. Perhaps a more effective approach would be to use the ontology to support a decentralized, community-maintained web repository or wiki that compares and cross-references available systems by dimensions and use-cases, where the community of researchers, developers, and learners using the various tutoring systems can keep up-to-date records of the systems in use and their availability. Such a review or system would be valuable to stakeholders connected to tutoring systems, particularly those who are building tutoring systems and those who are making technical acquisitions (e.g., government agencies and school boards looking to adopt new training platforms).

The ontology is useful

The ontology and dimensions, despite being incomplete and not worth pursuing further, are useful because they lead to several insights about tutoring and tutoring systems. We take these insights up next.

There are multiple tools to help build tutoring systems

This listing of tools and tool types and the dimensions upon which they can vary is useful in various ways. The listing suggests that there are a lot of tools that can be used. This is helpful to know if you are about to build a system or are requesting support to build such a system—it is worth at least a quick look at the ontology to see if there are systems similar to what you are

trying to do, or if there are components or lessons you can use in your situation. This review suggest that developers should first clarify the features of system they are trying to develop, perhaps using the ontology in Table 2, and then determine if there are many unique features they need, in which case they may indeed end up developing their own system.

The listing also suggests that some attention should be spent looking at systems and system components that can be re-used rather than created from scratch. Many systems recreate editors and content producers inside themselves when, in most cases they need not. The range of powerful supporting tools available (e.g., Adobe Photoshop for editing instructional content) suggests that if a component process of creating a tutoring module can be done effectively outside the tutoring system, such functionality need not be built into new systems. This type of modular approach can save time and money, and lead to better final results.

The complexity of contexts reduces reuse

The summary of dimensions can be useful to understand why there are so many different tutoring systems—there is indeed a very broad range of situations that vary across a large number of dimensions. It is true that some dimensions will not matter in some cases, but it is also clear that many dimensions are important in many cases.

In developing our tutoring approach (D2P: Ritter et al., 2013), we examined a wide range of similar systems, both to learn from them and to see if we can use them instead of creating our own. For example, some of the dimensions we needed were: access to the source code, to be in a language we can find student programmers for, not be dependent on the Internet at run time, to be able to use a wide range of instructional materials including video and simulations. We examined several systems, but there we found none available meeting those constraints, although

we could take useful lessons from them. So, we had to build our own. Our own behavior was consistent with this conclusion.

Including other stakeholders can be important

One of our primary interests in studying tutoring systems is to better understand learning and forgetting, gaining knowledge that can then, hopefully, be used to improve learning. We are currently developing a tutoring architecture, called Declarative-to-Procedural (D2P), to apply and to test theories of learning and forgetting schedules on the consolidation of different forms of knowledge. So far, it is becoming a domain-general, adaptive tutoring system, but with constraints on numerous of the dimensions shown in Table 2.

One of our principle aims in building the architecture is to meet what we see as a fertile and under-developed space: a middle-ground of architecture power and complexity of use where more stakeholders (particularly instructional designers and researchers) are included in the design process and are better supported than has traditionally been done. This view arises from a book edited by Pew and Mavor (2007). In such a system, instructional designers are treated as stakeholders alongside the architects and learners. Treating instructional designers as stakeholders will help them create better tutors and increase reuse of systems that meet their needs. That instructional designers are stakeholders can be seen by the several dimensions referring to them in Table 2. If it is difficult for them to include video, they will be less likely to include video, limiting how they can share information with learners. If it is difficult to create and use questions, they will create and include fewer questions, leading to less practice and less learning.

We believe that the survey of tutoring systems that we have presented here, and others of this nature, will be useful in identifying such niches that could be profitably and usefully filled, but are not yet filled.

Final thought

The DNA of tutors taken from the dimensions list in Table 2 contains about 40 bits of information. Thus, there is about 2^{40}, or a trillion different types of tutors or tutoring situations. Thus, the large number of different tutors and limited reuse between projects are partially due to the large number of different tutoring situations. Most tutoring projects are thus working in different situations; researchers will be working in their own subspace; and universities, if they adopt a universal system, will fail to support many users because there are many dimensions of interest for instructional situations.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

Frank E. Ritter is a professor of IST and psychology at Penn State. He is interested in theories of learning particularly within cognitive architectures and how to improve learning theories by examining learning schedules, modeling them, and building tutoring systems that apply them. He has been working with the Declarative to Procedural (D2P) tutoring architecture since 2015.

Michael “Q” Qin is a program officer at the Office of Naval Research supporting a program in autonomous vehicles and intelligent systems.

Korey MacDougall is interested in learning, simulation, and building tutoring systems. He helped build the D2P tutoring system.

Chungil "Chad" Chae received a PhD from Penn State in 2018. He is interested in workforce development and learning.

References

Adams, A., Lunt, P., & Cairns, P. (2008). A qualitative approach to HCI research. In P. Cairns & A. L. Cox (Eds.), *Research methods for human-computer interaction* (pp. 138-157). Cambridge, UK: Cambridge University Press.

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *Journal of the Learning Sciences, 4*(2), 167-207.

Dear, B. (2017). *The friendly orange glow*. New York, NY: Pantheon.

Grubisic, A., Stankov, S., & Zitko, B. (2015). Adaptive courseware: A literature review. *Journal of Universal Computer Science, 21*(9), 1168-1209.

Muller, M. (2014). Curiosity, creativity, and surprise as analytic tools: Grounded Theory Method. In J. Olson & W. A. Kellog (Eds.), *Ways of knowing in HCI* (pp. 25-48). New York, NY: Springer Science+Business Media.

Murray, T. (1999). Authoring Intelligent Tutoring Systems: An analysis of the state of the art. *International Journal of Artificial Intelligence in Education, 10*, 98-129.

Pew, R. W., & Mavor, A. S. (Eds.). (2007). *Human-system integration in the system development process: A new look*. Washington, DC: National Academy Press. books.nap.edu/catalog/11893, checked Feb 2020.

Psotka, J., Massey, L. D., & Mutter, S. A. (Eds.). (1988). *Intelligent tutoring systems: Lessons learned*. Hillsdale, NJ: Erlbaum.

Ritter, F. E., Yeh, K.-C., Cohen, M. A., Weyhrauch, P., Kim, J. W., & Hobbs, J. N. (2013). Declarative to procedural tutors: A family of cognitive architecture-based tutors. In *Proceedings of the 22nd Conference on Behavior Representation in Modeling and Simulation*, 13-BRIMS-127. 108-113. Centerville, OH: BRIMS Society.

Singh, R., Saleem, M. B., Pradhan, P. R., Heffernan, C., Heffernan, N. T., Razzaq, L., et al. (2011). Improving K-12 Homework with Computers. In *Proceedings of the Artificial Intelligence in Education Conference 2011*, 328-336.

Sleeman, D., & Brown, J. S. (Eds.). (1982). *Intelligent Tutoring Systems*. London: Academic Press.

Smith, S. G., & Sherwood, B. A. (1976). Educational uses of the PLATO computer system. *Science, 192*(4237), 344–352.

Sottilare, R. A., Graesser, A. C., Lester, J., & Baker, R. (2017). Special issue on the Generalized Intelligent Framework for Tutoring (GIFT): Creating a stable and flexible platform for innovations in AIED research. *International Journal of Artificial Intelligence in Education, 28*, 139–151.

VanLehn, K. (1982). Bugs are not enough: Empirical studies of bugs, impasses and repairs in procedural skills. *Journal of Mathematical Behavior, 3*(2), 3-71.

VanLehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist, 46*(4), 197-221.

Wenger, E. (1987). *Artificial intelligence and tutoring systems: Computational and cognitive approaches to the communication of knowledge*. Los Altos, CA: Morgan Kaufmann.