Int J Artif Intell Educ (2018) 28:336–384
https://doi.org/10.1007/s40593-017-0157-9

ARTICLE

Authoring Tools for Designing Intelligent Tutoring Systems: a Systematic Review of the Literature

Diego Dermeval[1] · **Ranilson Paiva**[2] · **Ig Ibert Bittencourt**[2] · **Julita Vassileva**[3] · **Daniel Borges**[2]

Published online: 31 October 2017

Abstract Authoring tools have been broadly used to design Intelligent Tutoring Systems (ITS). However, ITS community still lacks a current understanding of how authoring tools are used by non-programmer authors to design ITS. Hence, the objective of this work is to review how authoring tools have been supporting ITS design for non-programmer authors. In order to meet our goal, we conduct a Systematic Literature Review (SLR) to identify the primary studies on the use of ITS authoring tools, following a pre-defined review protocol. Among the 4622 papers retrieved from seven digital libraries published from 2009 to June 2016, 33 papers are finally included after applying our exclusion and inclusion criteria. We then identify the main ITS components authored, the ITS types designed, the features used to facilitate the authoring process, the technologies used to develop authoring tools and the time at which authoring occurs. We also look for evidence of the benefits of ITS authoring tools. In summary, the main findings of this work are: (1) there is empirical evidence of the benefits (i.e., mainly in terms of effectiveness, efficiency, quality of authored artifacts, and usability) of using ITS authoring tools for non-programmer authors, specially to aid authoring of learning content and to support authoring of model-tracing/cognitive and example-tracing tutors; 2) domain and pedagogical models have been much more targeted by authoring tools; (3) several ITS types have been authored, with an emphasis on model-tracing/cognitive and example-tracing tutors;

✉ Diego Dermeval
diego.matos@penedo.ufal.br

1 Penedo Educational Unity, Campus Arapiraca, Federal University of Alagoas, 57200-000, Penedo, AL, Brazil

2 Computing Institute, Federal University of Alagoas, 57072-900, Maceió, AL, Brazil

3 Department of Computer Science, University of Saskatchewan, Saskatoon, SK, Canada

(4) besides providing features for authoring all four ITS components, current authoring tools are also presenting general features (e.g., view learners' statistics and reuse tutor design) to create broader authoring tools; (5) a great diversity of technologies, which include AI techniques, software solutions and distributed technologies, are used to develop ITS authoring tools; and (6) authoring tools have been mainly targeting ITS design before students' instruction, but works are also addressing authoring during and/or post-instruction relying both on human and artificial intelligence. We conclude this work by showing several promising research opportunities that are quite important and interesting but underexplored in current research and practice.

Keywords Intelligent tutoring systems · Authoring tools · Systematic literature review · Intelligent tutoring systems design

Introduction

Intelligent Tutoring Systems (ITSs) are concerned with the use of artificial intelligence techniques for performing adaptive tutoring to learners according to what they know about the domain (Sleeman and Brown 1982). As reported by du Boulay (2016), there have been a number of recent positive reviews in support of the effectiveness of ITSs (Kulik and Fletcher 2016; Ma et al. 2014; Steenbergen-Hu and Cooper 2013, 2014; VanLehn 2011). Thus, it is well known that well-designed ITS can successfully complement and substitute other instructional models at all educational levels and in many common academic subjects (Ma et al. 2014).

Despite all this positive evidence, ITS design remains costly and expensive (Woolf 2010). Hence, for many years, researchers develop ITS authoring tools in order to speed up ITS development, to reduce production efforts, to increase the number and diversity of available tutors, to extend the number of participants in ITS development process and so on (Murray 2003; Woolf 2010; Sottilare et al. 2015).

Due to the increasing interest of using authoring tools to design ITS, researchers have been concerned in summarizing the contributions of these systems in order to describe the state of the art on this topic. Murray (2003) presents a deep analysis of the state of the art until 2003– updating a previous survey (Murray 1999) with similar aims. It describes the ITS types built with authoring tools, the features and methods used to facilitate authoring, the evaluation strategies used, the availability of authoring tools and lessons learned from using these systems. Murray (2003) also presented some unanswered questions regarding ITS authoring tools such as the extent to which the difficult task of modeling ITS could be scaffolded, identifying instructional situations that are both specific enough to make authoring template-based, yet general enough to be attractive to many educators, and whether ITSs would ever be in demand enough to warrant the effort of building authoring tools for them. Several future directions are pointed out by Murray (2003), suggesting more empirical testing of authoring tools using multiple authors and domains, more research on authoring student models, more research on the effectiveness of instructional strategies and so on.

Later on, Woolf (2010) used Murray's classification to update the state of the art on the topic. Woolf (2010) presents examples of other authoring tools, discusses issues regarding authoring tools design tradeoffs (e.g., specific vs. general tutors), and describes a variety of primary building blocks for developing ITS authoring tools–classified into four levels: representation and domain, student, teaching and communication knowledge. Woolf (2010) also mentioned that although intelligent tutors have already succeeded in finding use within education, the development of ITS authoring tools has been progressing slowly and pointed out issues mainly associated with the use of ITS in classrooms. More recently, Sottilare et al. (2015) edited a comprehensive collection of papers discussing several important issues about ITS authoring tools, for instance, perspectives of authoring tools and methods, approaches to reduce workload and skill requirements in ITS authoring and so on. It also presents several contributions for authoring different ITS types, such as model-tracing, agent-based and dialog-based tutors.

The reviews heretofore have largely focused on implementation details and system design. However, it has not been until fairly recently that empirical research has been emphasized with ITS authoring tools and, in the meantime, new technologies and trends have appeared (e.g., mobile learning, gamification, learning analytics and so on). Hence, although these contributions summarize well the body of knowledge regarding ITS authoring tools so far, they could not capture all the recent and interesting aspects of the field. For instance, analyzing the evidence presented by empirical works, investigating current tools in light of "life-cycle" authoring–i.e., in which authoring might occur at different stages of learning, including, before, during and after instruction–identifying special features and technologies used to facilitate the authoring process for non-programmers and so on. Moreover, they did not perform any kind of systematic investigation of the literature covering the use of authoring tools to design ITS. Hence, the objective of this work is to conduct a systematic review of the literature to review how authoring tools have been supporting ITS design[1] for non-programmer authors recently. We also investigate if there is real evidence for improvements of using these tools to support ITS design. Thus, inspired by Murray's survey research questions, we intend to understand:

- which ITS components can be authored?;
- which ITS types can be designed by authoring tools?;
- which features facilitate the ITS authoring process?;
- what authoring technologies have been used to design ITS?;
- when does the authoring occur?;
- what is the evidence that support reported benefits of using ITS authoring tools?

In this paper, we use the systematic literature review (SLR) method (Kitchenham and Charters 2007) to identify, evaluate, interpret and synthesize the available studies

[1]Note that ITS design can cover many aspects of ITS such as the creation of content, pedagogical intervention, the content of feedback messages, interfaces, features to be included, and so on. In this way, although we acknowledge that there is a clear difference between creating content for ITS with letting people designing their own tutor, we are considering in this paper both contributions if they are proposed for non-programmer authors.

to answer particular research questions on the application of authoring tools in ITS design and to establish the state of evidence with in-depth analysis. In an SLR, all decisions used to select, extract data and compile information about papers are meant to be explicit, allowing the reader to check the review process (and even reproduce it) as well as assessing the potential of bias (Garg et al. 2008). Thus, the SLR method tends to be more transparent than narrative reviews and has been already used in the context of ITS research as a prior step to conduct a meta-analysis (Kulik and Fletcher 2016; Ma et al. 2014; Steenbergen-Hu and Cooper 2013, 2014; VanLehn 2011).

This paper presents the results of an SLR on studies published from January 2009 to June 2016[2] and was conducted following a pre-defined review protocol. Our decision on such period was made to reduce repetitive effort and make use of existing work since Woolf (2010) provides a general description of the use of authoring tools to design ITS before 2009 updating the analysis of state of the art provided by Murray (2003). Moreover, we also intend to gather more recent papers about the topic in order to get insights as well as to consider emerging technologies that could be used along with authoring tools (e.g., mobile learning, gamification, learning analytics and so on) for non-programmer authors.

This paper is organized as follows. "Methods" describes the SLR method used in this review. "Results and Analysis" first presents the results of the quality assessment and an overview of the studies. It then reports the findings of the review along with a detailed analysis and discussion of each research question. "Discussion" discusses the scope of this systematic literature review, some threats to the validity of our work and points out further research to be explored on the use of authoring tools in ITS design. Finally, "Conclusions" presents conclusions and future works.

Methods

A Systematic Literature Review (SLR) is a means of identifying, evaluating and interpreting the available research findings related to a research question, topic area, or phenomenon. The main purpose of conducting a systematic review is to gather evidence on which to base conclusions (Kitchenham and Charters 2007).

In order to perform this SLR, the guidelines and the systematic review protocol template proposed by Kitchenham and Charters (2007) were used. The SLR process includes several activities, which can be grouped into three main phases: planning of the SLR, conducting the SLR and reporting the SLR. It consists of the following steps: i) identification of the need for a systematic review; ii) formulation of a focused review question; iii) a comprehensive, exhaustive search for primary studies; iv) quality assessment of included studies; v) identification of the data needed to answer the research question; vi) data extraction; vii) summary and synthesis of study results; viii) interpretation of the results to determine their applicability; and ix) report-writing.

[2]The studies are until June 2016, because we've conducted the search and selection process of the review in June 2016.

A software tool, called StArt (State of the Art through Systematic Reviews) (LAPES 2014), was used to support the SRL protocol definition. It is used to provide support to researchers conducting SLRs. StArt has been empirically evaluated and it was demonstrated that such tool had positive results in the execution of SLRs (Hernandes et al. 2012).

Research Questions

This systematic review's purpose is to understand and synthesize how authoring tools support intelligent tutoring systems design regarding non-programmer authors' point of view and identify to what extent these tools have been applied for designing this kind of system. Thus, we intend to answer the main research question:

How are authoring tools supporting the design of intelligent tutoring systems for non-programmer authors?

Based on the main research question, specific questions were raised according to authoring tools and ITS aspects that we are interested. The questions, along with their descriptions and motivations are described in Table 1.

Table 1 Research questions and motivations

Research question	Description and motivation
RQ1. Which ITS components can be authored?	This question provides a starting point to understand which are the main ITS components (i.e., student model, domain model, pedagogical model and interface model) supported by the use of authoring tools;
RQ2. Which ITS types can be designed by authoring tools?	This question intends to identify which are the main ITS types (e.g., example-tracing, constraint-based and so on) that are been designed by the use of authoring tools;
RQ3. Which features facilitate the ITS authoring process?	This question aims to describe how authoring tools are supporting the authoring process of ITS. It is important because it provides a set of contributions regarding the use of authoring tools to address ITS design, which can be used by researchers that might be interested in using authoring tools for this kind of educational system;
RQ4. What authoring technologies have been used to design ITS?	This question intends to identify which are the main technologies used to develop authoring tools in order to design ITS. The answer to this question is important because it can serve as a guide to researchers that might use some specific technology to develop authoring tools for ITS;
RQ5. When does the authoring occur?	The answer to this question allows to identify when the authoring process occurs (i.e., pre-instruction, during instruction and post-instruction). This question investigates how authoring tools are supporting different stages of the ITS life-cycle;
RQ6. What is the evidence that support reported benefits of using ITS authoring tools?	This question intends to analyze if such studies provide some evidence that the use of authoring tools benefits the ITS design process. Evidence should consider positive and negative results including empirical and non-empirical evaluation. They are important since they form a knowledge base about the use of authoring tools in ITS

Inclusion and Exclusion Criteria

The aim of defining a criterion is to identify those primary papers which provide direct evidence about the research questions and also to reduce the likelihood of bias (Kitchenham and Charters 2007). Note that we consider as primary papers the studies which present some kind of proposal to the area or present some kind of empirical evaluation of its contributions, whereas secondary papers are studies which only review a topic area, e.g., surveys, systematic literature reviews or systematic mappings.

Studies were eligible for inclusion in the review if they presented a peer-reviewed primary study, published since January 2009 to June 2016 and that presented some contribution on the use of authoring tools to support ITS design. As previously explained, our decision on such period was made to reduce repetitive effort and make use of existing work as well as to gather more recent papers about the topic, considering emerging technologies that could be used along with authoring tools.

Studies were excluded if they were secondary, short papers, non-peer reviewed, duplicated, non-English written, gray literature papers (e.g., books, theses, dissertations and so on), redundant papers of same authorship,[3] position papers and if their focus was not using authoring tools to support ITS design for non-programmer authors. Furthermore, this research is concerned with generic and technology/paradigm ITS authoring tools, i.e., we are not including works that propose authoring tools that need to handle strict ITS constraints. For this reason, simulation, augmented reality, serious games, storytelling, and disability (focusing on learners' disabilities) exclusive papers were also excluded. For instance, an ITS authoring tool that considers learners' disabilities (e.g., blindness) should need to design a special pedagogical model tied to such disability that would not be generic enough to be used in other contexts. The summarized inclusion and exclusion criteria are presented in Table 2.

Sources Selection and Search

The search strategy included only electronic databases and was validated by experts on ITS and authoring tools. According to Chen's recommendation (Chen et al. 2010), the following electronic databases were automatically searched: ScienceDirect,[4] ISI Web of Science,[5] Scopus,[6] SpringerLink,[7] ACM Digital Library,[8] IEEE Xplore[9] and Compendex.[10]

[3]If similar papers are included from the same authorship, we keep in the review the more complete and recent paper (priority is given to journal papers)

[4]http://www.sciencedirect.com/

[5]http://apps.webofknowledge.com

[6]http://www.scopus.com

[7]http://link.springer.com/

[8]http://dl.acm.org/

[9]http://ieeexplore.ieee.org

[10]http://www.engineeringvillage.com/

Table 2 Inclusion/exclusion criteria

#	Inclusion criterion
1	Primary studies
2	Peer-reviewed studies
3	Study published between January 2009 and June 2016
4	Studies that use authoring tool to design ITS for non-programmer authors
#	Exclusion criterion
5	Secondary studies
6	Short-papers (≤ 5 pages)
7	Non peer-reviewed studies
8	Duplicated studies (only one copy of each study was included)
9	Non English written papers
10	Gray literature
11	Redundant paper of same authorship
12	Position paper
13	Studies that do not present or evaluate any authoring tool for non-programmers
14	Papers about simulation
15	Papers about augmented reality
16	Papers about serious games
17	Papers about storytelling
18	Papers about disability
19	Studies that do not use authoring tools to design ITS

Figure 1 shows the systematic review process and the number of papers identified at each stage. Before describing these stages, it is worth emphasizing that, although the scope of this paper is reviewing the use of authoring tools in ITS design, this research is part of an ongoing work which intends to review the use of authoring tools in computers and education, including for example, several types of educational systems (e.g., computer supported collaborative learning, massive open online courses, adaptive educational hypermedia systems, etc) and research trends (e.g., gamification, mobile learning and education data mining/learning analytics). Hence, the search and selection strategy (i.e., the search string and Steps 1-5) aims to capture studies related to all these topics. As such they will be useful for several other studies. The papers related to ITS, which are the focus of this review, are only identified and selected in Step 6 of the process, as it will be further described.

In Step 1 the studies were obtained from electronic databases using the following search terms:

(1) "authoring tool" OR "authoring system" OR "intelligent authoring"
(2) "computers and education" OR "e-learning"
(3) "educational environment" OR "educational system" OR "learning environment"
(4) "learning management system"

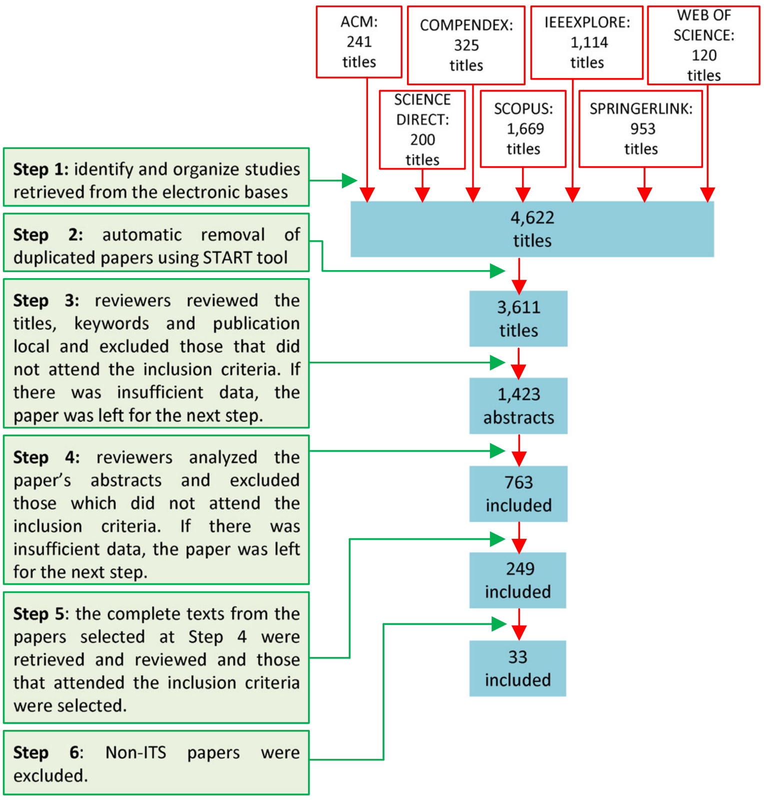
ACM: 241 titles
COMPENDEX: 325 titles
IEEEXPLORE: 1,114 titles
WEB OF SCIENCE: 120 titles
SCIENCE DIRECT: 200 titles
SCOPUS: 1,669 titles
SPRINGERLINK: 953 titles
Step 1: identify and organize studies retrieved from the electronic bases
4,622 titles
Step 2: automatic removal of duplicated papers using START tool
3,611 titles
Step 3: reviewers reviewed the titles, keywords and publication local and excluded those that did not attend the inclusion criteria. If there was insufficient data, the paper was left for the next step.
1,423 abstracts
Step 4: reviewers analyzed the paper's abstracts and excluded those which did not attend the inclusion criteria. If there was insufficient data, the paper was left for the next step.
763 included
249 included
Step 5: the complete texts from the papers selected at Step 4 were retrieved and reviewed and those that attended the inclusion criteria were selected.
33 included
Step 6: Non-ITS papers were excluded.

Fig. 1 Paper selection flowchart

(5) "m-learning" OR "mobile learning"
(6) "t-learning" OR "tv learning"
(7) "online education" OR "online learning" OR web-based education" OR "semantic web-based education" OR "semantic web and education"
(8) "collaborative learning" OR "computer supported collaborative learning" OR "CSCL"
(9) "intelligent tutoring system" OR "intelligent educational systems"
(10) "MOOCS" OR "massive open online courses"
(11) "adaptive educational hypermedia systems"
(12) "adaptive educational systems" OR "adaptive learning systems" OR "artificial intelligence in education"
(13) "gamification"

These search terms for several applications of authoring tools to computers and education were combined in the following way:

(1 AND (2 OR 3 OR 4 OR 5 OR 6 OR 7 OR 8 OR 9 OR 10 OR 11 OR 12 OR 13))

The definition of these terms was based on two main sources: i) the scope of relevant journals on the topic (e.g., the International Journal of Artificial Intelligence and

Education (IJAIED) and IEEE Transactions on Learning Technologies) in order to identify different types of educational systems and ii) asking suggestions to experts on the topic (authoring tools and ITS). Furthermore, as we intended to retrieve recent papers in the literature, our search only considered the period between January 2009 and June 2016, which is an inclusion criterion, as described in "Inclusion and Exclusion Criteria".

The search results (4622 papers) were automatically downloaded and were inserted into and organized with the aid of StArt tool. Figure 1 depicts the steps of the selection process showing the number of studies in each one these steps.

At Step 2, duplicated papers were automatically detected and removed using the StArt tool, remaining a set of 3,611 papers. Then, in Step 3 authors reviewed titles, keywords, and publication venue of each paper and excluded those that were not related to the research questions (− 2188 papers). If there was insufficient data, the paper was left for the next assessment. After finishing the Step 3, 1423 papers remained in the selection process and reviewers analyzed, in Step 4, paper's abstracts and excluded those according to 14 exclusion criteria (#4–18 criteria from Table 2), excluding 660 papers. If there was insufficient data, the paper was left for the next step.

In Step 5, the complete texts of the papers selected at Step 4 (763 papers) were retrieved, the introduction and conclusion of each paper were read and each paper was full-screened. Papers were excluded according to the #4-18 exclusion criteria again (− 514 papers).

Until Step 5, any application of authoring tool to computers and education was considered to be included in the review. Recall that this is intentional, as we may use the studies identified so far for several types of research under development. Hence, the specific exclusion criterion for non-ITS authoring papers was applied, in Step 6, to the 249 remaining studies of Step 5, in order to filter the papers exclusively related to ITS design (the focus of this paper). As a result, 33 papers were finally included for the next stage of the review.

Quality Assessment

The quality assessment (QA) of selected studies was achieved by a scoring technique to evaluate the credibility, completeness, and relevance of the selected studies. All papers were evaluated against a set of 10 quality criteria. Eight of them were adapted from existing study quality assessment criteria used in the literature, the remaining two questions were proposed according to the scope and research questions of this systematic literature review. The assessment instrument used is presented in Table 3. Q1, Q2, Q3, Q4, Q5, Q6, Q9, and Q10 were adopted from the literature, while Q7 and Q8 were proposed.

We relied on systematic literature reviews published in a high reputation venue (i.e., Information and Software Technology Journal) in the context of empirical software engineering research to define seven of the quality assessment criteria. In particular, we adapted some of our criteria following the works by Mahdavi-Hezavehi et al. (2013) (Q1 and Q5), Dyb and Dingsyr (2008) (Q2, Q3, Q5 and Q10), Achimugu et al. (2014) (Q4 and Q10) Dermeval et al. (2016) (Q6) and Ding et al. (2014) (Q9).

Table 3 Study quality assessment criteria

#	Questions	Possible answers
Q1	Is there a rationale for why the study was undertaken? (Mahdavi-Hezavehi et al. 2013)	Y=1, N=0, P=0.5
Q2	Is the paper based on research (or is it merely a "lessons learned" report based on expert opinion)? (Dyb and Dingsyr 2008)	Y=1, N=0
Q3	Is there a clear statement of the goals of the research? (Dyb and Dingsyr 2008)	Y=1, N=0, P=0.5
Q4	Is the proposed technique clearly described? (Achimugu et al. 2014)	Y=1, N=0, P=0.5
Q5	Is there an adequate description of the context (industry, laboratory setting, products used and so on) in which the research was carried out? (Dyb and Dingsyr 2008; Mahdavi-Hezavehi et al. 2013)	Y=1, N=0, P=0.5
Q6	Does the study provide a tool? If yes, is the tool available for download or on the web? (Dermeval et al. 2016)	Y=1, N=0, P=0.5
Q7	Was the study empirically evaluated?	Y=1, N=0
Q8	Is there a discussion about the results of the study?	Y=1, N=0, P=0.5
Q9	Are the limitations of this study explicitly discussed? (Ding et al. 2014)	Y=1, N=0, P=0.5
Q10	Does the study also evaluate the proposal in industrial settings? (Dyb and Dingsyr 2008; Achimugu et al. 2014)	Y=1, N=0

Y = Yes, N = No, and P = Partially

The scores of questions Q2 and Q7 were determined using a two-grade scale score (Yes/No). If the answer were Yes, the study received 1 point in this question, otherwise, it received 0 point. Besides these alternatives, the questions Q1, Q3, Q4, Q5, Q8 and Q9 also allowed a third one. If the contribution was not so strong, the study received 0.5 point, consisting of a three-grade scale score to these questions. Q6 receives 1 point if the paper proposes an authoring tool which is available for download or on the web, it receives 0.5 point if the tool is not available and receives 0 if it does not propose an authoring tool. Q10 receives 1 point if the study is applied in industry and 0.5 point if its setting is academic.

After finishing the selection and extraction stages of our review, first and second authors independently assessed–according to the criteria presented in Table 3–the 33 papers included in the review. Then, the scores marked by the authors are organized in a spreadsheet and, for each criterion and paper, scores are compared to identify disagreements. All studies with non-agreement are discussed among all the authors, and the study is reevaluated with the aim of reaching consensus. The resulting study quality score is computed by finding the sum of all consensual scores of the answers to the questions in Table 3.

Data Extraction and Synthesis

After the definition of the search and the selection processes, the data extraction process was performed by reading each one of the selected papers. In order to guide this data extraction, the data collection from Kitchenham and Charters (2007) was adopted. During this stage, data was extracted from each of the 33 primary studies included in this systematic review according to an extraction form (see Table 4). This form enabled us to record full details of the papers under review and to be specific about how each of them addressed our research questions. Like the selection process, the data extraction was fully aided by the StArt tool.

Results and Analysis

A total of 33 studies met the inclusion criteria and their data were extracted. Prior to presenting the results and analysis for each research question, we depict the quality assessment results and give a detailed overview of the general characteristics of the studies.

The data are tabulated to show general information about the studies: identifier, authors, paper type, application context, and research method. Furthermore, we also tabulate data regarding our research questions and present bubble charts in order to provide a deeper visualization in the case of multiple categories to be presented.

Table 4 Extraction form

#	Study data	Description	Relevant RQ
1	Study identifier	Unique id for the study	Study overview
2	Date of data extraction		Study overview
3	Authors, Year, Title, Country		Study overview
4	Article source		Study overview
5	Type of article	Journal, conference, workshop, book chapter	Study overview
6	Application context	Industrial, academic	Study overview
7	Research method (based on Easterbrook et al. 2008)	Controlled experiment, case study, survey, ethnography, action research, illustrative scenario, not applicable	Study overview
8	Name of the contribution		Study overview
9	ITS component	What were the ITS components addressed by the authoring tool? (Student Model, Domain Model, Pedagogical Model and Interface Model)	RQ1
10	ITS type	What ITS type has been authored by the tool?	RQ2
11	Kind of support (feature)	How tools are supporting ITS authoring process?	RQ3
12	Technology	Which technologies have been used?	RQ4
13	Authoring time regarding course	When does the authoring occurs? (Pre-course, during the course and post-course)	RQ5
14	Evidence	What was the evidence which indicate that the use of authoring tools benefits the ITS design? (Negative argumentation, negative with empirical evaluation, positive argumentation, positive with empirical evaluation)	RQ6

Quality Assessment Results

The quality assessment of the selected studies is useful to increase the accuracy of the data extraction results. This evaluation helped to determine the validity of the inferences proffered and in ascertaining the credibility and coherent synthesis of results.

The quality assessment results are showed in the Table 5 according to the questions described in Table 3. In fact, the quality score of the papers is quite scattered. There are papers with high-quality scores, whereas there are papers with low-quality scores. Taken together, these 10 criteria provided a measure of the extent to which we could be confident that a particular study's findings could make a valuable contribution to this review.

Overview of the Studies

In following we depict general characteristics of the studies included in the review: type of source, research method and application context.

Application Context

The study settings were categorized either as an industry or academic context. Most the papers (31 studies) are considered academic, while 2 studies (S03 and S32) were conducted in an industrial setting. This result indicates that the application that the majority of authoring tool papers are published by the academic community since only 6% of the papers are applied in an industrial context.

Type of Source

The studies included in this review may be a journal, conference, workshop or book chapter publications. The majority of studies are conference papers (51.51%; 17 studies), followed by journal publications (36.36%; 12 studies) and workshop and book chapter publications, each with 6.06% (2 studies).

Table 13 in Appendix presents the distribution of selected studies over publication sources, including the publication name, type, count (i.e., the number of selected studies from each source), and the percentage of selected studies. The 33 selected studies are distributed over 25 publication sources.

As shown in Table 13, the leading venues in this study topic are the International Journal of Artificial Intelligence in Education (IJAIED), followed by the International Conference on Intelligent Tutoring Systems (ITS), International Conference on Artificial Intelligence (AIED) and the International Conference on Artificial Intelligence workshops. These results are expected since most papers on ITS are published by these communities, but might also indicate a positive aspect in the quality of the papers included in this review since the leading venues are high reputation vehicles in ITS research. However, a great number of the publications about the topic is widespread in different venues from computers and education, and artificial intelligence research areas.

Table 5 List of papers included in the review along with their quality scores

ID	Author	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Total Score	Qual.
S03	Aleven et al. (2009)	1	1	1	1	0.5	1	1	1	0.5	1	9	90.0%
S04	Aleven et al. (2016)	1	1	1	1	1	1	1	1	1	0	9	90.0%
S08	Blessing et al. (2009)	1	1	1	1	1	1	1	1	1	0	9	90.0%
S15	Gilbert et al. (2015)	1	1	1	1	1	1	1	1	1	0	9	90.0%
S20	MacLellan C.J. et al. (2014)	1	1	1	1	1	0.5	1	1	1	0	8.5	85.0%
S07	Blessing et al. (2015)	1	1	1	1	1	1	1	1	0	0	8	80.0%
S24	Mitrovic et al. (2009)	1	1	1	1	1	0.5	1	1	0.5	0	8	80.0%
S29	Suraweera et al. (2010)	1	1	1	1	1	0.5	1	1	0.5	0	8	80.0%
S11	Chou et al. (2011)	1	1	0.5	1	1	0.5	1	1	0.5	0	7.5	75.0%
S19	Lane et al. (2015)	1	1	1	1	1	0.5	1	1	0	0	7.5	75.0%
S23	Matsuda et al. (2015)	1	1	1	1	0.5	0.5	1	1	0.5	0	7.5	75.0%
S12	Devasani et al. (2012)	1	1	1	1	1	0	1	1	0	0	7	70.0%
S01	Abbas et al. (2014)	1	1	1	1	1	0.5	1	0	0	0	6.5	65.0%
S33	Zatarian-Cabada and Barrón-Estrada (2011)	0.5	1	1	1	1	0.5	1	0.5	0	0	6.5	65.0%
S10	Chakraborty et al. (2010)	1	1	1	1	0.5	0.5	1	0	0	0	6	60.0%
S18	Heffernan and Heffernan (2014)	1	0	1	1	0.5	1	0	1	0.5	0	6	60.0%
S21	MacLellan et al. (2015)	1	0	1	1	0.5	0.5	0	1	1	0	6	60.0%
S17	Guin and Lefevre (2013)	0.5	0	1	1	1	0.5	0	1	0.5	0	5.5	55.0%
S27	Paquette et al. (2010)	1	0	1	1	0.5	1	0	0.5	0.5	0	5.5	55.0%
S32	Wilches and Palacio (2014)	1	0	1	1	0.5	0	0	1	0	1	5.5	55.0%
S14	Fox et al. (2011)	1	0	1	1	1	0.5	0	0.5	0	0	5	50.0%

Table 5 (continued)

ID	Author	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Total Score	Qual.
S22	Marcus et al. (2010)	1	0	0.5	1	1	0.5	0	1	0	0	5	50.0%
S26	Olsen et al. (2014)	1	0	1	1	0.5	0.5	0	1	0	0	5	50.0%
S25	Olney and Cade (2015)	1	0	1	1	0.5	0.5	0	0.5	0	0	4.5	45.0%
S30	Troussas et al. (2014)	1	0	1	1	0.5	0.5	0	0	0	0	4	40.0%
S13	Escudero and Fuentes (2010)	1	0	0.5	1	0.5	0.5	0	0	0	0	3.5	35.0%
S16	Grubisic et al. (2009)	0.5	0	1	1	0	0.5	0	0.5	0	0	3.5	35.0%
S28	Sklavakis and Refanidis (2011)	0.5	0	1	1	0	0.5	0	0.5	0	0	3.5	35.0%
S31	Virvou and Troussas (2011)	1	0	0.5	1	0.5	0.5	0	0	0	0	3.5	35.0%
S06	Barron-Estrada et al. (2010)	0	0	0.5	1	0	1	0	0	0	0	2.5	25.0%
S09	Brawner (2015)	0	0	0.5	1	0	1	0	0	0	0	2.5	25.0%
S02	Alepis and Virvou (2014)	0	0	0.5	1	0	0.5	0	0	0	0	2	20.0%
S05	Barrón-Estrada et al. (2011)	0	0	0.5	0.5	0.5	0.5	0	0	0	0	2	20.0%
	Average	0.82	0.45	0.88	0.98	0.65	0.61	0.45	0.64	0.27	0.06	5.82	58.2%
	Standard Deviation	0.35	0.51	0.22	0.09	0.36	0.27	0.51	0.44	0.38	0.24	2.19	22%

Ids are alphabetically sorted by first author

Research Method

The classification of publications was based on the categories (i.e., controlled experiment, quasi-experiment, case study, survey research, ethnography and action research) defined by Easterbrook et al. (2008). However, we have defined two extra categories: illustrative scenario and not applicable. The first is appropriate for papers that just explain their contributions using small examples or argumentation. The latter refers to the papers that do not present any kind of research method or explanation of using the proposal.

As shown in Table 6, Illustrative Scenarios (39.39%; 13 studies) constitute the majority of the studies, followed by Controlled Experiments (27.27%; 9 studies), Case Studies (15.15%; 5 studies), Not Applicable (15.15%; 5 studies) and Survey (3.03%; 1 study). There were no quasi-experiment, ethnography and action research papers in our classification.

Note that there are more non-empirical papers than empirical papers. Fifteen papers (45.45%) are concerned in conducting empirical studies (i.e., controlled experiment, case study, and survey) on the applications of authoring tools ITS design. The number of papers that conducted controlled experiments might indicate a maturity in the area about evaluating authoring tools since controlled experiments provide more reliable evidence about specific research hypotheses. However, the number of papers that do not perform any kind of empirical evaluation for their proposal is still high and deserves attention by the community. We will further recall this discussion in "Analysis and Discussion" when analyzing the evidences of ITS authoring tools.

RQ1: Authoring Tools in ITS Components

Results

The purpose of this research question was to identify the main ITS components that have been supported by the use of authoring tools. We categorized these components according to the well-known ITS components (Woolf 2010): domain model, pedagogical model, interface model and student model (see Table 7). Most of the papers use authoring tools to design the *Pedagogical model* of ITS (81.82%; 27 studies) and *Domain model* (75.75%; 25 studies), followed by *Student model* (18.18%; 6 studies)

Table 6 Studies over research methods

Research method	Studies	Freq	%
Illustrative scenario	S05, S13, S14, S16, S17, S21, S22, S25, S26, S27, S30, S31, S32	13	39.39%
Controlled experiment	S01, S04, S08, S11, S12, S15, S19, S20, S29	9	27.27%
Case study	S03, S07, S10, S23, S24	5	15.15%
Survey	S33	1	3.03%
Not applicable	S02, S06, S09, S18, S28	5	15.15%

Table 7 Authoring tools in ITS components

ITS component	Studies	Freq.	%
Pedagogical model	S02, S03, S04, S05, S06, S07, S08, S09, S11, S12, S14, S16, S17, S18, S19, S20, S21, S22, S23, S24, S25, S26, S27, S28, S29, S32, S33	27	81.82%
Domain model	S01, S03, S04, S05, S07, S08, S09, S10, S11, S12, S13, S14, S15, S16, S17, S18, S19, S21, S23, S24, S25, S29, S30, S31, S33	25	75.76%
Student model	S06, S10, S13, S16, S30, S31	6	18.18%
Interface model	S04, S08, S23, S24, S27	5	15.15%

and *Interface model* (15.15%; 5 studies). Note that a study could have met more than one ITS component, thus the sum of the percentages is greater than 100%.

Analysis and Discussion

In summary, results shown in Table 7 indicate that all classic ITS components are covered by the studies. The *Pedagogical model* is addressed by more than 80% of the studies. This result was somewhat expected since users of authoring tools are non-programmer authors that may intend to customize how learning process should take place in the ITS. The *Domain model* component also has a great number of studies (more than 75%). This result is also interesting because it shows that a great part of the studies are delegating or aiding authors in defining what should be learned by students using the designed ITS. Moreover, 19 studies (more than 57% of papers included) met both *Pedagogical model* and *Domain model* in the same paper, indicating the interest of using authoring tools not only to customize learning processes but also to allow the definition of content, problems and so on, according to learning processes defined. It may be worth noting that 6 other papers covered a combination of two different models, e.g., Domain Model and Student Model - 4 papers (S10, S13, S30 and S31), one (S06) covered Pedagogical and Student Model and another one (S27) - Pedagogical and Interface Model.

On the other hand, the use of authoring tools to design *Student models* and *Interface models* are not so much significant in comparison to other ITS components, respectively, 18.18% and 15.15%. For the case of student models, these results are expected since most of the papers are strongly relying only on the artificial intelligence features of tutoring systems to automatic represent student models during instruction, i.e., mainly using mechanisms such as overlay models and Bayesian networks. However, some works still allow authoring of the student model component enabling authors to configure student modeling rules. For instance, S10 presents an authoring tool that allows teachers to author different aspects of the student model for different categories of students. With respect to the authoring of interface models, most of the authoring tools identified in the papers are relying on fixed tutor interfaces, which may not favor authoring of this component. Few works are allowing interface authoring, for example, in CTAT (S04), authors can design and create one or more tutor interfaces specific to the problem types for which the tutor will provide tutoring. Tutor interfaces can be built through drag and drop techniques within an existing interface builder, such as the Flash IDE.

Among all 33 studies, none of them addressed all four classic ITS components. Four papers (S04, S08, S23, S24) met at the same time *Domain model*, *Pedagogical model*, and *Interface model*. One paper (S16) met Domain, Pedagogical, and Student models. These results might suggest an opportunity to use ITS authoring tools to support the design of ITS considering the four main classic components. However, each component has its own function and unique properties which may be more or less amenable to authoring depending on several aspects, for instance, type of ITS, technologies used, needed pedagogical expertise, trade-off choices between usability and flexibility, and so on. In this way, we discuss in "Further Research" a research opportunity aiming to investigate which aspects must be considered when designing ITS authoring tools, including a discussion of what ITS components could be prioritized in the design of authoring tools.

RQ2: ITS Types

Results

The purpose of this research question was to identify the main ITS types that have been developed by the use of authoring tools. The classification of the ITS types was made after the data extraction of the studies, i.e., during the extraction, the ITS type addressed in the paper was identified according to the type explicitly stated by the authors. Next, in the synthesis step, the categories presented in Table 8 were defined according to the distribution of the studies. Note that, even though an ITS could be classified in more than one category, we classified the study in the ITS type that is explicitly argued in the paper. We also defined some categories (i.e., *Content and problem-based* and *Machine and human-based*) according to ITS features discussed in the paper.

Table 8 Authoring tools in ITS types

ITS type	Studies	Freq.	%
Model-tracing/cognitive tutor	S08, S15, S16, S23, S26, S27, S28	7	21.21%
Example-tracing	S03, S04, S12, S20, S21, S32	6	18.18%
Content and problem-based	S02, S10, S17, S19	4	12.12%
Dialog-based	S07, S09, S25	3	9.09%
Constraint-based	S24, S29	2	6.06%
Machine and human-based	S11, S22	2	6.06%
Non-specific	S01, S05, S06, S13, S14, S30, S31, S33, S18	9	27.27%
Total		33	100.00%

As shown in Table 8, the predominant ITS types identified was *Model Tracing/Cognitive Tutor* (21.21%/ 7 studies), followed by *Example-Tracing* (18.18%; 6 studies), *Content and problem-based* (12.12%; 4 studies), and *Dialog-based* (9.09%; 3 studies). *Constraint-based* and *Machine and human-based* have 6.06% (2 studies) each one. In nine studies (27.27%), we could not define a specific ITS type, thus they were categorized as *Non-Specific* type.

Analysis and Discussion

Model-tracing tutors contain a cognitive model of the domain that the tutor uses to check student responses. This model is based on a cognitive psychology theory of problem-solving and learning and is verified by the tutor in each step of the problem-solving process in order to maintain the student in the model path (Blessing et al. 2009). Cognitive tutors are special trademark products that implements model-tracing tutors. They provide a problem-solving environment, including some features such as step-by-step feedback, messages in response to common errors, and instructional hints (Koedinger and Aleven 2007). Once these tutors are very similar, authoring tools targeting them are categorized in the same ITS type. This category includes studies which address the use of authoring tools for designing model-tracing tutors in all four ITS components (see Fig. 2), with an emphasis on domain and pedagogical model.

Example-tracing is also a significant ITS type identified in our results. This category includes studies on the domain, interface, and pedagogical models, but all of them are concerned with the pedagogical model, as seen in Fig. 2. Example-tracing tutors interpret and assess student behavior with reference to generalized examples of

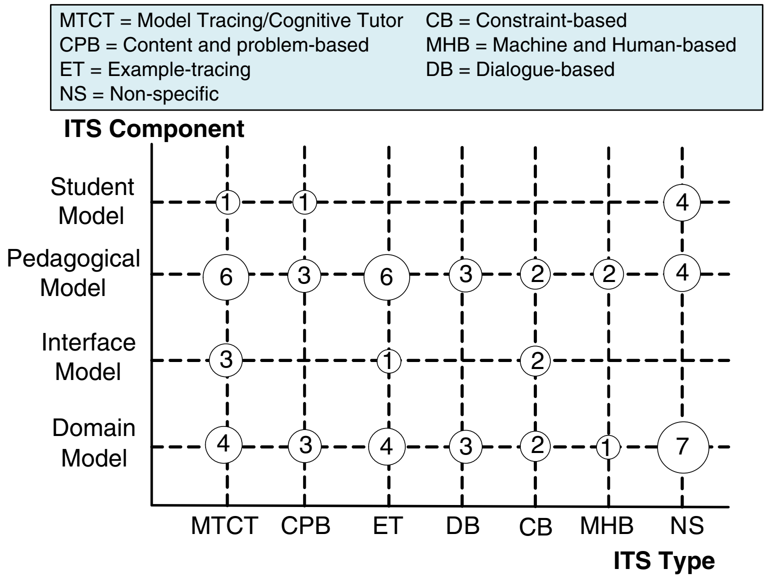

Fig. 2 ITS types over ITS components

problem-solving behavior (Aleven et al. 2009). These examples intend to reduce the technical costs of tutor development by allowing domain experts and cognitive psychologists to build a cognitive model by demonstration rather than by programming a production rule model (MacLellan C.J. et al. 2014).

Content and problem-based category contains five studies and includes papers which mainly relies on authoring tools to author content and learning objects for ITS. Authoring tools categorized in this type basically target ITSs on which students intensively interact with some content and answer problems/tests in the tutor. For example, S02 describes an authoring tool that has been re-built for the Android OS. In this authoring tool, students have the possibility to read the theory offered by the mobile application, interact with it and take tests in order to evaluate his/her level of knowledge. The studies within this category addressed domain, pedagogical and student models.

Dialog-based category is represented by three studies which propose to use authoring tools to design this type of tutor. The studies within this category rely on natural language processing mechanisms to provide a more natural tutoring with studies. These papers address pedagogical and domain models.

Constraint-based tutors are based on Ohlsson's theory of learning from performance errors and are designed to reduce the effort needed to develop a generic model of the domain (Mitrovic et al. 2009). It uses an evaluative model involving constraints defined over a set of pedagogically relevant solutions. The two studies within this category also addressed domain, pedagogical and interface models.

Machine and Human-based category is created to include studies that use authoring tools to design ITS which strongly relies on a machine and human intelligence in a complementary way during the tutoring process. The two studies within this category addressed only the domain and pedagogical ITS components.

The *Non-specific* category includes several tutors with distinct features. Papers are classified into this category if their authoring tool are specific enough to not deserve an own category. For instance, the ASSISTments platform (S18) provides a way to assist student while it assesses them. In this authoring tool, students find out immediately if they had the wrong answer to a problem allowing them to try again right away, whereas, teachers get assessment results in real time, which can be used to plan their next lesson, bring attention to misconceptions, and so on.

Note that two of the most frequent categories presented in Table 7 share a similar tutoring theory, i.e, Anderson's ACT Theory of Cognition (Anderson 1983). *Model-tracing/Cognitive Tutor* and *Example-tracing* ITS types are responsible for almost 40% of the total of papers included in this review. This result might happen due to the popularity of these tutors (i.e., CTAT) that provides several features for authoring these types of ITS for non-programmer authors. In fact, it is likely that the number of authoring tools is simply following the popularity of the ITS types they are targeting. Another result that deserves some attention is that almost 30% of the papers are categorized as *Non-specific*. This result may indicate that there is not a shared understanding in the ITS community of the underlying theories, technologies, and features of ITSs since many researchers are developing authoring tools for designing their own type of tutor.

To aid our analysis, Fig. 2 depicts the number of studies considering the ITS types over the ITS components. Note that the sum of the numbers of studies on specific ITS components exceeds the total number of studies within a specific category because one study could have been addressed by more than one ITS component. As presented in Fig. 2, the *Domain model* and *Pedagogical model* were addressed by all ITS types. The *Interface model* was met by *Model-tracing/Cognitive Tutor* and *Example-tracing* types as well as by the *Constraint-based* ITS type. The *Student model* was addressed by the *Model-tracing/Cognitive Tutor*, *Content and problem-based*, and *Non-Specific*.

These results may also suggest that some ITS types are more amenable than others to target ITS components. For instance, for a paper that presents an authoring tool for example-tracing tutors (e.g., S04), it might be more amenable than dialog-based tutors to author interface models, since the former type of ITS has a flexible architecture that allows personalization of interfaces. In this way, ITS types may constrain the ITS components that authoring tools can address, but it is not clear how it happens and what components and other aspects should be considered when designing authoring tools for specific types of tutors. In "Further Research", we present a research direction on this topic.

RQ3: Features for Aiding Authoring Process

Results

This question intends to identify the features provided by the authoring tools that aid the authoring process. As well as in RQ1, we identified the categories by classifying the studies after the extraction step. A study could also have met more than one feature, thus the sum of the percentages is greater than 100%. As presented in Table 9, we identified 21 categories for the studies. The *Not Applicable* category was defined to classify papers we could not identify any special feature to aid authoring process as well as papers that do not present a new authoring tool, i.e., they use or evaluate authoring tools proposed by other authors.

The most frequent feature identified was the *Define/Give feedback* (27.27%; 9 studies), followed by the *Define problem solutions* (18.18%; 6 studies). Five studies (15.15%) provided the *Authoring by demonstration* feature. The features *Automatic domain model generation* and *View learners' statistics* feature are both presented by four studies each one (12.12%). The *Define behavior graphs*, *Make assignments* and *Define hints* are provided by three studies, each one with 9.09%.

The *Define cognitive model*, *Reuse of learning content/domain model*, *Define students stereotypes*, *Drag and drop interface authoring*, *Mobile authoring* and *Reuse/Export tutor design* features are included by 2 studies (each with 6.06%). We also found several features presented in only one study (3.03%): *Authoring based on learning styles*, *Create class lists*, *Define behavior graphs*, *Define collaboration scripts*, *Define hints*, *Human computation*, and *Reuse of students' profiles*. Four papers (12.12%) were categorized as *Not applicable*.

Table 9 Features for aiding authoring process

ITS component	Feature/Facility	Studies	Freq.	%
Student model	Define students stereotypes	S10, S16	2	6.06%
	Authoring based on learning styles	S06	1	3.03%
	Reuse of students' profiles	S31	1	3.03%
Pedagogical model	Define/Give feedback	S02, S04, S07, S11, S19, S20, S22, S23, S24	9	27.27%
	Define behavior graphs	S04, S21, S32	3	9.09%
	Make assignments	S03, S11, S14	3	9.09%
	Define cognitive model	S08, S32	2	6.06%
	Define collaboration scripts	S26	1	3.03%
Interface model	Drag and drop interface authoring	S04, S26	2	6.06%
Domain model	Define problem solutions	S07, S11, S14, S23, S24, S29	6	18.18%
	Authoring by demonstration	S04, S11, S19, S21, S23	5	15.15%
	Automatic domain model generation	S09, S23, S24, S29	4	12.12%
	Define hints	S04, S15, S19	3	9.09%
	Reuse of learning content/domain model	S01, S13	2	6.06%
	Human computation	S25	1	3.03%
General	View learners' statistics	S02, S03, S30, S31	4	12.12%
	Mobile authoring	S02, S30	2	6.06%
	Reuse/Export tutor design	S13, S30	2	6.06%
	Create class lists	S03	1	3.03%
	Not applicable	S05, S12, S27, S28	4	12.12%

Analysis and Discussion

The results of this research question show a plethora of features that have been considered to aid authoring decision-making process. In the following we depict the function of each feature and present how the feature is supporting the authoring tools presented by the papers. In the end of this section, we discuss these results. As expected, most of these features are related to the Pedagogical and Domain models since most of them are designed to assist authors in defining pedagogical instruction as well as to aid authors to define learning objects to be used in the authored ITS. However, as shown in Table 9, there are also some features related to the Student and Interface models as well as features related to general aspects of authoring tools.

As seen in Table 9, the features *Define students stereotypes*, *Authoring based on learning styles*, and *Reuse of students' profiles* are targeting the student model component. The *Define students stereotypes* feature allows teachers to define student stereotypes by defining characteristics that are used by agents to generate different courseware plans for each stereotype defined (Chakraborty et al. 2010). The *Authoring based on learning styles* feature aids authors to design student models based on a learning style model (i.e., the Felder-Silverman model (Felder and Silverman 1988)) that classifies students according to where they fit on a number of scales pertaining to the ways they receive and process information. The *Reuse of students' profiles* feature let teachers updating a student's profile by interacting with the system, namely pressing buttons, choosing from a drop-down list and picking one from given multiple choices. This feature also offers the possibility to the teachers to register a new student so that s/he is able to make use of the system and learn multiple languages (Virvou and Troussas 2011).

The features *Define/Give feedback*, *Define behavior graphs*, *Make assignments*, *Define cognitive model*, and *Define collaboration strategy* are supporting users to author pedagogical model. The *Define/Give feedback* feature is basically the function that enables authors to define some kind of feedback in the authoring tool to be given to students during instruction. The *Define behavior graphs* feature is frequently used in example-tracing tutors. In this feature, an author can create different ways of solving a problem that is captured as different paths in a behavior graph. Next, the author may generalize the graph to indicate the range of student behavior that the graph stands for (Aleven et al. 2016). The *Make assignments* feature allows authors to create assignments specifically to adjust the students' learning behavior, for instance, S14 enables teachers to make assignments after diagnosing students' learning errors. In order to lower the bar in creating the cognitive model of model-tracing/cognitive tutors, the *Define cognitive model* feature aims to allow non-programmer authors to create the intelligence behind these types of tutors, or at least modify in a meaningful way an already produced cognitive model (Aleven et al. 2009). Finally, using the *Define collaboration scripts* feature, authors can develop collaborative ITSs with embedded collaboration scripts, so that features that support effective collaboration can be intertwined with those that support problem-solving (Olsen et al. 2014).

The features *Define problem solutions*, *Authoring by demonstration*, *Automatic domain model generation*, *Define hints*, *Reuse of learning content/domain model*, and *Human computation* are addressing the Domain model. The function of the *Define problem solutions* feature is to allow authors to enter (before tutor instruction) into the authoring tool, the solution of problems that are given to students. The *Authoring by demonstration* feature is mainly used in a special type of cognitive tutor (e.g., SimStudent (Matsuda et al. 2015)) and enables authors to demonstrate solution steps, and, in the meantime, the authoring tool attempts to induce underlying domain principles by generalizing those worked-out examples. In the *Automatic domain model generation* feature, the authoring tool provides a way to automatically generate elements of the domain model of a tutor. For instance, S24 and S29 use constraint-generation algorithms to produce constraints that verify the syntactic validity of solutions. Similarly to the *Define problem solutions* feature, the *Define Hints* feature enables authors to create and associate hints to problems of ITS. The *Reuse of learning content/domain model* feature supports the reuse of existing learning content from other tutors in the same domain of the tutor being authored. Finally, we found a work (S25) that uses *Human computation*–i.e., a subfield of computer science on which studies represent computationally difficult tasks so that humans will be motivated to work on them (Olney and Cade 2015)–to motivate authors in creating ITS.

We have found only one feature that is supporting users to author the interface model component. As seen in Table 9, S04 and S26 support drag-and-drop interface building to author the interface model of their tutors. We also identify some features that are targeting general aspects of authoring tools. As seen in Table 9, the features *View learners' statistics*, *Reuse/Export tutor design*, *Mobile authoring*, and *Create class lists*. The *View learners' statistics* feature is basically supporting authors to check learners' statistics in the authoring tool, for instance, students' performance in the tutor, interaction with the tutor, and so on. The *Reuse/Export tutor design* feature enables authors to reuse or export previous authoring decisions in a new tutor. This feature saves author time in designing new tutors as well as may favor reuse of already validated tutors. We also identified the *Mobile authoring* feature, which enables authors to design ITS in mobile devices (e.g., S30). Last but not least, using the *Create class lists* feature, teachers can create class lists in order to assign work to an entire class or an individual student and view reports of their students' progress.

One might note that the identified features are much more focused to aid authors in aspects regarding domain (6 features), pedagogical (5 features) and more general purpose (4 features). Whereas, as previously mentioned, few authoring tools have been presenting facilities to enable student (only 3 features) and interface (1 feature) authoring. Particularly, it is possible that researchers are, in general, considering the tradeoff between flexibility and usability to decide whether to incorporate or not features for authoring interface model. We suspect that the extra effort needed to author ITS interfaces has a higher weight over the potential flexibility benefits that could be given to authors. In "Further Research" we present open questions on this topic that might be further investigated.

Another result that deserves to be highlighted is the significant number of features related to general aspects (e.g., *View learners' statistics* and *Reuse/Export tutor design*) of authoring tools. As shown in Table 9, our results suggest that researchers are also interested in providing more powerful authoring tools in order to support authoring beyond traditional ITS components. In "Further Research", we present a research direction on this topic.

Note that there might be a direct relation between the number of papers that address specific features and particular kinds of tutors that are more targeted by authoring tools. For example, as previously presented, example-tracing tutors are addressed by more papers than constraint-based tutors, thus, it is expected a higher frequency in the number of features that are commonly provided by example-tracing tutors (e.g., *Define behavior graphs*).

RQ4: Authoring Technologies

Results

The purpose of this research question was to identify the main technologies used to build authoring tools and the problems these technologies address. We classified the studies according to the type of technology used in the work, after data extraction, by analyzing and grouping the technologies reported in the papers.

As shown in Table 10, most papers (39.39%; 13 studies) use artificial intelligence technologies, concepts or theories to address different kinds of problems within ITS authoring tools (e.g., to support domain knowledge representation, to enable intelligent tutoring, and so on). Moreover, eleven studies (33.33%) use specific tools, platforms, frameworks or plugins to address software engineering problems related to the construction of ITS by using authoring tools, for instance: faster development of tutors, enabling the extensibility of ITS, etc. Three papers (9.09%) use technologies from the distributed systems subarea in order to address interoperability problems regarding ITS. In eight papers (9.09%) we could not identify any specific technology, hence they are in the Non-specific category.

Analysis and Discussion

The results of this research question may be analyzed from the research background on which the technology belongs as well as by identifying particular technologies used in the papers and the problems they are targeting.

As seen in Table 10, 39.39% of the papers use some kind of AI technology, concept or theory. Ontologies are used by the papers S01, S16, S24, S28 and S29 to mainly support domain knowledge representation. These works aid authors in defining the domain model of tutors as well as relying on the reasoning and inference capabilities provided by ontologies to effectively use the domain model during tutoring. Particularly, S16 uses semantic networks, which is more focused on a visual notation to represent knowledge. It also uses intelligent agents arguing that agents can make a good choice to adapt courseware elements to students since they have abilities to learn, personalize and adapt, allowing to manage new situations and providing pedagogically appropriate courseware presentation. Machine learning is also used by four papers, in which S05, S06, and S33 use specific algorithms based on neural networks to address different kinds of problems: S05 uses it to implement emotions recognition in the tutor supported by its authoring tool; S06 and S33 use the technique to provide authors automatic discovered features based on students' performance and interactional patterns (e.g., learning style, students' grades in the course and so on). S23 developed a machine-learning solution, called SimStudent to help novice authors

Table 10 Technologies used to build authoring tools

Technology	Studies	Freq.	%
Tools, platforms, frameworks or plugins	S03, S04, S08, S09, S15, S17, S20, S21, S23, S26, S32	11	33.33%
AI technologies, concepts or theories	S01, S05, S06, S07, S10, S14, S16, S23, S24, S25, S28, S29, S33	13	39.39%
Distributed systems technologies	S02, S04, S30	3	9.09%
Non-specific	S11, S12, S13, S18, S19, S22, S27, S31	8	24.24%

to create cognitive tutors. This tool is integrated into CTAT and helps authors to create an expert model for a cognitive tutor by tutoring it on how to solve problems (Matsuda et al. 2015). S07 and S25 relies on natural language processing techniques for improving the authoring of natural language ITS. For instance, S07 proposes the ConceptGrid authoring tool that intends to check sentence-length natural language answers. S10 supports pedagogical model authoring by using a fuzzy rule-based strategy, allowing authors to configure the rule-base and define the teaching strategy (represented by the rules). The last work in this category (S14) uses hierarchical classification to perform students' diagnosis receiving as input some types of information (e.g., hierarchy of learning errors) from authors.

Regarding the eleven papers addressing software engineering issues of ITS authoring tools, they include works that propose tools (e.g., CTAT), platforms (e.g., Ambre-Add) or frameworks (e.g., GIFT and Tutor Runtime Engine) to support ITS development. Most of the papers included in this category use CTAT (S03, S04, S20, S21, S23, S26, and S32), which is, to best of our knowledge, the most advanced solution reported in the literature to develop different types of ITS (i.e., cognitive and example-tracing tutors). CTAT mainly targets the problem of supporting non-programmers authors to efficiently and cost-effectively develop ITS capable of capturing sophisticated tutoring behaviors that are effective in helping students learn in a wide range of domains Aleven et al. (2016). In other direction, S09 integrates their own authoring tool (called TRADEM) with components of the GIFT framework. It uses the domain module and the engine for providing the pedagogical model from the GIFT framework in order to allow authoring of these components in their tool. S08 uses the Tutor Runtime Engine (TRE), which is a representation of a tutor delivery environment, in order to provide a clear separation between student's interface and the underlying cognitive model that provide the tutoring. This technology intends to enable the integration of third-party interfaces with the tutor-generated by the authoring tool. In a similar manner to what was done with the TRE, S15 uses the Tutor Link plugin to make an existing tool (called xPST) extensible to serve as an intermediary between third-party applications and the xPST Engine. It maps actions in the interface to the proper pieces in the tutor model. It can also display hints and other tutoring information within the application (Gilbert et al. 2015). We also identified a paper (S17) proposing an authoring tool that enables teachers to adapt a specific tool (i.e., AMBRE-add) in order to act on how the ITS automatically adapts itself to the profile of the student.

Only three papers use technologies related to the distributed systems area. These works (S02, S04, and S30) rely on Web Services to enable interoperability between different architectures used in the ITS authoring tool. For instance, S30 uses this technology to allow interoperation between mobile devices used by authors and an ITS architecture. It is worth noting that many ITSs are actually not that tied to AI–i.e., they focus on VanLehn's inner and outer loops (Vanlehn 2006), and may be based on more simple mechanics. This way, we believe that this is a possible reason some papers are not intensively relying on AI technologies to address ITS authoring tools. However, as we could not identify the explicitly use of technologies in eight papers (categorized as Non-Specific papers), we can not say that the papers that are not included in the AI category are indeed not using artificial intelligence in the research.

Considering the results found in this research question, one might note that the use of CTAT is remarkable. CTAT-built tutors have been demonstrated to be robust for use in real educational settings over a wide range of projects (Aleven et al. 2016). Many reasons could explain why CTAT is much more popular than other authoring tools. For example, in order to create example-tracing tutors, it allows authors to easily create graphical user interfaces, to generate behavior graphs, and to aid the deployment of components in structures that executes the example tracer algorithm. Moreover, many researchers are contributing to extending and improve CTAT in several different situations.

Regarding the technologies from the perspective of using or not web technologies, only seven studies (21.21%) strongly rely on web technologies, i.e., ontologies (S01, S24, S28 and S29) and web services (S02, S04 and S30), while 26 studies (78.79%) are not explicitly using web technologies in their works. This result indicates that there is still space for improving existing ITS authoring tools to take advantage of web technologies capabilities, for example: interoperability, distribution, portability and so on. In "Further Research", we present a research direction on this topic.

We can also discuss the results of this research question by analyzing the technologies' subareas over the ITS components and types identified in our previous research questions. As it i shown in the bubble plot in Fig. 3, AI technologies, software solutions and distributed technologies target the domain, pedagogical, and interface models. There is an emphasis on the first two components evidenced by a higher number os papers targeting them (see Table 10). Note that our results suggest that the papers related to software solutions are not addressing the student model component. This result is interesting since it is expected that authoring tools rely on the automatic student modeling representation of intelligent tutors (on the learner's side) to target this component.

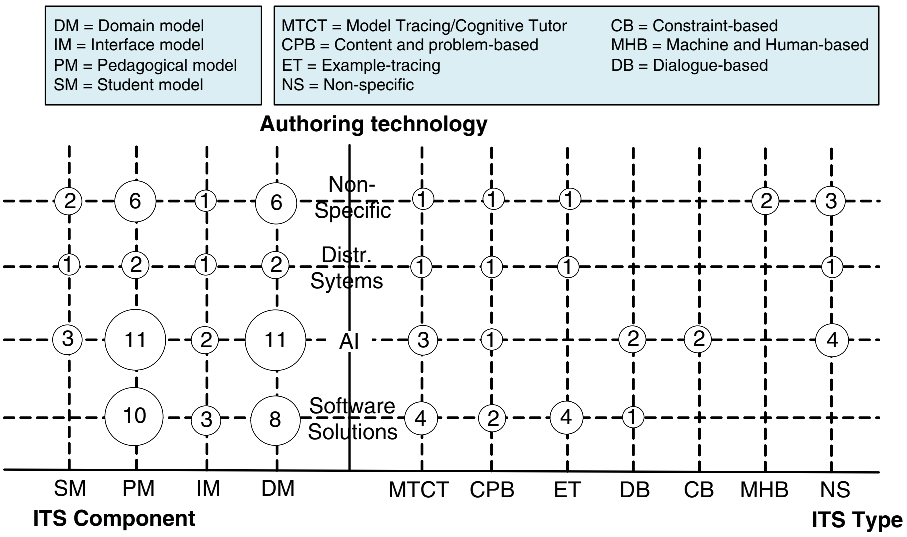

Fig. 3 ITS components and types over technologies subareas

As shown in Fig. 3, our results suggest that authoring tools targeting example-tracing tutors are more concerned with providing software solutions to construct tutors than relying on specific artificial intelligence techniques. This result does not imply that these authoring tools are not using artificial intelligence. These results also suggest that the dialog-based ITS is the type of ITS that relies more on artificial intelligence, which can be explained by the fact that these kinds of tutors strongly depend on AI techniques, for example, natural language processing. Moreover, as seen in Fig. 3, our results indicate that constraint-based tutors are mainly supported by AI techniques. This result may be correlated with the results of our previous research question since we could identify a feature (i.e., Automatic domain model generation) dependent on AI algorithms that are used by the two papers targeting this type of ITS. One might also note in Fig. 3 that the authoring tools, addressing machine and human-based tutors are only using non specific technologies, which suggests these tools are investigating non-conventional technologies to contribute to the development of such type of tutor. In "Further Research", we point out further research directions in this topic.

RQ5: Authoring Time

Results

The purpose of this research question was to identify how the studies are distributed concerning when they are using authoring tools to support ITS. Our intention is to investigate how the studies included in this review are targeting authoring tools considering the ITS "life-cycle". We mean that in addition to being merely used to develop ITS, authoring tools might be strategic to manage tutoring systems at other stages since it is by using these tools that tutors are generated. We argue that, beyond using ITS authoring tools to develop tutor before students use it, authoring tools could also enable authors to act, for example, at critic learning situations in order to avoid students dropping out the tutor, to allow reconfiguration of tutors after a tutoring cycle to improve the tutor, and so on.

Thus, we defined three categories: 1) Pre-instruction (before putting students to use the ITS); 2) During instruction (while students are interacting with the tutor), and 3) Post-instruction (after finishing a cycle of tutoring with students). Table 11 presents the distribution of the studies within these categories. Similarly to other

Table 11 When the authoring process occurs

Authoring timing	Studies	Freq.	%
Pre-instruction	S01, S02, S03, S04, S05, S06, S07, S08, S09, S10, S11, S12, S13, S14, S15, S16, S17, S18, S19, S20, S21, S22, S23, S24, S25, S26, S27, S28, S29, S30, S31, S32, S33	33	100%
During instruction	S11, S18, S22, S30	4	12.12%
Post-instruction	S11	1	3.03%

research questions, a study could also have met more than one category, thus the sum of the percentages is greater than 100%. As seen in Table 11, all studies (100%; 33 studies) use authoring tools during development time (Pre-instruction), followed by studies (15.15%; 5 studies) that use authoring tools during tutoring with students (During instruction) and by studies (6.06%; 2 studies) that use authoring tools after students finish their use of the ITS (Post-instruction).

Analysis and Discussion

Our results suggest that all studies use ITS authoring tools before students begin to use the tutor. This result is not surprising since the main goal of current authoring tools is enabling the design of ITS, which happens before allowing students to interact with the produced tutor. However, as shown in Table 11, we identified that there are some authoring tools that also allow authors to intervene in the tutoring process. S18 presents the ASSISTments tool, which intends to put teachers in charge of the instruction by providing student-level data on assignments to them as well as giving immediate feedback to students. This tool may rely on teachers to differentiate instruction between students. For example, teachers could give extra practice on the skill to students who got the problem wrong (Heffernan and Heffernan 2014). Additionally, S22 intends to give teachers "pedagogical ownership" of the educational technology, which means that the owner understands the content, the delivery mechanism, and the pedagogy underpinning it. Using this work, the owner/teacher can deliver the content to learners, reflect on the effectiveness of that content and adapt it to better suit the learning needs of students at the time students are interacting with the tutor (Marcus et al. 2010). S30 presents a mobile authoring tool that allows instructors to create and administer the domain knowledge model of the existent ITS and namely the domain to be taught and the tests along with the characteristics of students (Troussas et al. 2014).

We also identified one work that is targeting during and post-instruction ITS authoring, besides addressing pre-instruction authoring as well. S11 provides authoring during instruction by proposing a virtual teaching assistant to share teacher tutoring load in helping students practice program tracing. This assistant adopts a mechanism (which include assigning a problem for students to solve, diagnosing student solutions and offering feedback to help students, and providing the correct solution) for enabling human intelligence and machine intelligence to share tutoring tasks to reduce the system requirement. This study also provides a second mechanism aiming to provide post-instruction authoring. It applies machine intelligence (i.e., interaction data reuse approach), including records of error situations and teacher hints, and searches the records for similar errors and hints, to reuse human intelligence (teacher-generated hints for specific error situations) to provide program-specific hints (Chou et al. 2011).

With respect to these results, one might note that the papers that are targeting during and post-instruction authoring are relying more on the human intelligence of teachers than on artificial intelligence. These results are very interesting and point to a current discussion in the ITS research. Hence, in order to contribute to this discussion, it would be important to investigate what and when ITS aspects (i.e., components, features, and son) should be delegated to artificial and/or human intelligence. In "Further Research", we point out a research direction in this topic.

Furthermore, we may also analyze when authoring occurs according to the ITS components and types (Fig. 4). As previously mentioned, the *Pre-instruction* category was addressed by all papers and, hence, by all ITS components (left part) and ITS types (right part) that we identified. The authoring tools that fall into *During instruction* targeted three ITS components (i.e., *Domain Model*, *Pedagogical Model*, and *Student Model*) as well as only one specific ITS types: *Machine and Human-based*. These results may confirm how the authoring tools are supporting authoring during tutoring process, for instance, enabling teachers to intervene in the pedagogical tutoring, updating content in the tutor, and so on. As also expected, this result also suggests that the machine and human-based tutors rely on authors during the instruction. Lastly, the *Post-instruction* authoring tools are targeting the *Domain* and *Pedagogical* models as well as the *Machine and Human-based* type of ITS. These results may be explained for the same reasons of the during instruction category.

Moreover, the *Interface model* has been only authored in Pre-instruction time, which may indicate that researchers are not interested (or believing) in the potential of interface authoring after a course has begun. Maybe because it might cause confusion in the students if graphical interfaces change after starting to interact with the tutor. Finally, one might note that the *Machine and Human-based* ITS type has been used in all three authoring time categories. This result may suggest the empowerment of authors, relying on human intelligence to design the tutoring process in different authoring time frames–we discuss it in "Further Research".

RQ6: Evidences of Authoring Tools Benefits

The purpose of this research question was to gather and classify evidence to state how the use of authoring tools benefits the design of ITS. In order to answer it, we

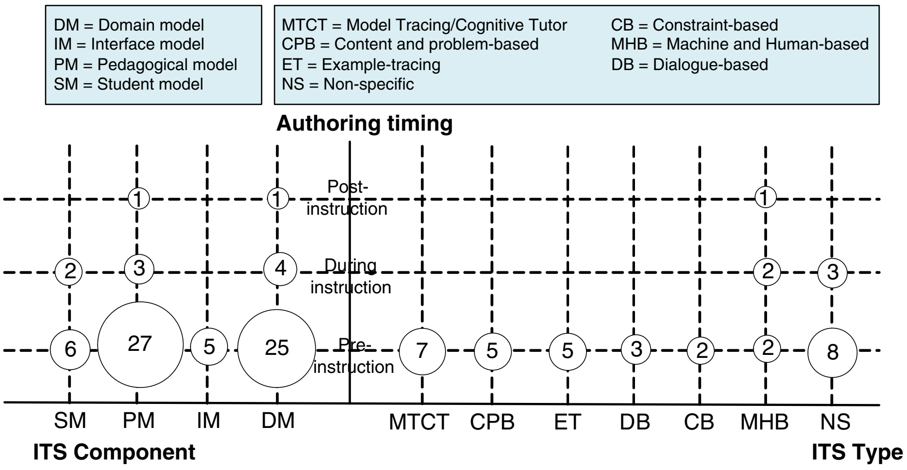

Fig. 4 ITS models and types over authoring timing

synthesized ("Results") and analyzed ("Analysis and Discussion") the research method of the studies along with the studies' indication of positive or negative results of the use of authoring tools to effectively design ITS. Then, in "Empirical Benefits and Metrics Investigated"), we identify the benefits and metrics reported by the empirical studies and discuss each work according to its contribution, empirical method and metric used.

Results

The classification of the studies was defined according to the presence or absence of empirical evaluation in the study and by the positive or negative indication that using authoring tools benefits ITS design. The defined categories are: positive with empirical evaluation, positive without empirical evaluation, negative with empirical evaluation and negative without empirical evaluation (see Table 12). However, we did not find any study reporting negative evidence of using authoring tools to design ITS, thus negative categories are not presented in Table 12.

In fact, the distribution of positive reports about the use of authoring tools to design ITS is quite balanced with few more non-empirical papers. As previously mentioned, the *Positive argumentation* (i.e., papers without empirical evaluation, but stating a positive argumentation about the benefits of using ITS authoring tools) category includes 18 studies (54.55%), whereas the *Positive with empirical evaluation* category includes 15 studies (45.45%).

Analysis and Discussion

The results of this research question are of utmost importance to verify if there is real evidence to state that the use of authoring tools have been effectively benefiting ITS design.

These results show that more than half of the studies provide only a positive argumentation about the benefits of using authoring tools to design ITS. They did not present any empirical evidence, but rather presented some line of argumentation and/or illustrated their contributions using small examples. However, a significant

Table 12 Evidences of using ITS authoring tools

Evidences	Studies			Freq.	%
	Controlled exp.	Case study	Survey		
Positive with empirical evaluation	S01, S04, S08, S11, S12, S15, S19, S20, S29	S03, S07, S10, S23, S24	S33	15	45.45%
Positive argumentation	S02, S05, S06, S09, S13, S14, S16, S17, S18, S21, S22, S25, S26, S27, S28, S30, S31, S32			18	54.55%

amount of studies presented empirical evidence of such benefits. In the following, we analyze and discuss these specific studies from two dimensions: (i) type of empirical evidence, determined by the research method used in the study and (ii) context on which the study was conducted, i.e., academic or industry.

As shown in Table 12, most of these empirical studies used controlled experiments (9 papers), followed by case studies (5 papers) and a survey (1 paper). Moreover, examining the studies (S01, S04, S08, S11, S12, S15, S19, S20, S29) which performed controlled experiments, we can observe that all of them reported evidence from an academic context. By analyzing the empirical evidence through the use of case studies, it can be observed that 1 study is from industry (S03) and four studies (S07, S10, S23, and S24) are academic. The survey study was also conducted in academic settings.

The empirical studies can also be analyzed over the ITS components that they address. Figure 5 presents a bubble plot chart distributed over three dimensions: study settings, research method, and ITS components. The left part in Fig. 5 denotes the relationship between empirical studies' settings and the ITS component that they address. The number in a bubble represents the number of studies on a specific setting addressing a certain component. By contrast, the right part of Fig. 5 denotes the relationship between studies' research methods and ITS components. The number in a bubble represents the number of studies on a specific empirical research method in a certain component. Note that, the sum of the numbers of studies on a specific research method or settings category exceeds the total number of studies within a specific category because the same study could meet more than one ITS component.

On the other hand, one might note that no papers report negative evidence of using authoring tools to design ITS. This result may suggest that authoring tools are effectively benefiting the design of ITS for non-programmer authors. However, this result is somewhat expected since researchers usually publish papers that produce (or expect to produce) positive effects for addressing theirs research problems.

By analyzing Fig. 5, we can draw some conclusions about the empirical studies identified in this review: (i) the Domain model is addressed by controlled experiments, case studies and a survey, with more emphasis on academic studies; (ii) the

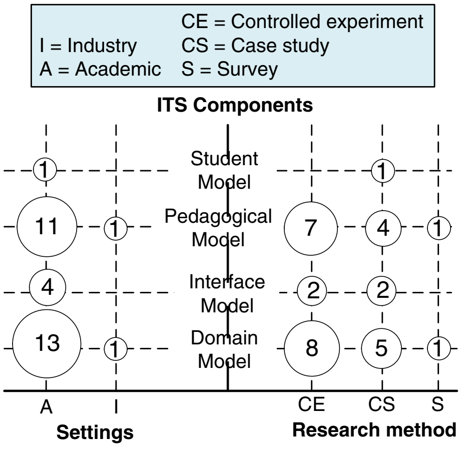

Fig. 5 Research method and context of empirical studies over ITS components

Pedagogical model is more addressed by controlled experiments than by case studies and is also addressed by a survey study. Moreover, the majority of studies in this component are academic; (iii) the Interface model is addressed by case studies and controlled experiments, no survey met this component. In addition, the settings of the papers in this phase are in great part academic; and (iv) the Student model is only addressed by one academic case study.

Empirical Benefits and Metrics Investigated

In this section, we describe each one of the fifteen works in terms of contribution, the empirical method used and strength of evidence reported. In the end of this section, we summarize the metrics investigated by the ITS authoring tool proposed in the studies.

S01 (Abbas et al. 2014) aims to facilitate authoring of learning contents for instructional and constructive teaching strategies. As such, they allow authoring of content through either automated or semi-automated way. In an automated way, authors use ontologies to perform reasoning over domain knowledge and infer new learning contents. To evaluate their contributions (i.e., the MySekolah framework), they conduct studies considering four metrics: *correctness*, *efficiency*, *effectiveness*, and *usability*. *Correctness* is automatically verified for checking the domain model authored through the use of ontology reasoners (e.g., Fact++ reasoner). The contents authored from MySekolah are evaluated with respect to *efficiency* (time for creating learning content). All contents produced were manually evaluated by the teacher at the end of the operational session. The sample is fourteen kindergarten female teachers and twenty-one students (age between 4 and 6) from five schools in Malaysia. A controlled experiment was conducted comparing the time difference between the content creation activity done through traditional way (delivery of pre-authored learning contents) and with the help of MySekolah. The results suggest that MySekolah required a substantially shorter time for the creation of learning content (not statistically verified) than the traditional approach. The *effectiveness* of MySekolah was performed by evaluating the auto-generated contents by MySekolah against a single user profile. Authors suggest that the auto-generated content makes the cognitive skills tutoring more effective, however, the evidence is not so strength, since authors do not run statistical tests to provide more reliable evidence about these results. Finally, the *usability* of MySekolah is evaluated according to the level of agreement and disagreement of teachers on eight Likert type questions. The results show that participants agreement is mostly toward "Satisfied" or "Strongly Satisfied" with respect to the software interface, easiness to generate learning content manually and automatically, time required to learn MySekolah and the time required to author learning contents.

S03 (Aleven et al. 2009) supports the development of example-tracing tutors. Authors report an experience of using ITS authoring tools (i.e., using CTAT's programming-by-demonstration approach) on a large scale project. They have created an open-access Web site, called Mathtutor, where middle school students can learn mathematics with guidance from ITS. The Mathtutor project intends to address authoring of ITS considering increase in *simplicity* and in *cost-effectiveness* as well as supporting sophisticated tutoring behaviors such as multiple solution strategies, dependencies among problem steps, and multiple interpretations of student behavior and large-scale development, iterative refinement, and maintenance of ITS. Although not presenting many pieces of evidence about the benefits of using CTAT's programming-by-demonstration approach to build Mathtutor, authors state that their experience with the project give confidence that this authoring approach is appropriate to support the range of tutoring behaviors needed for the middle school math tutors, and it will support iterative content development and refinement on a large scale.

Likewise S03, S04 (Aleven et al. 2016) supports the development of example-tracing tutors through the use of authoring tools. To investigate the *effectiveness* of example-tracing tutors, authors look at example-tracing tutors built with CTAT since their former work (Aleven et al. 2009). As such, authors review 18 example-tracing tutors that were used in real educational settings, many with statistically significant pre/post *learning gains*. Authors clustered the reviewed tutors based on aspects of their pedagogy (e.g., problem-solving tutors, tutors that use worked examples or erroneous examples, tutors that emphasize the use of interactive graphical representations, and tutors that use pedagogical approaches other than standard tutored problem solving). Regarding the problem-solving tutors, evidence on *learning gains* is provided the *Genetics* tutors (pretest-to-posttest averaged in almost two letter grades) and a basic equation solving tutor, called *Lynnette*, which led to pre/post gains in basic equation solving skill with medium to large effect sizes ($d = .69$, $d = 1.65$, and $d = 1.17$). With respect to the tutors that used worked or erroneous examples, the *Stoichiometry* tutor that used this pedagogy strategy is more efficient in terms of time (d ranging between 1.76 and 3.31) and mental effort (d ranging between .89 and 1.04) than normal problem-solving tutors. The *AdaptErrEx* project also used erroneous examples to help students learn decimals and performed significantly better on a delayed test ($d = .62$ and $d = .33$) than students in a tutored problem-solving condition with explanation steps. Finally, using the *Proportional Reasoning* example-tracing tutor in the worked example condition, learners took less time and scored higher on the post-test than learners in two other conditions. Regarding the tutors that emphasize the use of interactive graphical representation, evidence on learning gains from three example-tracing tutors is reported by Aleven et al. (2016). The *Fractions* tutor presents learning gains, up from the pre-test, were $d = .40$ for the immediate post-test and $d = .60$ for a delayed post-test. Moreover, the *Chem* tutor led to large learning gains in a field study with 74 undergraduate students enrolled in an introductory course for science majors ($d = 1.44$) and in a lab experiment with 117 undergraduates ($d = .78$). Using the *RedBlackTree* tutor, two small evaluation studies led to large learning gains ($d = 1.66$ and $d = 3.06$). A non-standard tutored problem-solving tutor also present evidence about the learning gains of using example-tracing tutors built with CTAT, a new project with *Fractions* tutor. The new project investigates whether an ITS can be made more effective by adaptively targeting a broader range of learning mechanisms (i.e., grounded in the Knowledge-Learning-Instruction (KLI) framework (Koedinger et al. 2012), which links cognitive theory and instructional design) than ITS typically do. In a classroom study with

1068 fourth and fifth-grade students from 12 schools, the tutor led to significant pre- to post-test learning gains ($d = .47$).

S07 (Blessing et al. 2015) presents ConceptGrid which provides a template-style approach to check natural language responses by students using a model-tracing style intelligent. To evaluate the approach, ConceptGrid is integrated within the xPST authoring tool and two controlled experiments are conducted. The first experiment aimed to evaluate the tutor's *effectiveness* overall in the domain of statistics and the second one investigated the *usability* of the approach by non-programmers and non-cognitive scientists. Authors argue that both experiments show the effectiveness of the ConceptGrid approach. In the first experiment, the creator of ConceptGrid enjoyed an accuracy rate above .97 in checking answers given in real-world problem scenarios. The second experiment provided evidence as to the worthiness of the approach, as not only did an intermediate level user successfully use the tool but non-programmers and non-cognitive scientists also quickly learned how to use ConceptGrid in order to check answers given on a typical homework assignment.

S08 (Blessing et al. 2009) supports the development of model-tracing cognitive tutors through the use of authoring tools. The tutor produced with such authoring tool is embedded with a cognitive model that has traditionally been difficult to create. Thus, to evaluate the authoring tool, Blessing et al. (2009) conducted a controlled experiment on which participants (17 students) created a cognitive model using the Cognitive Model SDK with respect to three metrics: *capacity of creating a runnable cognitive model*, *time to create the model*, and *quality of the cognitive model*. For the first metric, authors state that the majority of participants (77%) created a usable cognitive model with minimal instruction (less than 1 hour). Moreover, participants created their models relatively quickly, in under 8 hours on average. According to the authors, the quality of models created by participants was very similar to a number of measures.

S10 (Chakraborty et al. 2010) aims to support authoring of different components of intelligent tutoring systems. To evaluate the authoring tool, authors conduct a case study with four teachers using the system proposed with respect to the *time taken* by teachers to perform tasks in the system. Authors state that the results show that the most time-consuming part of the whole process is authoring contents, which is compensated by the feature of reusing existing contents from other sources.

S11 (Chou et al. 2011) proposes the development of virtual teaching assistant by combining machine and human-based intelligence to help students practice program tracing. As such, two mechanisms are used: the first mechanism applies machine intelligence to extend human intelligence (teacher answers) to evaluate the correctness of student program tracing answers, to locate student errors, and to generate hints to indicate errors and the second one applies machine intelligence to reuse human intelligence (previous hints that the teacher gave to other students in a similar error situation) to provide program-specific hints. To evaluate their proposal, authors conducted two controlled experiments (with 85 and 64 participants) with respect to *teacher tutoring load*. The results suggest that the two mechanisms significantly reduce teacher load. The system effectively helped most (above 89%) students find and correct their errors in program tracing. After the teacher provided the correct answer, the system-generated error-indicating hints helped students to correct above half of the errors. Finally, each teacher-generated hint was reused averagely three times by the second mechanism.

Similarly to S03 and S04, S12 (Devasani et al. 2012) aims to investigate how authoring tools could support the development of example-tracing tutors. In the study, authors evaluate two intelligent tutoring system authoring tool paradigms, graphical user interface-based and text-based by taking the examples of CTAT and xPST, respectively, and in two domains, statistics, and geometry. Sixteen tutor-authors divided into 2 groups (programmers and non-programmers) participated in the controlled experiment considering the following metrics: *quality score for the authored tutor* and *time to author the tutor* of the authoring tool. Authors state that they cannot claim that their results are statistically significant, partially because of the small sample size. Nevertheless, examining how well the finished tutors instructed learners, no differences were observed between the tutors produced by the programmer and non-programmers, with all but 2 tutors (one xPST tutor and one CTAT tutor, both done by non-programmers) being judged at the highest caliber. Both tools allowed non-programmers to create *effective* instruction. Authors also argue that making judgments concerning *time* is speculative. However, the CTAT authors spent less time constructing their tutors in the beginning, but by the third tutor problem, there were no significant time differences, with all but one participant spending no more than 25 min per problem, regardless of the tool.

S15 (Gilbert et al. 2015) proposes the Extensible Problem Specific Tutor (xPST), which allows non-programmer authors to create an intelligent tutoring system that provides instruction related to a model-tracing tutor. The xPST authoring system was developed to allow tutoring on any interface. It allows an author to create a model for a particular problem instance by creating hints and other tutoring aspects while the author manipulates the interface. To evaluate the authoring system, three studies were described in the paper. The first one investigated the use of an early version of the Web xPST with ten students in an introductory human-computer interaction graduate class. The results showed that the 10 participants produced 26 tutors. To evaluate the tutors, authors scored tutors on a 5-point scale that indicated the quality of the tutor. Eighteen (69%) received a score of either 4 or 5, indicating that the tutor went above the minimum needed to scaffold the learner through the task. Seven tutors met the minimum by receiving a score of 3, and only 1 model was deficient in tutoring. In the second study (with the final web xPST system), five non-programmers developed a set of problems to be used in a college-level statistics course. According to (Gilbert et al. 2015), authors of this study averaged 28.57 h logged on to the system across the month for the total *time* (with a range of 18.45 to 36.85 h), and a mean of 7.37 h editing the xPST Instruction File (a range of 4.87 to 9.48 h). By the end of the experiment, participants spent less than 45 minutes authoring a problem in total, with less than 18 minutes of that time spent writing xPSTcode. In addition, the participant authors filled out two separate surveys and ranked Web xPST tutor as *powerful* and *easy to use*, and not *being frustrated* in its use. The third study refers to the work previously described (S12).

S19 (Lane et al. 2015) presents the Situated Pedagogical Authoring (SitPed) system that seeks to allow non-programmer authors to create ITS content. SitPed is designed for eliciting ITS content from subject-matter experts and the implementation focuses on problem-solving through conversation. To evaluate the system, a two-phase study was conducted with subject-matter experts (phase 1), focusing on the authoring of content, and with college students (phase 2) who had no experience with a selected domain, focusing on how well students learn from it. In the first phase, 11 domain experts with academic training and practical experience in the domain authored ITS content for one scenario across three authoring conditions (full SitPed, SitPed Lite and spreadsheet). Due to the low number of authors participating in the first phase, authors provide only descriptive data. In summary, authors using spreadsheet created more tutoring messages than using both versions of SitPed. However, authors in SitPed conditions created longer messages. In phase 2, participants (i.e., students) were randomly assigned to one of three groups and the data from 54 participants is used. The only significant difference found between the groups favored SitPed over the spreadsheet group (mean gains of .135 to .054, F(2,52)=3.635, p=.033) with respect to true/false responses.

S20 (MacLellan C.J. et al. 2014) also intends to investigate the use of authoring tools to support the development of model-tracing/cognitive tutors. As such, authors conduct an experiment to compare a tutor built with SimStudent–a CTAT module that tries to bridge the gap between example-tracing and cognitive tutors by learning production rule models from demonstrations and problem-solving feedback–and with example-tracing using CTAT. The comparison considers the authoring of an algebra tutor and two metrics are analyzed: average *authoring time* and *quality* of the models authored. For the first metric, authoring using the SimStudent approach may yield improved authoring efficiency over the standard Example-Tracing approach. This efficiency gain was because SimStudent only required feedback, instead of demonstrations, when it had applicable production rules. Providing feedback (2.4 s) takes much less time than performing a demonstration (8.8 s for Example Tracing and 10.4 s for SimStudent), so this results in a decrease in authoring time. Authors also argue that by the end of tutor authoring both approaches had equivalent model quality. However, results suggest that SimStudent produces a model that is more general than the specific demonstrations, bridging the gap between an Example-Tracing Tutor and a full-edged Cognitive Tutor.

S23 (Matsuda et al. 2015) investigates the use authoring tools to author the expert model of cognitive tutors. Authors conduct evaluation studies (simulation study and case study) to investigate which of two authoring strategies used in SimStudent (i.e., authoring by tutoring or authoring by demonstration) better facilitates authoring. In the authoring by tutoring strategy, the author interactively tutors SimStudent by posing problems to SimStudent, providing feedback on the steps performed by SimStudent, and also demonstrating steps as a response to SimStudent's hint requests when SimStudent cannot perform steps correctly. In the authoring by demonstration, the author demonstrates solution steps, and SimStudent attempts to induce underlying domain principles by generalizing those worked-out examples. The results reported suggest that for the purpose of authoring an expert model, authoring by tutoring is a better alternative than authoring by demonstration. Results showed that when authoring an expert model with SimStudent, authoring by tutoring better facilitates authoring both in the *quality of the expert model generated* and the *time spent for authoring*. The study also showed that the benefit of authoring by tutoring in creating a high-quality expert model is largely due to the feedback on the correctness of learned predictions applied proactively when solving problems.

S24 (Mitrovic et al. 2009) investigates the use of authoring tools to develop constraint-based tutors, including the authoring of learning content of these tutors. As such, ASPIRE (Authoring Software Platform for Intelligent Resources in Education) is presented, which is an authoring and deployment environment for constraint-based ITSs. ASPIRE consists of the authoring server (ASPIRE-Author), which enables domain experts to develop new constraint-based tutors, and a tutoring server (ASPIRE-Tutor), which deploys the developed systems. ASPIRE-Author supports the authoring of the domain model as well as examples of problems and their solutions. ASPIRE allows the development of ITSs by generating constraints automatically, thus not requiring programming and AI expertise from authors. As discussed in the paper, evaluations showed that the constraint generation algorithms used were capable of generating 90% of constraints needed. Mitrovic et al. (2009) also report some former experiences with the use of ASPIRE to develop tutors, for instance, the development of a Capital Investment decision-making tutor. To evaluate the benefits of the ASPIRE platform, an experiment of an ITS authored using such platform in the domain of Capital Investment decision-making with 21 students participating of a summer school course was conducted. The course had two scheduled tutorial streams, and one of them was randomly selected to serve as the experimental group, while the other served as the control group. Both groups improved on the post-test, with the control group having a significant ($p<0.04$), and the experimental group a marginally significant improvement ($p=0.066$) on *learning gains*.

S29 (Suraweera et al. 2010) aims to use authoring tools to reduce the knowledge acquisition for constraint-based tutors and to enable non-programming domain experts to build the domain models required for ITS. As such, an authoring system, called Constraint Authoring System (CAS), was developed (using ontologies and machine learning) in order to be capable of producing a domain model with the assistance of a domain expert. To evaluate the *effectiveness* of the system, three studies were conducted. According to Suraweera et al. (2010), the first study verified whether the creation of an ontology is beneficial in the process of manually composing constraints. The results revealed that the task of composing ontologies indeed assisted in the process of developing constraint bases. The effectiveness of CAS's constraint generation was then evaluated in a study that generated domain models for database modeling and data normalization. Analysis of the generated constraints revealed that they accounted for over 90% of the constraints required for the domain. The final evaluation study evaluated CAS's effectiveness in generating constraints with the assistance of novice 13 ITS authors. Results have shown that CAS is also effective in generating constraints when assisted by only novice ITS authors; the achievement by novice ITS authors producing constraints in a time similar (1.1 h per constraint) to the time reported (1.3 h per constraint) by a former study is significant. Moreover, under these conditions, it still produced constraint sets that were over 90% complete.

Finally, S33 (Zatarian-Cabada and Barrón-Estrada 2011) supports the authoring of content in the context of ITS. In the work, authors have designed and implemented a software tool (EDUCA) to create adaptive learning material in a Web 2.0 collaborative learning environment. The material is initially created by a tutor/instructor and later maintained and updated by the user/learner community to each individual course. To evaluate the authoring of content using the authoring tool EDUCA, a survey was applied to the 30 students that participated in the study, asking the learners to evaluate the software tool on the Likert scale according to their level of agreement or disagreement. The results indicate that most of the group in the workshop "agree" or "strongly agree" with respect to the interfaces, *easiness to generate an intelligent tutor*, *learning time for using the tool*, *time to produce an intelligent tutor* (1 h) and the course organization.

Figure 6 presents the metrics investigated by these studies. Seven main metrics have been considered by the proposals that conduct some empirical study to evaluate their ITS authoring tools. They are: "Efficiency and time to author" (S01, S08, S10, S12, S15, S20, S23, S29, and S33), "Quality of authored artifact" (S01, S08, S12, S15, S19, S20, and S23), "Effectiveness of authored tutors" (S01, S03, S04, S07, S19, and S24), "Usability of the authoring tool" (S01, S03, S07, S15, and S33), "Capacity of authoring by authors" (S08 and S29), "Powerfulness of the authoring tool" (S15), and "Teacher tutoring load" (S11).

In summary, we can highlight that the results found in this work suggest that there is empirical evidence to state that authoring tools, indeed, benefit the development of ITS in several ways. However, the strength of the pieces of evidence is somewhat limited to the context (e.g., ITS types, features provided, technologies used, when the authoring happens, and so on) on which the studies were performed. Thus, in "Summary of the Findings", we discuss these results linking our findings with further research directions.

Discussion

In this section, we present and discuss the main findings of this review. Next, we discuss further research directions on the use of authoring tools to design ITS. In the end of this section, we discuss the threats to the validity of this review.

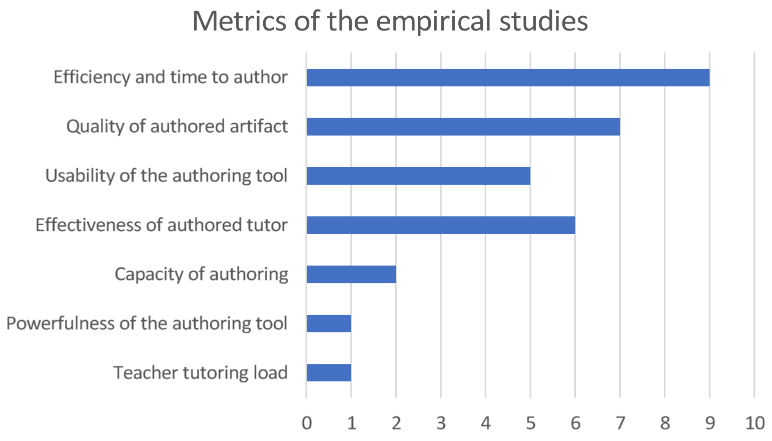

Fig. 6 Metrics investigated by the empirical studies

Summary of the Findings

In the beginning of this review, we described three main unanswered questions stated by Murray (2003) in his well-known review on ITS authoring tools. Taking those questions and the results of this systematic review, in this section we discuss the current state of the art in authoring tools.

Murray's first question is related to the extent to which the difficult task of modeling ITS could be scaffolded. According to the results found in this systematic review, we argue that this question is currently solved. Throughout answering the research questions of this systematic review, we have presented several ITS authoring tools (e.g., CTAT, xPST, ASPIRE, and so on) that indeed allow modeling and development by non-technical authors (e.g., teachers, non-programmers, or non-domain experts) of several effective ITS.

The second question described by Murray (2003) is the degree to which identifying instructional situations that can be embodied in special purpose authoring shells that are both specific enough to make authoring template-based, yet general enough to be attractive to many educators. Woolf (2010) also highlights the complexity to build authoring tools considering these design tradeoffs (specificity vs. generality of tutors). The results of our review have shown that there are effective tutors (i.e., in terms of increase in learning gains) of several types (e.g., example-tracing, cognitive tutor, constraint-based) and level of generality (i.e., domain-independent, such as ASPIRE) that were produced by different kinds of non-technical authors using authoring tools relying on several features and technologies. Moreover, as stated by Woolf (2010), the primary goal of authoring tools is to simplify the process of building tutors. As such, ITS authoring tools community should keep in mind that the decision on which level of generality to focus might not come at the expense of higher complexity and decrease in its usability. Our results have shown that there are few ITS authoring tools that are not so complex and time-consuming to use. Thus, we believe that Murray's question is still unanswered. A good starting point to support the design decision-making for building authoring tools might be the conceptual framework proposed by Woolf (2010) on which the development team of authoring tools must identify the tutors to be produced (e.g., considering the type of ITS and level of generality), identify the authors (i.e., if they are teachers, domain experts, software engineers), and identify the students (i.e., context and level) that will use the tutor.

In his third question, Murray (2003) raises a larger question of whether intelligent tutoring systems would ever be in demand enough to warrant the effort of building authoring tools for them. According to the results reported in this review and recent literature on ITS, our answer would be "there is enough demand for ITS to vindicate the effort of building authoring tools". ITS has been broadly used in a commercial scale and in schools in many places (e.g., USA, Europe, Japan, Brazil, and other countries) and there is an increasing demand for tutors in different domains and educational levels. For instance, our results reported that at least eighteen example-tracing tutors, many of them with evidence for increasing learning gains, were built with CTAT for several domains (e.g., Genetics, Stoichiometry, Fractions and so on). Without CTAT, the cost of building those tutors would be much higher. Note that these 18 tutors described by Aleven et al. (2016) are classified in only one of the ITS types we found in this review. As previously presented, other intelligent tutoring systems built by using other authoring tools are also highly demanded.

The results of this review showed that it is possible to create tutors using authoring tools that benefit students' learning. For instance, CTAT, SitPed, ASPIRE, among other authoring tools, presented evidence about the learning gains provided by tutors built with them. In his review, Murray (2003) stated that in that time there were several authoring tools at or near commercial quality. In fact, since his review, authoring tools have matured substantially; perhaps, the best example of the commercial quality found in this review is CTAT. Actually, it is noteworthy to mention that the evidence on the effectiveness provided by the tutors built with CTAT is remarkable. However, although there are effective tutors produced by the authoring tools reported in this review, some of them with commercial quality, there is not so enough evidence about the costs to build and the simplicity to use them. We believe these factors would be decisive to influence the adoption of authoring tools in market places. Moreover, higher levels of simplicity and usability would be of utmost importance when considering teachers as authors. The results of this review showed that few authoring tools targeted teachers as authors of their systems, but teachers play a key role in introducing pedagogical innovations in schools and classrooms. Hence, although tutors built with authoring tools have been increasingly demanded and used in schools, if the community intends to bring more of authoring tools to schools there is need to provide more simple and usable solutions for teachers.

Thus, the results shown in this review allow us to state that more research and development is still needed in the field of ITS authoring tools. We need more research on interface of authoring tools to develop simpler and more usable solutions for non-programmer authors; development of authoring tools for different types of tutors (e.g., machine and human-based ITS); more research to empower authoring tools to allow management of the ITS life-cycle; explore more the use of new technologies in authoring tools and tutors such as mobile learning, persuasive technologies, gamification, human-computation, affective computing, motivation-aware ITS, software product line, device-based instruction (i.e., internet of things); more research to develop data-driven and theory-aware authoring tools; more research to strengthen the empirical evidences of authoring tools, particularly with respect to cost-effectiveness and usability. In the following section, we describe in more details these further research directions.

Further Research

This SLR has generated several promising research directions that are important but unanswered or underexplored in current research:

(1) How to simplify the use of ITS authoring tools by non-programmer authors considering design tradeoffs? As discussed in "RQ1: Authoring Tools in ITS Components", our results suggest that ITS authoring tools could be used to design all four main classic ITS components. In fact, each component has its own function and unique properties which may be more or less amenable to

authoring depending on several aspects, i.e., type of ITS, technologies used, needed pedagogical expertise, trade-off choices between usability and flexibility, and so on. For instance, if an ITS authoring tool allows authoring of all four ITS components, it must have high flexibility, but this would come at the expense of higher complexity and decrease in its usability. On the other side, if an ITS authoring tool only provides authoring of few ITS components, it might have high usability and low flexibility levels. Thus, we encourage researchers to conduct more studies (e.g., controlled experiments) to identify what factors may influence on the design decision-making of which ITS components must be considered when designing ITS authoring tools and what are the implications of making these decisions on the simplicity and usability of ITS authoring tools;

(2) How could authoring tools become more than just authoring tools? One might note that some ITS authoring tools identified in this paper are also providing general features (e.g., create class lists, view learners' statistics, reuse/export tutor design, etc), besides authoring features. This result might suggest that researchers are relying on the potential of authoring tools to fulfill other purposes rather than only developing ITS. We believe that authoring tools could work in an integrated way with ITS, playing a crucial role, for example, in the ITS management with the aid of authors. In this direction, there might be an emerging need for authoring tools to provide learning analytics for the teacher or system owners, ensuring transparency (i.e. access and interpretation) of the ITS system collected data, both for the teacher and for the learners. Moreover, they could also provide features to redesign an ITS according to some quality aspects (e.g., performance or motivation) of students' learning using a previous design;

(3) What technologies to use for developing authoring tools? This SLR showed that researchers are mainly considering artificial intelligence techniques, software solutions and distributed technologies to develop their authoring tools. Many factors might influence the design-decision making on which technologies to use in the development of authoring tools. For instance, the designer's background (research/academia versus industry) and their personally favorite technology, technologies they are most comfortable with, the ”fashionable” technology in AI at the time the authoring tool was developed. However, there are current technologies that have been drawing the attention of academics and practitioners in different computer science areas that could be better explored in the design of ITS authoring tools. For example, software product lines (SPL) from software engineering research offer characteristics such as rapid product development, reduced time-to-market, quality improvement, and more affordable development costs in the design of a family of systems. In comparison to other software development strategies used in the papers (e.g., frameworks and services), SPLs are more efficient since they are systematically designed for reuse and provide ways to personalize a product. It seems in general few ITS designers has a background in software engineering, most have a background in AI, Multi-Agent systems, and knowledge representation, and are researchers,

rather than industry people. Nevertheless, we encourage researchers to investigate the use of technologies (including from other research areas) such as SPL in the design of ITS authoring tools. Additionally, our results suggested that web technologies are also underexplored in the design of ITS authoring tools. Perhaps, most of the authoring tools that have been effectively used to design new tutors use non-web systems and the cost to redesign them are high. Thus, once current web technologies are cheap, easy and quick to develop systems, we also suggest more investigation of web technologies along with ITS authoring tools;

(4) How could we provide effective ITS authoring tools relying both on human intelligence and artificial intelligence? To support this question, we recall an interesting discussion about the role of human and artificial intelligence in ITS provided by Baker (2016). In his paper, Baker (2016) argues that tutoring systems that are currently being used at scale are much simpler than the initial vision of ITS. He also raises the possibility that we need "stupid tutoring systems" that are augmented with human intelligence. It means that we probably need tutors that are designed intelligently, and that leverage human intelligence, rather than relying only on artificial intelligence. In this way, regarding the intelligent design of ITS, we believe that authoring tools play an important role to achieve it effectively. Moreover, to leverage human intelligence, humans should be involved as early as possible in ITS design. Hence, a natural way to accomplish it is relying on non-programmer authors from the beginning of an ITS design by using authoring tools and throughout the ITS life-cycle. Thus, we also encourage further investigation on how developing ITS authoring tools that may take advantage of systems' and human intelligence in a balanced way. This research would possibly involve empirical studies to discover in which conditions (e.g., features, technologies, ITS types, ITS components, authoring timing and so on) the effects on learners' performance have significant results;

(5) How to measure the costs and benefits of using ITS authoring for non-programmer authors? The results of research question 6 suggest that there are several pieces of evidence to state that authoring tools properly support the development of intelligent tutoring systems. As presented in "Empirical Benefits and Metrics Investigated", several metrics have been considered by researchers to empirically evaluate ITS authoring tools, for example, efficiency and time for authoring, the effectiveness of authored tutors in terms of learning gains, quality of authored tutor, usability, and so on. In the discussion of "Empirical Benefits and Metrics Investigated", we could detect an agreement in the studies on how to measure the effectiveness of authoring tools–most of the studies conducted pre- and post-tests to ascertain students' learning gains. However, note that researchers do not come to an agreement on what metrics should be considered to evaluate ITS authoring tools for non-programmers authors, particularly, for measuring costs and usability. Some reasons that might explain the diversity of metrics used are: researchers use data they have available; the difficulty of carrying out experiments in-vivo and using data for research purposes (ethics) present hurdles. So often researchers resort to what is possible to get, not what data would ideally answer the research question or evaluate

the system best. In this way, we encourage the further investigation of the relevant metrics for evaluating authoring tools as well as the correlation of these metrics with each other. Regarding the costs of using ITS authoring tools for non-programmer authors, we could identify few metrics to assess the impact that using ITS authoring tools brings to non-programmer authors (e.g., teacher load, learning curve, time for authoring). Thus, it is also necessary to conduct more researches on the measurement of the costs of using ITS authoring tools for this type of author;

(6) Why is there no negative evidence of using authoring tools to design ITS? We might consider two possible reasons for this result: (i) researchers are not publishing papers reporting negative evidence of using ITS authoring tools; and (ii) there is a lack of experimentation on the effects of using ITS authoring tools under several conditions (i.e., ITS components, ITS types, features, technologies and authoring time). In this way, we encourage further experiments controlling these conditions to discover new evidence of the benefits and drawbacks of ITS authoring tools. We also suggest more sharing of research data with other researchers as well as the conduction of replication studies to verify reported evidence;

(7) How could we design evidence and theory-aware ITS authoring tools? One of the potential benefits of ITS authoring tools is to comprise good design principles to support the development of tutors. These principles might include both the available theory about ITS, considering the different types and components of ITS, as well as evidence provided by the empirical studies in the literature. Embedding these principles in authoring tools could favor a new generation of even more intelligent (i.e., human and artificial intelligence) authoring tools that would be aware of both theory and evidence for developing more effective intelligent tutoring systems. To provide awareness of theories and evidence to authoring tools, some directions could be followed. For instance, representing the knowledge about good design principles in a way that enables ITS authoring tool to reason on such knowledge at runtime (e.g., using ontologies), developing a reconfigurable platform that enables reasoning on such knowledge in order change/adapt ITS features according to particular contexts (e.g., students' performance and motivation). Although we acknowledge this seems far-fetched, especially when it is fairly easy to build a simple tutor in a small domain, would it be worth the huge investment, we also encourage the further investigation of these issues, but keeping in mind the cost-benefits for using this kind of authoring system.

Threats to Validity

This section describes concerns that must be improved in future replications of this study and other aspects that must be taken into account to generalize the results of the SLR performed in this work. In order to organize this section, the threats to validity were classified using the Internal, External, Construct and Conclusion categories (Wohlin et al. 2012).

The main constructs in this review are the two concepts “authoring tools” and “intelligent tutoring systems”. For the first concept, we use the term “authoring tool” and its synonyms “authoring system” and ”intelligent authoring” to make sure that all selected studies are related to authoring tool approaches. For the second concept, the terms “intelligent tutoring system” and “intelligent educational systems” are used to ensure high coverage of potentially relevant studies from the database search. A complementary manual search was not performed in the SLR due to the fact there are no conferences and journals specifically focused on the joint use of these concepts. This threat is mitigated by including the general intervention term “authoring tool” along with “intelligent tutoring system” in the terms for the search in seven reputable databases.

As threats to the internal validity, some subjective decisions may have occurred during paper selection and data extraction since some primary studies did not provide a clear description or proper objectives and results, making difficult the objective application of the inclusion/exclusion criteria or the impartial data extraction. In order to minimize selection and extraction mistakes, the selection process was performed in an iterative way; the data extraction was realized collaboratively by reviewers, and any conflicts were discussed and resolved by all the authors. In this way, we tried to mitigate the threats due to personal bias on study understanding. It is also worth noting that the first, second and third authors are Ph.D. students working with authoring tools in the context of computers and education. Moreover, the remaining authors are researchers with expertise in Intelligent Tutoring Systems.

External validity is concerned with establishing the generalizability of the SLR results, which is related to the degree to which the primary studies are representative for the review topic. In order to mitigate external threats, the search process described in “Sources Selection and Search” was defined after several trial searches and validated with the consensus of authors. We tested the coverage and representativeness of retrieved studies, including automatic database search and references scan.

With regards conclusion validity, it is possible that some excluded studies in this review should have been included. To mitigate this threat, the selection process and the inclusion and exclusion criteria were carefully designed and discussed by authors to minimize the risk of exclusion of relevant studies. Furthermore, in the final round of study selection, reviewers conducted the selection process in parallel and independently, and then harmonized their selection results to mitigate the personal bias in study selection caused by individual reviewers. It is worth highlighting that we specified the time period of published studies for this SLR from January 2009 to June 2016. As mentioned in “Methods”, for the best of our knowledge, there is currently no SLR on the use of ITS authoring tools for non-programmer authors. Woolf (2010) provides a general survey about the use of authoring tools to design ITS in her book before 2009. In order to reduce repetitive effort and make use of existing work, we set the starting time of the published studies included in this SLR to January 2009.

Conclusions

In this work, we conducted an SLR to investigate the use of authoring tools to design ITS for non-programmer authors. Our goal was to improve the understanding of how authoring tools support ITS design as well as to identify if there is evidence of its use in this field.

Thirty-three studies out of 4,622 papers were finally included, in which four main ITS components, six ITS types been designed, nineteen features which are facilitating authoring process, three main groups of technologies for developing authoring tools, three main time frames when authoring tool takes place and seven metrics investigated by fifteen empirical studies.

Four metrics reported by empirical studies out of seven deserve special attention, since they report most of the evidence found in the empirical studies: (i) efficiency and time to author, (ii) quality of authored tutor artifacts; (iii) effectiveness of authored tutors with students; and (iv) usability of authoring tools. The review results suggest that the authoring tools benefit ITS authoring for non-programmer authors to deal with several problems such as, supporting the development of example-tracing and model-tracing tutors, checking natural language responses, using machine and human-based intelligence to develop virtual assistant, authoring expert model of cognitive tutors, supporting the development and reducing knowledge acquisition of constraint-based tutors.

Our results also suggest that: 1) among the four ITS components, domain and pedagogical models were much more targeted by authoring tools; 2) several ITS types have been authored, with an emphasis on model-tracing/cognitive and example-tracing tutors; 3) besides providing features for authoring all four ITS components, authoring tools are also presenting general features (e.g., reuse tutor design) to create broader authoring tools; 4) a great diversity of technologies, which include AI techniques, software solutions and distributed technologies, are supporting authoring of the domain, pedagogical and interface models of ITS; 5) all works use ITS authoring tools before instruction, but some studies also target authoring during and/or post-instruction relying both on human and artificial intelligence;

The results presented in this systematic review can be useful to the artificial intelligence in education community, since it gathers evidence from the primary studies included in the review, forming a recent body of knowledge regarding the use of ITS authoring tools for non-programmer authors. As future work, we intend to further investigate some of the research directions presented in this paper, with an emphasis on using theory and evidence-aware authoring tools to design ITS that intelligently manage its life-cycle relying on the human intelligence of teachers and artificial intelligence. Moreover, we intend to expand the scope of this systematic review to explore how authoring tools have been generally supporting different kinds of educational systems.

Acknowledgements This work has been supported by the Brazilian institutions: Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) and Coordenação de Aperfeiçoa-mento de Pessoal de Nível Superior (CAPES).

Appendix: Publication Sources

Table 13 Distribution of studies over publication sources

Publication source	Type	Count	%
International journal of artificial intelligence in education	Journal	7	21.2%
International conf. on artificial intelligence in education (AIED)	Conf.	2	6.1%
International conf. on artificial intelligence in education (AIED) workshops	Workshop	2	6.1%
International conf. on intelligent tutoring systems (ITS)	Conf.	2	6.1%
Advances in intelligent tutoring systems	Book Ch.	1	3.0%
Annual conf. on behavior representation in modeling and simulation (BRiMS)	Conf.	1	3.0%
Computers and education	Journal	1	3.0%
Expert systems with applications	Journal	1	3.0%
IASTED international conf. on computers and advanced technology in education (CATE)	Conf.	1	3.0%
Ibero-American conf. on artificial intelligence (IBERAMIA)	Conf.	1	3.0%
IEE students' technology symposium (TechSym)	Conf.	1	3.0%
IEEE transactions on learning technologies	Journal	1	3.0%
International conf. on artificial intelligence (ICAI)	Conf.	1	3.0%
International conference on foundations of augmented cognition	Conf.	1	3.0%
International conf. on information technology: new generations (ITNG)	Conf.	1	3.0%
International conf. on information, intelligence, systems and applications (IISA)	Conf.	1	3.0%
International conf. on intelligent interactive multimedia systems and services (IIMSS)	Conf.	1	3.0%
International conf. on knowlege-based and intelligent information and engineering systems (KES)	Conf.	1	3.0%
International journal on learning technologies	Conf.	1	3.0%
Journal of information science and engineering	Journal	1	3.0%
Knowledge-based systems	Journal	1	3.0%
Mexican international conf. on artificial intelligence (MICAI)	Conf.	1	3.0%
Object-oriented user interfaces for personalized mobile learning	Book Ch.	1	3.0%
World congress on information and communication technologies (WICT)	Conf.	1	3.0%

References

Achimugu, P., Selamat, A., Ibrahim, R., & Mahrin, M.N. (2014). A systematic literature review of software requirements prioritization research. *Information and Software Technology*, *56*(6), 568–585.

Aleven, V., Mclaren, B.M., Sewall, J., & Koedinger, K.R. (2009). A new paradigm for intelligent tutoring systems: example-tracing tutors. *International Journal of Artificial Intelligence in Education*, *19*(2), 105–154.

Anderson, J.R. (1983). *The architecture of cognition*. Cambridge: Harvard University Press.

Baker, R.S. (2016). Stupid tutoring systems, intelligent humans. *International Journal of Artificial Intelligence in Education*, *26*(2), 600–614.

du Boulay, B. (2016). Recent meta-reviews and meta–analyses of aied systems. *International Journal of Artificial Intelligence in Education*, *26*(1), 536–537.

Chen, L., Babar, M.A., & Zhang, H. (2010). Towards an evidence-based understanding of electronic data sources. In *Proceedings of the 14th international conference on evaluation and assessment in software engineering, British Computer Society, Swinton, UK, EASE'10* (pp. 135–138).

Dermeval, D., Vilela, J., Bittencourt, I.I., Castro, J., Isotani, S., Brito, P., & Silva, A. (2016). Applications of ontologies in requirements engineering: a systematic review of the literature. *Requirements Engineering*, *21*, 1–33.

Ding, W., Liang, P., Tang, A., & van Vliet, H. (2014). Knowledge-based approaches in software documentation: a systematic literature review. *Information and Software Technology*, *56*(6), 545–567.

Dyb, T., & Dingsyr, T. (2008). Empirical studies of agile software development: a systematic review. *Information and Software Technology*, *50*(910), 833–859.

Easterbrook, S., Singer, J., Storey, M.A., & Damian, D. (2008). Selecting empirical methods for software engineering research. In Shull, F., Singer, J., & Sjberg, D. (Eds.) *Guide to advanced empirical software engineering* (pp. 285–311). London: Springer.

Felder, R.M., & Silverman, L.K. (1988). Learning and teaching styles in engineering education. *Engineering Education*, *78*(7), 674–681.

Garg, A.X., Hackam, D., & Tonelli, M. (2008). Systematic review and meta-analysis: when one study is just not enough. *Clinical Journal of the American Society of Nephrology*, *3*(1), 253–260.

Hernandes, E.M., Zamboni, A., Fabbri, S., & Thommazo, A.D. (2012). Using gqm and tam to evaluate start - a tool that supports systematic review. *CLEI Electron J*, *15*(1), 3–3.

Kitchenham, B., & Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. Tech. Rep. EBSE 2007-001, Keele University and Durham University Joint Report.

Koedinger, K.R., & Aleven, V. (2007). Exploring the assistance dilemma in experiments with cognitive tutors. *Educational Psychology Review*, *19*(3), 239–264.

Koedinger, K.R., Corbett, A.T., & Perfetti, C. (2012). The knowledge-learning-instruction framework: bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*, *36*(5), 757–798. https://doi.org/10.1111/j.1551-6709.2012.01245.x.

Kulik, J.A., & Fletcher, J. (2016). Effectiveness of intelligent tutoring systems: a meta-analytic review. *Review of Educational Research*, *86*(1), 42–78.

LAPES (2014). Start - state of the art through systematic review tool. Available in http://lapes.dc.ufscar.br/tools/start_tool. Accessed date: October 2013.

Ma, W., Adesope, O.O., Nesbit, J.C., & Liu, Q. (2014). Intelligent tutoring systems and learning outcomes: a meta-analysis. *Journal of Educational Psychology*, *106*, 901–918.

Mahdavi-Hezavehi, S., Galster, M., & Avgeriou, P. (2013). Variability in quality attributes of service-based software systems: a systematic literature review. *Information and Software Technology*, *55*(2), 320–343. Special Section: Component-Based Software Engineering (CBSE), 2011.

Murray, T. (1999). Authoring intelligent tutoring systems: an analysis of the state of the art. *International Journal of Artificial Intelligence in Education (IJAIED)*, *10*, 98–129.

Murray, T. (2003). An overview of intelligent tutoring system authoring tools: updated analysis of the state of the art. In *Authoring tools for advanced technology learning environments* (pp. 491–544): Springer.

Sleeman, D., & Brown, J.S. (1982). *Intelligent tutoring systems*. London: Academic Press.

Sottilare, R., Graesser, A., Hu, X., & Brawner, K. (2015). Design recommendations for intelligent tutoring systems: authoring tools and expert modeling techniques. Robert Sottilare.

Steenbergen-Hu, S., & Cooper, H. (2013). A meta-analysis of the effectiveness of intelligent tutoring systems on k–12 students' mathematical learning. *Journal of Educational Psychology*, *105*(4), 970.

Steenbergen-Hu, S., & Cooper, H. (2014). A meta-analysis of the effectiveness of intelligent tutoring systems (its) on college students' academic learning. *Journal of Educational Psychology*, *106*, 331–347.

Vanlehn, K. (2006). The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education*, *16*(3), 227–265.

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

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., & Wesslén, A. (2012). Experimentation in software engineering. Springer Science and Business Media.

Woolf, B.P. (2010). Building intelligent interactive tutors: student-centered strategies for revolutionizing e-learning. Morgan Kaufmann.

Systematic Literature Review References

Abbas, M.A., Ahmad, W.F.W., & Kalid, K.S. (2014). Semantic web technologies for pre-school cognitive skills tutoring system. *Journal of Information Science and Engineering*, *30*(3), 835–851.

Alepis, E., & Virvou, M. (2014). Mobile authoring in educational software. *Object-Oriented User Interfaces for Personalized Mobile Learning*, *64*, 31–46.

Aleven, V., McLaren, B.M., & Sewall, J. (2009). Scaling up programming by demonstration for intelligent tutoring systems development: an open-access web site for middle school mathematics learning. *IEEE Transactions on Learning Technologies*, *2*(2), 64–78.

Aleven, V., McLaren, B.M., Sewall, J., van Velsen, M., Popescu, O., Demi, S., Ringenberg, M., & Koedinger, K.R. (2016). Example-tracing tutors: intelligent tutor development for non-programmers. *International Journal of Artificial Intelligence in Education*, *26*(1), 224–269.

Barron-Estrada, L.M., Zatarain-Cabada, R., Zatarain-Cabada, R., Barbosa-Leon, H., & Reyes-Garcia, C.A. (2010). Building and assessing intelligent tutoring systems with an e-learning 2.0 authoring system. In *Proceedings of the Ibero-American conference on artificial intelligence (IBERAMIA)* (pp. 1–9).

Barrón-Estrada, M., Zatarain-Cabada, R., Tamayo, P., Tamayo, S., & Pérez-Espinoza, H. (2011). A learning social network with multi-modal affect. In *Proceedings of the 10th mexican international conference on artificial intelligence: advances in artificial intelligence and applications, MICAI 2011* (pp. 163–168).

Blessing, S.B., Gilbert, S.B., Ourada, S., & Ritter, S. (2009). Authoring model-tracing cognitive tutors. *International Journal of Artificial Intelligence in Education*, *19*(2), 189–210.

Blessing, S.B., Devasani, S., Gilbert, S.B., & Sinapov, J. (2015). Using conceptgrid as an easy authoring technique to check natural language responses. *International Journal of Learning Technology*, *10*(1), 50–70.

Brawner, K.W. (2015). Rapid dialogue and branching tutors. In *Proceedings of the workshops at the 17th international conference on artificial intelligence in education, AIED 2015, Madrid, Spain, June 22 + 26, 2015*.

Chakraborty, S., Roy, D., Kumar Bhowmick, P., & Basu, A. (2010). An authoring system for developing intelligent tutoring system. In *Proceedings of the IEE students' technology symposium (TechSym)* (pp. 196–205).

Chou, C.Y., Huang, B.H., & Lin, C.J. (2011). Complementary machine intelligence and human intelligence in virtual teaching assistant for tutoring program tracing. *Computers and Education*, *57*(4), 2303–2312.

Devasani, S., Gilbert, S., & Blessing, S. (2012). Evaluation of two intelligent tutoring system authoring tool paradigms graphical user interface-based and text-based. In *Proceedings of the 21st annual conference on behavior representation in modeling and simulation, BRiMS 2012* (pp. 51–58).

Escudero, H., & Fuentes, R. (2010). Exchanging courses between different intelligent tutoring systems: a generic course generation authoring tool. *Knowledge-Based Systems*, *23*(8), 864–874.

Fox, R., Schleifer, M., & Cellio, J. (2011). A tool to generate computer assisted instruction systems through hierarchical classification. In *Proceedings of the 2011 international conference on artificial intelligence, ICAI 2011,* (Vol. 1 pp. 385–391).

Gilbert, S.B., Blessing, S.B., & Guo, E. (2015). Authoring effective embedded tutors: an overview of the extensible problem specific tutor (xpst) system. *International Journal of Artificial Intelligence in Education*, *25*(3), 428–454.

Grubisic, A., Stankov, S., & Glavinic, V. (2009). Agent based intelligent courseware generation in e-learning. In *Proceeding of the IASTED international conference on computers and advanced technology in education (CATE)*.

Guin, N., & Lefevre, M. (2013). From a customizable ITS to an adaptive ITS. *International Conference on Artificial Intelligence in Education (AIED)*, *7926 LNAI*, 141–150.

Heffernan, N.T., & Heffernan, C.L. (2014). The assistments ecosystem: building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, *24*(4), 470–497.

Lane, H.C., Core, M.G., Hays, M.J., Auerbach, D., & Rosenberg, M. (2015). Situated pedagogical authoring: authoring intelligent tutors from a student's perspective. In *International conference artificial intelligence in education* (pp. 195–204): Springer International Publishing.

MacLellan C.J., Koedinger K.R., & Matsuda N. (2014). Authoring tutors with simstudent: an evaluation of efficiency and model quality. In *International conference on intelligent tutoring systems* (pp. 551–560): Springer.

MacLellan, C.J., Harpstead, E., Wiese, E.S., Zou, M., Matsuda, N., Aleven, V., & Koedinger, K.R. (2015). Authoring tutors with complex solutions: a comparative analysis of example tracing and simstudent. In *AIED workshops*.

Marcus, N., Ben-Naim, D., & Bain, M. (2010). Instructional support for teachers and guided feedback for students in an adaptive elearning environment. In *Proceedings of the 8th international conference on information technology: new generations, ITNG 2011* (pp. 626-631).

Matsuda, N., Cohen, W.W., & Koedinger, K.R. (2015). Teaching the teacher: tutoring simstudent leads to more effective cognitive tutor authoring. *International Journal of Artificial Intelligence in Education*, *25*(1), 1–34.

Mitrovic, A., Martin, B., Suraweera, P., Zakharov, K., Milik, N., Holland, J., & McGuigan, N. (2009). ASPIRE: an authoring system and deployment environment for constraint-based tutors. *International Journal of Artificial Intelligence in Education*, *19*, 155–188.

Olney, A.M., & Cade, W.L. (2015). Authoring intelligent tutoring systems using human computation: designing for intrinsic motivation. In *International conference on augmented cognition* (pp. 628–639): Springer.

Olsen, J.K., Belenky, D.M., Aleven, V., Rummel, N., Sewall, J., & Ringenberg, M. (2014). Authoring tools for collaborative intelligent tutoring system environments. In *International conference on intelligent tutoring systems* (pp. 523–528): Springer.

Paquette, L., Lebeau, J.F., & Mayers, A. (2010). Authoring problem-solving tutors: a comparison between astus and ctat. In *Advances in intelligent tutoring systems* (pp. 377–405): Springer.

Sklavakis, D., & Refanidis, I. (2011). The MATHESIS semantic authoring framework: ontology-driven knowledge engineering for ITS authoring. *Knowlege-Based and Intelligent Information and Engineering Systems*, pp. 114–123.

Suraweera, P., Mitrovic, A., & Martin, B. (2010). Widening the knowledge acquisition bottleneck for constraint-based tutors. *International Journal of Artificial Intelligence in Education*, *20*(2), 137–173.

Troussas, C., Alepis, E., & Virvou, M. (2014). Mobile authoring in a multiple language learning environment. In *The 5th international conference on information, intelligence, systems and applications, IISA 2014* (pp. 405–410): IEEE.

Virvou, M., & Troussas, C. (2011). Knowledge-based authoring tool for tutoring multiple languages. In *Intelligent interactive multimedia systems and services* (pp. 163–175): Springer.

Wilches, C.O.E., & Palacio, G.V.H. (2014). Development of example-tracing tutors for teaching control systems performance fundamentals. In *Proceedings of the 4th world congress on information and communication technologies* (pp. 290–295): WICT.

Zatarian-Cabada, R., & Barrón-Estrada, M. (2011). Reyes García, C.A. *EDUCA: A web 2.0 authoring tool for developing adaptive and intelligent tutoring systems using a Kohonen network. Expert Systems with Applications*, *38*(8), 9522–9529.