RESEARCH ARTICLE

DomainBuilder: the knowledge authoring system for SlideTutor Intelligent Tutoring system [version 1; peer review: 1 approved, 1 approved with reservations]

Eugene Tseytlin¹, Faina Linkov ^{1,2}, Melissa Castine¹, Elizabeth Legowski¹, Rebecca S. Jacobson¹

¹Department of Biomedical Informatics, University of Pittsburgh, Pittsburgh, PA, 15206, USA

²Ob/Gyn, University of Pittsburgh School of Medicine, Pittsburgh, PA, 15213, USA

V1 First published: 31 Oct 2018, 7:1721
<https://doi.org/10.12688/f1000research.16060.1>

Latest published: 31 Oct 2018, 7:1721
<https://doi.org/10.12688/f1000research.16060.1>

Abstract

One of the major challenges in the development of medical Intelligent Tutoring Systems (ITS) is the development of authored content, a time-consuming process that requires participation of discipline experts. In this publication, we describe the development of software systems called DomainBuilder and TutorBuilder, designed to streamline and simplify the authoring process for general medical ITSs. The aim of these systems is to allow physicians without programming or ITSs background to create a domain knowledge base and author tutor cases in a time efficient manner. DomainBuilder combined knowledge authoring, case authoring, and validation tasks into a single work environment, enabling multiple authoring strategies. Natural Language Processing (NLP) methods were integrated for parsing existing clinical reports to speed case authoring. Similarly, TutorBuilder was designed to allow users to customize all aspects of ITSs, including user interface, pedagogic module, feedback module, etc. Both systems underwent formal usability studies with physicians specializing in dermatology. Open-ended questions assessed usability of the system and satisfaction with its features. Incorporating feedback from usability studies, DomainBuilder and TutorBuilder systems were deployed and used across multiple universities to create customized medical tutoring curriculum. Overall, both systems were well received by medical professionals participating in usability studies with participants highlighting ease of utilization and clarity of presentation. Usability study participants were able to successfully use the system for the authoring tasks. DomainBuilder and TutorBuilder are novel tools that combine comprehensive aspects of content creation, including creation of domain ontologies, case authoring, and validation.

Open Peer Review

Approval Status

	1	2
version 1
31 Oct 2018	view	view

1. **Ana Marusic** , University of Split, Split, Croatia
2. **Claudia Mello-Thoms**, University of Sydney, Sydney, Australia

Any reports and responses or comments on the article can be found at the end of the article.

Keywords

Natural language processing, Intelligent Tutoring Systems, Concept recognition, Biomedical terminologies, ontology development, Auto-coding, System evaluation

Corresponding author: Faina Linkov (linkfy@mail.magee.edu)

Author roles: **Tseytlin E:** Conceptualization, Data Curation, Formal Analysis, Investigation, Methodology, Resources, Software, Validation, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing; **Linkov F:** Formal Analysis, Investigation, Methodology, Project Administration, Supervision, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing; **Castine M:** Conceptualization, Data Curation, Formal Analysis, Investigation, Methodology, Project Administration, Software, Validation, Visualization, Writing – Review & Editing; **Legowski E:** Conceptualization, Data Curation, Investigation, Methodology, Project Administration, Resources, Software, Supervision, Writing – Review & Editing; **Jacobson RS:** Conceptualization, Data Curation, Funding Acquisition, Investigation, Methodology, Project Administration, Resources, Supervision, Validation, Writing – Original Draft Preparation, Writing – Review & Editing

Competing interests: No competing interests were disclosed.

Grant information: The work described in this manuscript has been partially supported in by the National Cancer Institute grant R25CA101959 and Agency for Healthcare Research and Quality grant RR024153. This project was also partially supported by NIH grants 1U24CA180921, R01CA132672, and U54RR023506.

The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Copyright: © 2018 Tseytlin E *et al.* This is an open access article distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. Data associated with the article are available under the terms of the [Creative Commons Zero "No rights reserved" data waiver](#) (CC0 1.0 Public domain dedication).

How to cite this article: Tseytlin E, Linkov F, Castine M *et al.* **DomainBuilder: the knowledge authoring system for SlideTutor Intelligent Tutoring system [version 1; peer review: 1 approved, 1 approved with reservations]** F1000Research 2018, 7:1721 <https://doi.org/10.12688/f1000research.16060.1>

First published: 31 Oct 2018, 7:1721 <https://doi.org/10.12688/f1000research.16060.1>

Introduction

While intelligent tutoring systems (ITSs) are becoming more common in medicine and proving to be increasingly effective in educating medical personnel and patients¹⁻³, they are difficult and expensive to build. One of the major challenges in the development of medical ITSs is the creation of authored content, a time-consuming process that requires domain knowledge expertise.

Authoring systems, computer programs that can allow an instructor to prepare computer-based medical instructional materials without the need to know programming languages or have more than minimal familiarity with the computer hardware, are especially needed in the field of medical resident training, where traditional apprenticeship models of education are costly and time consuming. Previously published reports highlighted the use of various authoring systems for ITSs in non-medical settings, including WETAS⁴, NetCoach⁵, and CTAT⁶. WETAS is a web-based authoring system built to reduce the authoring efforts for constraint-based ITSs. It also functions as a web-based tutoring engine that performs all of the common functions of text-based tutors⁴. This system is well suited for institutional, web-based tutors like SQL-Tutor⁷. NetCoach was designed to enable authors to develop adaptive learning courses without programming knowledge⁵. Cognitive Tutor Authoring Tools (CTAT) is a suite of tools for quick development of cognitive and example tracing tutors⁶. Authoring systems are commercially available for traditional computer aided instruction (CAI) and multimedia-based training, but these authoring systems lack the sophistication required to build intelligent tutors⁸, especially for the medical education setting.

There are many authoring tools that in the past decade lowered the skill threshold for building tutors⁹⁻¹⁶. For example, Blessing *et al.* explored multiple approaches to authoring example-based tutors for procedural tasks. They discussed convergence of multiple lines of authoring tools for step-based problem solving tutors toward example-based authoring, which provides an interface where the author builds tutoring content and student support for an individual example or set of examples¹⁷. Aleven, Roll, *et al.* reviewed work on the creation of a knowledge-engineered, rule-based, executable model of help-seeking that can drive tutoring¹⁸.

The Generalized Intelligent Framework for Tutoring (GIFT), developed by the Army Research Laboratory-Human Research and Engineering Directorate, allows researchers to manipulate the computer-based training system components in order to test empirically the effect for different assessments and instruction strategies on learning outcomes¹⁹. GIFT is an open-source intelligent tutoring system (ITS) architecture provides authoring, delivering, managing, and evaluating adaptive instruction that was very important for the field in the past decade. Similarly important is the previously published AutoTutor system, which helps students learn by holding a conversation in natural language. AutoTutor is adaptive to the learners' actions, verbal contributions, and in some systems their emotions²⁰.

In reviewing authoring tools that have been published in the past two decades, it is important to mention progress in the field of ontology development. An ontology is a general-purpose framework for representing knowledge in various domains. For example, Isotani *et al.* (2008, 2009) used ontologies to represent strategies in collaborative learning and created systems that can link domain independent knowledge with domain content^{21,22}. Their work, however, is specific to their system for collaborative learning and does not apply to the domain of pathology where an ontology is used to represent concept hierarchies of clinical findings and linking them with one or more pathologic diagnoses to support a model-tracing tutor in the pathology domain.

While all of the systems mentioned so far, including those reviewed by Murray⁸, offer a rich set of features and are designed specifically to simplify and streamline the content authoring process for their respective classes of ITS, none of them are designed for domains that present a user with an interactive virtual slide in the field of dermatopathology. The pathology domain is unique because it relies on the use of annotated virtual slide images for training, which makes general purpose ITS authoring systems unusable. Existing authoring systems (WETAS⁴, NetCoach⁵, and CTAT⁶), developed mainly for the domains of mathematics, physics, and programming, focus on the educational content that is presented to the student in the form of a text question with clear solution paths. A notable disadvantage of these three authoring systems for application in the field of medical education is that solution paths are complex and image recognition systems are required, thus these systems may not be adaptable to an ever-changing medical environment. In our previously published studies, we found that these existing systems are ill-suited for authoring knowledge for the Slide Tutor ITS, a system which was designed to train pathology residents²³.

Problem/preliminary studies

Our group started to address educational challenges in the field of pathology resident education by introducing SlideTutor ITS, a visual classification tutor that relies on expert medical knowledge represented in the form of an ontology²⁴. SlideTutor is a general visual classification tutor that was created to teach microscopic diagnosis of inflammatory dermatological diseases²³. SlideTutor is a model-tracing tutor that presents pathology trainees with a virtual whole slide image of dermatopathology and asks them to identify features on the slide in order to come to a differential diagnosis²⁵. For an ITS like SlideTutor, the task requires medical domain expertise, knowledge engineering experience, as well as an understanding of the general ITS architecture. In our previous publications, we have also described ReportTutor, which combines a virtual microscope and a natural language interface to allow students to visually inspect a virtual slide as they type a diagnostic report on the case of melanoma of skin²⁶. A version of SlideTutor was created that explored a student's metacognition²⁷. Most recently, we have created a simulation tutor geared toward practicing generalist pathologists that combined both SlideTutor and

ReportTutor into a single tutoring environment covering both diagnostic and prognostic aspects of solving a case in the melanocytic domain²⁸.

In our previous attempt to build an authoring system for the SlideTutor ITS, the knowledge base was developed using the Protégé ontology editor, which is a free, open-source ontology editor. The Protégé framework was selected as its plug-in architecture can be adapted to build both simple and complex ontology-based applications²⁹. The case authoring system was a Protégé plug-in that integrated the digital slide viewer, which was adequate for our initial research purposes. The disadvantage of using this framework was that mastering Protégé software required a steep learning curve, since understanding ontology development is not a basic skill that general users and even medical professionals generally possess. The lack of an intuitive interface also made collaboration between the knowledge engineer and medical domain experts difficult, as large amount of time was spent training the domain expert on the use of the complex interface. Thus, the disadvantage of the previous approaches developed to build ITS content in this area was that it was never designed for use by medical professionals and hence was not very intuitive for content creators outside of the core ITS group. Existing systems developed by us and others have been reviewed to inform the design of DomainBuilder³⁰ and TutorBuilder³¹ systems described in this paper.

Objectives

While the general architecture of tutoring systems developed in our laboratory remained unchanged³², our work over the past decade focused on improving knowledge representation to accommodate different domains in the field of pathology. Each version of the tutor had its own knowledge representation (specific to each domain), as well as an authoring system for tutor cases. The overall aim of our effort was to ensure that new content for our ITS was usable outside of our laboratory with a minimal learning curve on the side of the user. To accomplish this aim, we focused on developing an intuitive knowledge representation system. We ran four major studies in several pathology domains^{33–36}, focusing on various types of tutor feedback, user meta-cognition, and student modeling. During that time, we developed a list of features that an authoring system would need to have, to streamline the creation and deployment of new content for the SlideTutor ITS. The aim of this manuscript is to describe the development of the software systems called DomainBuilder and TutorBuilder that streamline and simplify the authoring process for general medical ITSs.

Innovative aspects of this work

DomainBuilder and TutorBuilder are innovative in several ways and were designed to fill a serious gap in the current literature on medical ITSs. To our knowledge, our system represents one of the first attempts to build an authoring system for a set of model tracing intelligent tutoring systems in a medical domain. Our approach also explores ways of building intelligent tutors that are adaptable to various healthcare settings and reduces the need to have an input from multiple clinicians (or other domain experts) during design stages. DomainBuilder system

uses a unique interface to link virtual slide annotations with diagnostic and prognostic features. Additionally, we used an innovative spreadsheet style interface for rapidly creating complex rules for diagnostic inference and comparing those rules between diagnoses.

This paper outlines the system descriptions of DomainBuilder and TutorBuilder authoring systems for a medical ITS, as well as describes two usability studies that were conducted to assess the usability of both systems among physicians without programming skills.

Methods

DomainBuilder development process

To meet the needs of training pathologists, the SlideTutor ITS architecture has been developed to accommodate tutor customization and pedagogic content. The domain knowledge for SlideTutor is represented in the form of an application ontology. The SlideTutor ITS utilizes two main constructs: *findings* and *diagnoses*. A *finding* is a piece of evidence that a user can identify in a tutor case, while a *diagnosis* is a solution for a case. Each *diagnosis* can be described by a set of *findings* that can distinguish it from other *diagnoses*. A tutor case is represented by an annotated virtual slide image and a set of findings and diagnoses that are linked to those annotations.

This system builds on our previous experience with the development of a plug-in to an existing ontology editor, Protégé. During review of domain content, the domain expert (an attending pathologist or fellow) often became overwhelmed with reviewing an entire ontology to provide feedback on the content. Also, substantial effort was needed from a knowledge engineer to rearrange the disease and finding hierarchies within the ontology editor. Another disadvantage to the ontology editor was that there was no way to visualize the entire domain space to see what was lacking. Finally, the ontology editor required that the tutoring domain always be added to the ontology from a top-down manner (e.g. once authoring process began, workflow had to be interrupted to add a missing disease to the knowledge base in a separate tool), which interfered with the workflow of case authoring. Systems described in this paper were designed to address these disadvantages of the existing systems.

Implementation

Based on our experience with intelligent tutoring systems^{23,26,32} and a formal usability study, our group developed a design that simplified and streamlined the process of content creation for SlideTutor. Given the unique nature of our ITS and its target group, several key steps were taken to develop DomainBuilder. The first step involved reviewing existing frameworks and previous authoring systems. We utilized the Space Tree System from the University of Maryland as an efficient way to represent concept hierarchies. This provided a simple interface to view the cases and allowed for easy rearrangement of disease and finding hierarchies, as illustrated in [Figure 1](#). SpaceTree is a tree browser that builds on the conventional layout node link diagrams along a single preferred direction³⁷.

Then, a design study/focus group was conducted to identify the most effective methods to represent relationships between “Diagnoses” and “Findings” to the user (a “spreadsheet” design inspired by Excel software format). The “spreadsheet” format allowed for creation and review of domains in small segments that simplified the interactions between the knowledge engineer and the domain expert (Figure 2). The report template layout was inspired by the College of American Pathologists (CAP) cancer protocols to ensure that the layout is consistent with reporting standards and to ensure that each report is completed with the necessary required elements. Google News Cloud and other Word/Tag cloud user interface techniques inspired our validation layout. This layout provided a “birds-eye view” of the domain demonstrating the relationships between findings, diagnoses, and their related cases. Lastly, the authoring layout used to link annotations with domain content, allowed for bottom-up data entry. For instance, DomainBuilder uses NLP to process an existing clinical report and extract the relevant ontology concepts that should be authored in a case. This expedited the authoring process as all that was required was annotating the slide and

TutorBuilder development process

Page 5 of 18

Design objectives

Our overarching design goals were consistent with those outlined by Murray *et al.*, which include: decrease the effort (time, cost, and/or other resources) for making intelligent tutors; decrease the skill threshold for building intelligent tutors; help the designer/author articulate or organize her domain or pedagogical knowledge; support good design principles (in pedagogy, user interface, etc.); and enable rapid prototyping of intelligent tutor designs (i.e. allow quick design/evaluation cycles of prototype software)^{8,41}.

TutorBuilder had the following design objectives: (i) to provide an intuitive graphical user interface (GUI) for authoring that enables any user, even those without the technical abilities, to construct tutoring systems without the need to understand the

underlying knowledge representation, (ii) to provide the ability to mix and match different inter-operable components of the tutoring system, (iii) to provide the ability to evaluate the tutoring system as it is currently configured from the authoring tool itself, (iv) to provide the ability to modify the feedback type, as well as its content, and (v) to provide the ability to reproduce previous configurations of the tutoring system that were hand crafted before TutorBuilder's creation within the authoring tool.

Implementation

Design of prototype. Incorporating feedback from the usability studies, DomainBuilder and TutorBuilder systems were deployed and used across multiple universities to create customized medical tutoring curriculum. The use of both systems allowed for several major research studies to be conducted in a short period of time^{23,26,32,42}. One of the best examples of user feedback that drove a change to the prototype was a request to develop a simpler user interface for tutor builder to accommodate the most frequently used tutor configurations.

Validation study design. A total of two separate usability studies were conducted: one for DomainBuilder and one for TutorBuilder. The DomainBuilder authoring system was used to create domain content, while TutorBuilder was used to create customized ITS configurations independent of content. As both systems were developed separately and had a different target user base, two usability studies were required. Considering internal feedback from the researchers that were using both systems during the development steps, as well as feedback from two formal usability studies, there were three major cycles of revisions for both systems.

DomainBuilder usability study. The DomainBuilder usability study was conducted with ten medically trained pathology users with varying levels of postgraduate training and no previous experience with ITSs building. The participants have been recruited through pathology research conferences such as Pathology Informatics APIII, College of American Pathologists (CAP), and American Society for Clinical Pathology (ASCP). Participants were given 32 tasks (Supplementary File 1) to complete, with each task consisting of multiple steps for testing the usability of the interface. Each step had four possible categorical response types: (i) successfully perform the task, (ii) fail to perform the task, (iii) complain about the step, and (iv) report a surprise (unanticipated problem with the interface).

Following the usability tasks, participants completed a survey assessing their opinions on various aspects of the system. The questionnaire (Supplementary File 2) included 29 scaled, closed-ended questions on ease of use, understanding, and sufficiency of numerous features of the system and 3 open-ended questions in which users were asked their opinion on what they liked and disliked about the system and suggestions for future tool improvements and modifications.

TutorBuilder usability study. The TutorBuilder usability study followed the same design as the DomainBuilder usability study; however, the tasks were different and specific to the TutorBuilder

system. A total of six pathologists with varying levels of post-graduate training and no previous experience with ITSs building participated in the usability study. They were recruited through pathology training programs in the greater Pittsburgh area.

While our original intent was for the usability study to follow the model of learning evaluation published by Kirkpatrick⁴³, which focused on user reaction, learning, behavior, and results, we found that this model has been criticized for lack of research. A newer model suggested by Holton focuses on three primary outcomes, including learning, individual performance, and organizational results. These are defined, respectively, as achievement of the learning outcomes, change in individual performance because of the learning being applied on the job, and results at the organization level as a consequence of the change in individual performance. Thus, our final plan for a usability study has been inspired by both, the models suggested by Kirkpatrick and Holton⁴⁴.

Ethical approval for the validation studies was obtained from the University of Pittsburgh (020348 and 0608117). All participants provided written informed consent prior to participation.

Operation

Minimum system requirements for both products. The system was developed in Java and deployed using Java WebStart technology. There was no specialized hardware requirements to run the software on a typical personal computer.

Domain builder description. The primary purpose of DomainBuilder is to author ITS domain content. This content includes a domain ontology in the form of findings, diagnoses, and their relationships. In addition to the ontology, the domain content also includes medical cases to be solved by students, which are generally presented as annotated virtual slide images. The ITS uses domain ontology along with the medical tutor case information to reason about a given domain and provide appropriate feedback to a student.

Knowledge representation and implementation. The SlideTutor ITS utilizes two main constructs: *findings* and *diagnoses*. A *finding* is a piece of evidence that a user can identify in a tutor case, while a *diagnosis* is a solution for a case. Each *diagnosis* can be described by a set of *findings* that can distinguish it from other *diagnoses* (e.g. the findings vacuolization in basal layer, increased mucin, and mild perivascular lymphocytic infiltrate indicate a diagnosis of dermatomyositis). While those constructs are specific to medical domains, any knowledge representation that distinguishes these two components can be easily adopted within our ITS framework. The Knowledge Base (KB) is implemented using Web Ontology Language (OWL)⁴⁵. We used Protégé-OWL for KB development and Pellet for Description Logic (DL) reasoning. Figure 2 shows the general KB schema that implements the required ITS framework. All concept entities in our KB are represented using OWL Classes, while OWL Individuals are used to represent tutor case-specific instance information.

Knowledge authoring. The graphical knowledge base creation tool is divided into three interfaces, each of them designed to streamline a specific subtask in the knowledge creation process.

The Hierarchy builder tab (Figure 1) is focused on building a desired hierarchy of concepts such as findings and diagnoses as well as adding meta-information for each concept, such as the concept's definition, possible synonyms, linkage to a controlled medical vocabulary and an optional pictorial example. For example, the user would be required to enter the findings vacuolization in basal layer, increased mucin and mild perivascular lymphocytic infiltrate into the findings hierarchy. Similarly, the user would create a concept for the related diagnosis, dermatomyositis, in the disease hierarchy. To facilitate easy and efficient navigation through the ontology, the SpaceTree technology is used⁴⁶. In addition to adding findings and diagnoses manually, a user can import existing concept hierarchies from any ontology deployed at BioPortal. Having well-structured concept hierarchies in a domain ontology allows the ITS to reason in general and specific terms about features in a domain. For example, if the student chooses a diagnosis of "perivascular skin disease" the tutoring system can provide feedback that the student must find more evidence to choose a more specific diagnosis for the given case. Meta information provided with each finding and diagnosis allows the ITS to provide definitions and examples to a student to aid in the learning of domain. Additional synonymy is used to generate dictionaries for the ITS that requires NLP to process a student's input for correctness and completeness.

The Diagnosis builder tab (Figure 2) is used to create relationships between findings and their related diseases. It uses a table layout, where each column represents a diagnosis with rows below representing a disjunction of findings. For example, a user would add a column by choosing the disease, dermatomyositis from the disease hierarchy to create a column. Once the column is created, the user can then click on each cell under the disease name to enter the findings related to that disease (e.g. vacuolization in basal layer, increased mucin and mild perivascular lymphocytic infiltrate). To create more complex relationships, findings can also be negated as well as conjoined together to form a logical OR expression. Also, this interface allows users to compare similarities between several diagnoses by rearranging rows and displaying them side-by-side. These relationships between findings and diagnoses (diagnostic rules) are used by the ITS reasoning module to check if a student's partial solution is sound and to provide student with possible next steps in a solution path if a hint is requested.

The Report builder tab is designed for creation of reporting templates that are used by the SlideTutor ITS when teaching students how to write clinical reports. The Report builder interface consists of a list of potential templates, where each template has a list of triggers that determine when this template should be used as well as a list of *reportable items* that should be mentioned in a student's clinical report. For instance, the user could create a trigger of dermatomyositis allowing the tutoring system to enforce the defined reportable items

(e.g. must report the tissue type of skin, the anatomic location of the biopsy and the biopsy procedure type before reporting the diagnosis) for that given diagnosis. Reporting templates are used by the prognostic ITS that checks students' input against a standard checklist for accuracy and completeness.

Case authoring. Based on our past experience with authoring domain knowledge, tutor case authoring is the most time-consuming task. A tutor case usually consists of one or more annotated digital slides, a list of findings that can be observed on those slides, one or more diagnoses for that case, as well as an optional expert report associated with that case (Figure 3).

The Virtual Slide Component is the most visible and the most widely used component of the system, since most of the time during case authoring is spent adding annotations to digital slides. It consists of a virtual slide viewer, a slide manager, and an annotation manager. The virtual slide viewer can load whole slide images from several different vendors and should be familiar to most pathologists. The slide manager is used to switch between several different slides that belong to a single case. Lastly, the annotation manager is used to organize annotations such as free hand polygons, rectangles, ovals and arrows into tagged groupings. These groupings can then be linked to findings and/or diagnoses associated with the case by dragging and dropping the tag name to a finding or diagnosis.

A list of findings from the knowledge base that are found or should be found in the given case are displayed to the user above the virtual slide. Findings can be added in three different ways: (i) manually added by a user (ii) inferred by the system from a diagnosis, or (iii) extracted from NLP processing of the clinical report. Those findings are used as objects that can be associated with annotations on a digital whole slide. DomainBuilder uses the underlying knowledge base to infer appropriate findings and diagnoses. Given a diagnosis, DomainBuilder can add a set of findings to the interface for the user to author in this tutor case. Conversely, given a set of findings, the tool can infer and add one or more relevant diagnoses to the interface for the user to author in the tutor case.

The expert report panel looks like a regular text panel with predefined report section headings. One of the advantages of DomainBuilder is that it uses Natural Language Processing (NLP) techniques to parse an already existing report and extract the relevant diagnoses and findings. For parsing, it uses the SPECIALIST noun phrase parser⁴⁷ in combination with a technique similar to IndexFinder's algorithm⁴⁸ to achieve accurate results. This functionality helps expedite the case authoring and also aids in validation of the authored tutor case content.

Validation. During the validation process, the knowledge base and authored cases are checked for consistency. To complete the validation process, the user needs to see the entire knowledge base, as well as all authored cases. This information needs to be presented in a compact fashion, so it does not overwhelm the user with large amounts of data and the relationships between elements can be viewed. After evaluating several UI techniques,

it was decided to use a tag cloud interface⁴⁹. This method allows users to quickly explore relationships between elements in a limited amount of screen space.

Description of domain authoring workflow. Since DomainBuilder encompasses modules for authoring of tutor cases as well as domains, it supports several approaches for content authoring. One option is to use a top-down approach, where knowledge is authored first and cases are authored using this authored knowledge. The second option is to use a bottom-up method, where new knowledge is added to the knowledge base while authoring tutor cases.

The diagram in [Figure 4](#) demonstrates two potential workflow scenarios. Grey boxes indicate steps that involve the use of the DomainBuilder tool. In the initial step, “**Acquire Domain Knowledge**”, authors must acquire adequate domain knowledge from a domain expert, medical texts, and other relevant materials. In “**Create Domain Knowledge Base**”, the DomainBuilder tool is used for knowledge authoring. Domain knowledge is created using the Knowledge Authoring tab, which

is further subdivided into Hierarchy Builder, Diagnosis Builder, and Report Builder tabs. In the “**Acquire Cases**” step, authors must acquire the desired number of medical cases for the target domain, preferably covering most of the knowledge base. For each case, the author should have a set of digital whole slide images and a de-identified expert report from the provider who diagnosed that case. The “**Case Authoring**” tab in the DomainBuilder tool is used for authoring tutor cases. In the “**Check Authoring**” step, users are checking the knowledge and case authoring in the tool with the domain expert. Changes to the system can be introduced, when relevant, based on the expert’s feedback.

After that, the “**Validation**” tab in DomainBuilder is used to check for consistency between existing, authored cases and the current knowledge base.

An alternative process of content creation involves the same steps with one small modification. In this alternative pathway, a user combines steps two and four into a continuous interactive process. “**Create Knowledge Base while Authoring Cases**”

Figure 3. Case authoring tab of DomainBuilder.

Figure 4. Content authoring workflow.

step uses the Case Authoring tab to add new cases. However, since all the required knowledge might not exist during the process of case authoring, the user may add missing concepts to the knowledge base as they are encountered straight from the authoring tab. The advantage of this technique is that it allows users to jump right into case authoring without a need to spend any time developing the underlying knowledge base. The big disadvantage of this technique is that the resulting knowledge base will only fit the set of authored cases, requiring revision of the knowledge base with each newly added case.

Tutor builder description. The primary purpose of TutorBuilder is to create and customize an ITS configuration. Aspects of the ITS such as type of feedback, user interface as well as pedagogic strategy can be customized using TutorBuilder.

Tutor Builder uses a wizard interface to create a configuration and customization for the SlideTutor ITS. SlideTutor is composed of several interlinking modules communicating with each other via a message bus. At any point of the configuration process, TutorBuilder can run a customized version of the ITS directly from the tool to demonstrate results of customization. [Figure 5](#) demonstrates the TutorBuilder interface with advanced configuration options enabled.

The presentation module presents a tutoring case to a trainee mostly in the form of a virtual slide or a question. The interface module is an interface where a student's solution to a tutoring case is presented. There are two available interfaces for solving diagnostic problems and one interface for typing a prognostic case report. The reasoning module describes tutor behavior in response to a student's input. The feedback module allows users to choose the type of feedback given to a student. For example, the tutor can provide immediate feedback to a student's errors and allow for hints, while delayed feedback displays a student's errors at the point of submission. The pedagogic module is designed to suggest the next case for a user to solve, while the student module keeps track of a student's learning progress. The behavior module describes a customizable behavior of the tutoring system when it is interacting with a user. Finally, the protocol module keeps track of all of student and tutor actions and saves them to a database.

The TutorBuilder workflow consists of the following sequence of steps within a wizard interface, which customizes each aspect of the ITS. A "simple" workflow allows users to create a configuration with minimal customization by selecting from a limited number of pre-configured options, while the "advanced" workflow gives the option to change every aspect of the ITS configuration.

Results

DomainBuilder usability study

DomainBuilder is an authoring system that is designed to be easy-to-use to complete the tasks of knowledge engineering and case authoring. Due to time constraints, the usability study had to be composed of much smaller sub-tasks that could be accomplished by untrained users to test for usability of the system. Good examples of usability problems were linking virtual slide annotations with findings, as well as creating complex rules for inferring diagnoses from a set of findings. Overall, several usability problems focused on small subtasks within the authoring workflow and were targeting users who were not familiar with the system, its architecture, and logic flow. Users were successful in accomplishing the task of completing knowledge engineering tasks. Users demonstrated a general ability to use a highly complex system with no prior instruction and accomplish the tasks given, establishing usability of the system and ability to use it for its intended purposes.

The DomainBuilder authoring tool usability study had 10 pathologists of varying levels of expertise (4 residents, 3 fellows, 3 practicing pathologists) as participants. Additionally, one of the authors of this manuscript (MC) also informally participated in the study and provided her feedback for system improvement. Participants were given 32 tasks to complete; these tasks were comprised of numerous actions that had to be performed to complete the tasks. Throughout these actions, approximately 95% of the interface was tested. Overall, between 166 and 202 steps were completed within the 32 tasks. A particular action was considered a usability problem for a user if one of the following criteria were met: (i) the elapsed time on task was greater than two minutes, (ii) three or more different actions were attempted for a task, (iii) the user gave up on the task, (iv) the user made a design suggestion, (v) the user showed a negative response to the task, (vi) the user showed surprise, or

Figure 5. TutorBuilder interface with advanced configuration options enabled.

(vii) the user did not perform the action correctly. An action was considered a major usability problem if two or more users demonstrated a usability problem at that step. A total of 75 actions met the criteria for a usability problem for at least one user. The mean number of usability problems per user was 14 out of an average 187 steps per user (8%) with the highest number of usability problems reported by fellows. There were 30 steps that were reported to have “major” usability problems (at least 2 users with a usability problem).

Following the usability tasks, users were given a questionnaire asking their opinions on various aspects of the system. The questionnaire included 29 scaled closed-ended questions on ease of use, understanding, and sufficiency of numerous features of the system and 3 open-ended questions in which users were asked their opinion on what they liked and disliked about the system and suggestions for future additions to the tool.

Several actions were taken in response to user feedback. A new feature was added that allowed users to enrich the concept hierarchy with synonyms and definitions from the Unified Medical Language System (UMLS). Another feature that was requested and implemented was the sorting of findings associated with each diagnosis to display differences and similarities between diagnostic parameters.

TutorBuilder usability study

The TutorBuilder usability study had a total of six participating pathology fellows. Subjects were given 11 tasks to complete; these tasks were comprised of numerous steps that had to be performed to complete the assignment. The usability problem was defined the same way as with the DomainBuilder study. The average number of usability problems reported per user was 15.3, with 34 steps (34/110 or 31% of steps) that caused usability problems for an average user. There were 24 steps

(or 21% of steps) that were “major” usability problems (at least 2 users with a usability problem). Just like with the DomainBuilder system, users were successful in accomplishing the tutor customization tasks and using the system for its intended purposes.

As part of the user and internal feedback, a simplified user interface was added to the TutorBuilder UI that covered most frequently used tutor configuration options in a single screen, while a more customizable configuration workflow was presented as an “advanced” option.

Dataset 1. Results of the usability study and responses to questionnaire

<https://dx.doi.org/10.5256/f1000research.16060.d221857>

Discussion

Summary of findings

This paper describes the development of the DomainBuilder and TutorBuilder software systems that streamline and simplify the authoring and medical tutor building process for general medical ITSs. These systems combined knowledge authoring, case authoring, and validation tasks into a single work environment, enabling multiple authoring strategies. While the underlying concepts utilized for the development of these systems have been published in the domain of mathematics and other sciences, DomainBuilder and TutorBuilder are different from these systems in several respects, with consequences for its performance, scalability, extensibility, and customizing capacities for highly specific medical domains.

Both physical and virtual slide sets are the closest educational resources training pathologists have to learn the domain of pathology. However, these tools are not classified as ITSs as they do not give the user just in time feedback and they do not adequately teach users inference of diagnosis from given set of findings in a case. They also fail to teach the training pathologists how to write complete and accurate prognostic reports.

The current versions of DomainBuilder and TutorBuilder incorporate the latest technology to ultimately achieve their goals of allowing medical doctors with minimal knowledge engineering training or ITSs background to create a domain knowledge base, to author tutor cases, and to develop novel educational modalities. Between the University of Pittsburgh and the University of Pennsylvania, DomainBuilder has already been used to create more than 20 medical domain ontologies and author more than 500 tutor cases. Two NIH-funded studies have been conducted using the new SlideTutor ITS, which uses cases produced by DomainBuilder⁴².

Several technologies that were developed specifically for DomainBuilder are beneficial for other applications not related to ITSs. The implementation of an independent Ontology API that wraps Protégé OWL and BioPortal 2.0 is used by the ODIE project. Several NLP tools – such as concept coder, reimplementation of string normalization, and negation detection – that were initially developed for DomainBuilder later morphed into

the NOBLE project⁵⁰. The Digital Whole Slide Image server that wraps vendor agnostic OpenSlide library from CMU^{51,52} is another example of general purpose technologies that stemmed from DomainBuilder development, but could be used anywhere.

Limitations

The limitation of the usability studies is the fact that the authoring system has been tested only in the setting of inflammatory dermatological diseases. Future studies will focus on testing the system in other medical domains. An additional limitation of our study is limited number of participants in our studies, which would also be addressed in our future efforts. It is also important to point out that while DomainBuilder and TutorBuilder can cover a wide range of model tracing tutors in the medical domain, they are not as general purpose as some of the other authoring systems reviewed in the introduction. Another limitation of the usability studies outlined in this manuscript is the inability to measure the complete workflow of either DomainBuilder or TutorBuilder systems to assess the quality of user generated content. Because of that, the quality of a user generated tutoring system cannot be assessed.

Current versions of DomainBuilder and TutorBuilder accommodate medical domains that require the use of a virtual microscope or another visual representation in cases that require users to identify clinically relevant findings. Instructional materials in other formats are not currently supported by either authoring systems.

Conclusion

DomainBuilder is a unique authoring framework that creates a systematic workflow to accomplish a highly complex task of content authoring for a medical visual classification ITS. While still in the beta version, it is already used in a research setting, delivering content that is already being used in two major research studies. By allowing instructors to create new curricula for a generic visual classification ITSs, DomainBuilder can address challenges in both the ITSs and medical education fields by delivering systems that can save resources needed for educating highly specialized physicians.

Similarly, TutorBuilder allows ITSs configuration and customization by non-programmers to fit the educational needs of a teaching curriculum.

As the domain ontology was developed in the same tool as tutoring cases were being authored, it was possible to author cases without having a complete domain knowledge mapped out and only add representation for components that had cases. Integration of the concept cloud in the validation tab, allowed authors to have a view of the entire domain content in a single screen, which aided in identifying possible authoring errors and generated domain statistics based on the informal impression of the participating authors.

Development of DomainBuilder also contributed to the development of several NLP related technologies, including NOBEL, a flexible concept recognition tool for large-scale biomedical natural language processing⁵⁰.

Recommendations and future work

We previously reported on the concept of remote pathology consultations with external organizations across the world⁵³. The same virtual slide technology that is a part of DomainBuilder, powered a remote pathology consultation system⁵³. We envision that the systems we describe in this paper as well as their enabling technologies will help with pathology education and potentially pathology consultations across the world. This technology may especially be useful for training medical pathologists focusing on rare diseases or conditions, as traditional apprenticeship based medical educational models may not expose students to important rare disease cases. Between DomainBuilder and TutorBuilder, an educator will be able to create his unique medical visual classification ITSs with his own knowledge representation, cases, tutor feedback mechanism, and message content that will be harnessed to improve pathology education globally.

Data availability

Dataset 1. Results of the usability study and responses to questionnaire. DOI: <https://doi.org/10.5256/f1000research.16060.d221857>⁵⁴.

Software availability

DomainBuilder software is available as a web start application: <http://slidetutor.upmc.edu/domainbuilder/>.

Source code available from: <https://github.com/dbmi-pitt/domainbuilder>.

Archived source code at time of publication: <https://doi.org/10.5281/zenodo.1458957>³⁰.

TutorBuilder software is available as a web start application: <http://slidetutor.upmc.edu/its>.

Source code available from: <https://github.com/dbmi-pitt/slidetutor>.

Archived source code at time of publication: <https://doi.org/10.5281/zenodo.1458959>³¹.

License: BSD 3-Clause.

Grant information

The work described in this manuscript has been partially supported in by the National Cancer Institute grant R25CA101959 and Agency for Healthcare Research and Quality grant RR024153. This project was also partially supported by NIH grants 1U24CA180921, R01CA132672, and U54RR023506.

The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Acknowledgements

The authors would like to thank the project members, students, and medical doctors who participated in this study for their valuable contributions to the design and testing of the systems described in this publication.

Supplementary material

Supplementary File 1. DomainBuilder usability tasks.

[Click here to access the data](#)

Supplementary File 2. DomainBuilder usability questionnaire.

[Click here to access the data](#)

References

- Association for Computing Machinery: **ACM Digital Library**. 2017.
Reference Source
- VanLehn K: **The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems**. *Educ Psychol*. 2011; 46(4): 197–221.
Publisher Full Text
- Wolfe CR, Reyna VF, Widmer CL, *et al.*: **Efficacy of a web-based intelligent tutoring system for communicating genetic risk of breast cancer: a fuzzy-trace theory approach**. *Med Decis Making*. 2015; 35(1): 46–59.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**
- Martin B, Mitrovic A: **WETAS: A Web-Based Authoring System for Constraint-Based ITS**. In: P. De Bra, P. Brusilovsky, R. Conejo (Eds.) *Adaptive Hypermedia and Adaptive Web-Based Systems: Second International Conference, AH 2002* Málaga, Spain, May 29–31, 2002 Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, 2002; 543–546.
Publisher Full Text
- Weber G, Kuhl HC, Weibelzahl S: **Developing Adaptive Internet Based Courses with the Authoring System NetCoach**. In: S. Reich, MM. Tzagarakis, PME. De Bra (Eds.) *Hypermedia: Openness, Structural Awareness, and Adaptivity: International Workshops OHS-7, SC-3, and AH-3 Aarhus, Denmark, August 14–18, 2001 Revised Papers*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2002; 226–238.
Publisher Full Text
- Aleven V, McLaren BM, Sewall J, *et al.*: **The Cognitive Tutor Authoring Tools (CTAT): Preliminary Evaluation of Efficiency Gains**. In: M. Ikeda, KD. Ashley, TW. Chan (Eds.) *Intelligent Tutoring Systems: 8th International Conference, ITS 2006*, Jhongli, Taiwan, June 26–30, 2006. Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006; 61–70.
Publisher Full Text
- Mitrovic A, Martin B, Mayo M: **Using Evaluation to Shape ITS Design: Results and Experiences with SQL-Tutor**. *User Model User-adapt Interact*. 2002; 12(2–3): 243–279.
Publisher Full Text
- Murray T: **An Overview of Intelligent Tutoring System Authoring Tools: Updated Analysis of the State of the Art**. In: T. Murray, S. Blessing, S. Ainsworth (Eds.) *Authoring Tools for Advanced Technology Learning Environments: Toward Cost-effective Adaptive, Interactive, and Intelligent Educational Software* Netherlands,

- Kluwer Academic Publishers, 2003; 491–544.
[Publisher Full Text](#)
9. Blessing S, Devasant S, Gilbert S: **Evaluation of WebxPST: A browser-based authoring tool for problem specific tutors.** In: G. Biswas, S. Bull, J. Kay, A. Mitrovic (Eds.) *Proceedings of the 15th International Conference on Artificial Intelligence in Education*. LNAI 6738, Springer, Berlin, 2011; 423–425.
[Publisher Full Text](#)
 10. Blessing SB, Gilbert SB, Ourada S, *et al.*: **Authoring model-tracing cognitive tutors.** *Int J Artif Intell Educ.* 2009; **19**: 189–210.
[Reference Source](#)
 11. MacLellan CJ, Koedinger KR, Matsuda N: **Authoring Tutors with SimStudent: An Evaluation of Efficiency and Model Quality.** In: S. Trausan-Matu, KE. Boyer, M. Crosby, K. Panourgia (Eds.) *Intelligent Tutoring Systems: 12th International Conference, ITS 2014, Honolulu, HI USA, June 5–9, 2014*. Proceedings, Springer International Publishing, Cham, 2014; 551–560.
[Publisher Full Text](#)
 12. Mitrovich A, Martin B, Suraweera P, *et al.*: **ASPIRE: An authoring system and deployment environment for constraint-based tutors.** *Int J Artif Intell Educ.* 2009; **19**: 155–188.
[Reference Source](#)
 13. Nye BD, Graesser AC, Hu X: **AutoTutor and Family: A Review of 17 Years of Natural Language Tutoring.** *Int J Artif Intell Educ.* 2014; **24**(4): 427–469.
[Publisher Full Text](#)
 14. Paquette L, Lebeau JF, Beaulieu G, *et al.*: **Designing a Knowledge Representation Approach for the Generation of Pedagogical Interventions by MTTs.** *Int J Artif Intell Educ.* 2015; **25**(1): 118–156.
[Publisher Full Text](#)
 15. Razzaq L, Patvarczki J, Almeida SF, *et al.*: **The ASSISTment Builder: Supporting the Life Cycle of Tutoring System Content Creation.** *IEEE Transactions on Learning Technologies Special Issue on Real-World Applications of Intelligent Tutoring Systems.* 2009; **2**(2): 157–166.
[Publisher Full Text](#)
 16. Sottolare R: **Considerations in the development of an ontology for a generalized intelligent framework for tutoring.** *ISM Defense and Homeland Security Simulation Conference, DHSS, Vienna, Austria.* 2012.
[Publisher Full Text](#)
 17. Blessing SB, Alevan V, Gilbert SB, *et al.*: **Authoring Example-based Tutors for Procedural Tasks.** In: R. Sottolare, A. Graesser, X. Hu, K. Brawner (Eds.) *Design Recommendations for Adaptive Intelligent Tutoring Systems: Authoring Tools*, U.S. Army Research Laboratory, Orlando, FL, 2015; 71–93.
[Reference Source](#)
 18. Alevan V, Roll I, McLaren BM, *et al.*: **Help Helps, But Only So Much: Research on Help Seeking with Intelligent Tutoring Systems.** *Int J Artif Intell Educ.* 2016; **26**(1): 205–223.
[Publisher Full Text](#)
 19. Sottolare RA, Brawner KW, Sinatra AM, *et al.*: **An Updated Concept for a Generalized Intelligent Framework for Tutoring (GIFT).** In: U.A.R.L.H.R.E.D. (ARL-HRED) (Ed.), *US Army Research Laboratory, Orlando, FL*, 2017.
[Publisher Full Text](#)
 20. Graesser AC: **Conversations with AutoTutor Help Students Learn.** *Int J Artif Intell Educ.* 2016; **26**(1): 124–132.
[Publisher Full Text](#)
 21. Isotani S, Inaba A, Ikeda M, *et al.*: **An ontology engineering approach to the realization of theory-driven group formation.** *Int J Comput Support Collab Learn.* 2009; **4**(4): 445–478.
[Publisher Full Text](#)
 22. Isotani S, Mizoguchi R: **Adventures in the Boundary between Domain-Independent Ontologies and Domain Content for CSCL.** In: I. Lovrek, RJ. Howlett, LC. Jain (Eds.) *Knowledge-Based Intelligent Information and Engineering Systems: 12th International Conference, KES, 2008, Zagreb, Croatia, September 3–5, 2008*, Proceedings, Part III Springer Berlin Heidelberg, Berlin, Heidelberg, 2008; 523–532.
[Publisher Full Text](#)
 23. Crowley RS, Medvedeva O: **An intelligent tutoring system for visual classification problem solving.** *Artif Intell Med.* 2006; **36**(1): 85–117.
[PubMed Abstract](#) | [Publisher Full Text](#)
 24. El Saadawi GM, Tseytlin E, Legowski E, *et al.*: **A natural language intelligent tutoring system for training pathologists: implementation and evaluation.** *Adv Health Sci Educ Theory Pract.* 2008; **13**(5): 709–722.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
 25. Crowley R, Legowski E, Medvedeva O, *et al.*: **An ITS for medical classification problem-solving: Effects of tutoring and representations.** In: CK. Looi, G. McCalla, B. Bredeweg, J. Breuker (Eds.) *Artificial Intelligence in Education*, IOS Press, Amsterdam, The Netherlands, 2005; 192–199.
[Reference Source](#)
 26. Crowley RS, Tseytlin E, Jukic D: **ReportTutor - an intelligent tutoring system that uses a natural language interface.** *AMIA Annu Symp Proc.* 2005; 171–175.
[PubMed Abstract](#) | [Free Full Text](#)
 27. Yudelov MV, Medvedeva O, Legowski E, *et al.*: **Mining student learning data to develop high level pedagogic strategy in a medical ITS.** *Workshop on Education Data Mining at AAAI 2006, AAAI, Menlo Park, CA*, 2006.
[Reference Source](#)
 28. Crowley R, Grzybicki D, Legowski E, *et al.*: **Use of a Medical ITS Improves Reporting Performance among Community Pathologists.** In: V. Alevan, J. Kay, J. Mostow (Eds.) *Intelligent Tutoring Systems: 10th International Conference, ITS 2010, Pittsburgh, PA USA, June 14–18, 2010*, Proceedings, Part I Springer Berlin Heidelberg, Berlin, Heidelberg, 2010; 338–348.
[Publisher Full Text](#)
 29. Noy NF, Sintek M, Decker S, *et al.*: **Creating Semantic Web contents with Protege-2000.** *IEEE Intell Syst.* 2001; **16**(2): 60–71.
[Publisher Full Text](#)
 30. Tseytlin E: **dbmi-pitt/domainbuilder: Final Release (Version 1.0).** *Zenodo.* 2018.
<http://www.doi.org/10.5281/zenodo.1458957>
 31. Tseytlin E: **dbmi-pitt/slidetutor: Final Release (Version 1.0).** *Zenodo.* 2018.
<http://www.doi.org/10.5281/zenodo.1458959>
 32. Crowley RS, Medvedeva O: **A general architecture for intelligent tutoring of diagnostic classification problem solving.** *AMIA Annu Symp Proc.* 2003; **2003**: 185–189.
[PubMed Abstract](#) | [Free Full Text](#)
 33. Crowley RS, Legowski E, Medvedeva O, *et al.*: **Automated detection of heuristics and biases among pathologists in a computer-based system.** *Adv Health Sci Educ Theory Pract.* 2013; **18**(3): 343–363.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
 34. El Saadawi GM, Azevedo R, Castine M, *et al.*: **Factors affecting feeling-of-knowing in a medical intelligent tutoring system: the role of immediate feedback as a metacognitive scaffold.** *Adv Health Sci Educ Theory Pract.* 2010; **15**(1): 9–30.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
 35. Feyzi-Behnagh R, Azevedo R, Legowski E, *et al.*: **Metacognitive Scaffolds Improve Self-Judgments of Accuracy in a Medical Intelligent Tutoring System.** *Instr Sci.* 2014; **42**(2): 159–181.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
 36. Payne VL, Medvedeva O, Legowski E, *et al.*: **Effect of a limited-enforcement intelligent tutoring system in dermatopathology on student errors, goals and solution paths.** *Artif Intell Med.* 2009; **47**(3): 175–197.
[PubMed Abstract](#) | [Publisher Full Text](#)
 37. Plaisant C, Grosjean J, Bederson BB: **SpaceTree: supporting exploration in large node link tree, design evolution and empirical evaluation.** *IEEE Symposium on Information Visualization, 2002. INFOVIS.* 2002; 57–64.
[Publisher Full Text](#)
 38. National Center for Biomedical Ontology: **BioPortal.** 2017.
[Reference Source](#)
 39. National Cancer Institute: **Enterprise Vocabulary Services.** 2017.
[Reference Source](#)
 40. U.S. National Library of Medicine: **Unified Medical Language System® (UMLS®).** 2017.
[Reference Source](#)
 41. Murray T: **Authoring intelligent tutoring systems: An analysis of the state of the art.** *Int J Artif Intell Educ.* 1999; **10**: 98–129.
[Reference Source](#)
 42. Tseytlin E, Castine M, Crowley R: **DomainBuilder – An Authoring System for Visual Classification Tutoring Systems.** In: V. Alevan, J. Kay, J. Mostow (Eds.) *Intelligent Tutoring Systems: 10th International Conference, ITS 2010, Pittsburgh, PA USA, June 14–18, 2010*, Proceedings, Part II Springer Berlin Heidelberg, Berlin, Heidelberg, 2010; 441–442.
[Publisher Full Text](#)
 43. Kirkpatrick DL: **Techniques for Evaluation Training Programs.** *Journal of the American Society of Training Directors.* 1959; **13**: 21–26.
[Reference Source](#)
 44. Holton EF: **The flawed four-level evaluation model.** *Hum Resour Dev Q.* 1996; **7**(1): 5–21.
[Publisher Full Text](#)
 45. Bechhofer S: **OWL: Web Ontology Language.** In: L. Liu, MT. Özsu (Eds.) *Encyclopedia of Database Systems*, Springer US Boston, MA, 2009; 2008–2009.
[Publisher Full Text](#)
 46. University of Maryland Human Computer Interaction Lab: **SpaceTree.** SpaceTree is a novel tree browser that builds on the conventional layout node link diagrams along a single preferred direction. 2017.
[Reference Source](#)
 47. Bodenreider O: **Lexical, terminological and ontological resources for biological text mining.** In: S. Ananiadou, J. McNaught (Eds.) *Text mining for biology and biomedicine*, Artech House, 2006; 43–66.
[Reference Source](#)
 48. Zou Q, Chu WW, Morioka C, *et al.*: **IndexFinder: a method of extracting key concepts from clinical texts for indexing.** *AMIA Annu Symp Proc.* 2003; **2003**: 763–767.
[PubMed Abstract](#) | [Free Full Text](#)
 49. Tag cloud. 2017.
[Reference Source](#)
 50. Tseytlin E, Mitchell K, Legowski E, *et al.*: **NOBLE - Flexible concept recognition for large-scale biomedical natural language processing.** *BMC Bioinformatics.* 2016; **17**: 32.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
 51. OpenSlide. 2017.
[Reference Source](#)
 52. Goode A, Gilbert B, Harkes J, *et al.*: **OpenSlide: A vendor-neutral software foundation for digital pathology.** *J Pathol Inform.* 2013; **4**: 27.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)

53. Romero Lauro G, Cable W, Lesniak A, *et al.*: **Digital pathology consultations- a new era in digital imaging, challenges and practical applications.** *J Digit Imaging.* 2013; **26**(4): 668–677.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
54. Tseytlin E, Linkov F, Castine M, *et al.*: **Dataset 1 in: DomainBuilder: the knowledge authoring system for SlideTutor Intelligent Tutoring system.** *F1000Research.* 2018.
<http://www.doi.org/10.5256/f1000research.16060.d221857>

Open Peer Review

Current Peer Review Status:

Version 1

Reviewer Report 05 March 2019

<https://doi.org/10.5256/f1000research.17539.r44626>

© 2019 Mello-Thoms C. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Claudia Mello-Thoms

University of Sydney, Sydney, NSW, Australia

In this paper, the authors describe the development of two powerful tools to aid in the implementation of Intelligent Tutoring Systems (ITS) in the medical domain. The first tool, named DomainBuilder, combines knowledge authoring, case authoring and validation tasks into a single environment, whereas the second tool, TutorBuilder, allows users to select and customize different aspects of the design on an ITS, such as the user interface, pedagogic and feedback modules, etc.

While the tools, as proposed, are clearly powerful and should be well received by the research community, there are some aspects of the paper that are less clear. For example, in several parts the authors claim that these tools are useful for “general medical ITS”, whereas in other parts they restrict the use of the tools to the domain of Pathology and their SlideTutor ITS. This dichotomy permeates the paper, and it is at times difficult to tell whether the authors are talking about DomainBuilder, TutorBuilder, or SlideTutor. For example, in a subheading reading “Operation”, it seems like the authors are discussing SlideTutor, not the other two systems.

In addition, I would have encouraged the authors to include more “Results” in the appropriate section, as the responses to the questionnaire and the usability study are presented as an aside dataset. Finally, several projects (which have apparently benefited from the development of DomainBuilder and TutorBuilder) are named, such as ODIE and NOBLE, but the reader does not benefit from this information as no description is given of these projects.

Is the work clearly and accurately presented and does it cite the current literature?

No

Is the study design appropriate and is the work technically sound?

Yes

Are sufficient details of methods and analysis provided to allow replication by others?

No

If applicable, is the statistical analysis and its interpretation appropriate?

Not applicable

Are all the source data underlying the results available to ensure full reproducibility?

Yes

Are the conclusions drawn adequately supported by the results?

Partly

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Image perception, observer performance, technology assessment

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Reviewer Report 12 November 2018

<https://doi.org/10.5256/f1000research.17539.r40040>

© 2018 Marusic A. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Ana Marusic

Department of Research in Biomedicine and Health, School of Medicine, University of Split, Split, Croatia

This study presents the creation of the DomainBuilder, an Intelligent Tutoring System (ITS) tool that allows users to apply a systematic workflow to the complex task of content authoring for a medical visual classification IT such as occurs in pathology. The study presents clearly the state of the art in the field, and describes the preliminary studies of the authors to then present the objectives of the study, as well as its innovativeness. In the Methods section, the authors describe in detail the development process for the individual components and processes of the tool and their implementation. The usability study described in the results clearly present the testing with 10 pathologist.

Minor comment: the authors present the mean number of usability problems per user - if means are used to present quantitative data, a measure of variability should also be presented.

The Discussion section presents the summary of the finding, their limitations and possible uses. It will be interesting to follow the success of the tool in actual practice.

Is the work clearly and accurately presented and does it cite the current literature?

Yes

Is the study design appropriate and is the work technically sound?

Yes

Are sufficient details of methods and analysis provided to allow replication by others?

Yes

If applicable, is the statistical analysis and its interpretation appropriate?

Not applicable

Are all the source data underlying the results available to ensure full reproducibility?

Yes

Are the conclusions drawn adequately supported by the results?

Yes

Competing Interests: No competing interests were disclosed.

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

The benefits of publishing with F1000Research:

- Your article is published within days, with no editorial bias
- You can publish traditional articles, null/negative results, case reports, data notes and more
- The peer review process is transparent and collaborative
- Your article is indexed in PubMed after passing peer review
- Dedicated customer support at every stage

For pre-submission enquiries, contact research@f1000.com

F1000Research