Athens Institute

Working Paper No. 2025-2783-28

3 November 2025

Instructional Material Improvement Through Software-Driven Standardization

Tom Wanyama Sr & Tom Wanyama Jr

This paper should be cited as follows:

Wanyama Sr, Tom, Wanyama Jr, Tom (2025) Instructional Material Improvement Through Software-Driven Standardization. Published by the Athens Institute: Working Paper No. 2025-2783-28, 3 November 2025. Pages 1-10

No.: 2025-2783-28
Date: 3 November 2025
DOI:
ISSN: 2241-2891

Previous Working Papers available at: www.atiner.gr/papers.htm

This series began in 2012 and was known as the Conference Paper Series until 2024. In 2025, the series was renamed and is now called the Working Paper Series.

Athens Institute (www.atiner.gr)
2025

Instructional Material Improvement Through Software-Driven Standardization

By Tom Wanyama Sr[*] *& Tom Wanyama Jr*[±]

Instructional materials are essential tools that ensure student engagement within an active learning environment. However, the complexity of producing instructional materials is a labor and time consuming process requiring continuous creative effort from content developers. Categorizing instructional materials as quizzes, lecture presentations, and supplementary content such as notes, and worksheets; we present a software-driven standardization solution to assist during the creative process. We believe that this technological adaptation within the educational ecosystem will not only improve the content creation process but also improve the teaching and learning experience. By leveraging templates and automation processes, we hope to streamline the creation, distribution, and accessibility of instructional materials. With an emphasis on improving the creation of effective and cohesive learning resources in both online and hybrid teaching models. To evaluate the feasibility and impact of our approach, we have developed a working python-based prototype that uses templates to generate educational content based on institutional guidelines, instructor requirements, and student preferences. By documenting the volume of unique and personalized content we have produce across various platforms such as video and file sharing sites; and media types including MP4, PDFs, and other document types; within a given timeframe. We've seen how well software can support institutions in creating standardized materials, how easily instructors assigned with course material development can use templates and automation during the design process, and how the content meets the diverse needs of students. We aim to highlight the potential advantages of template-based automation and its role in the future of course material development. While incorporating some of the principles and concepts used as design benchmarks from Clean Code by Robert Cecil Martin and The Pragmatic Programmer by Andy Hunt and Dave Thomas. These principles and concepts prioritizing three main shareholders that make up both lower and higher education: institutional direction, instructor needs, and student consumption of educational material.

Keywords: *software-driven standardization, educational materials, hybrid/ online teaching models, student engagement, content design and development, software automation*

Introduction

This paper highlights and demonstrates how software-driven standardization can improve the design, development, and distribution of high-quality educational resources. Our goal is to encourage further adoption of technological automation

[*] Associate Professor, McMaster University, Canada.
[±] McMaster University, Canada.

and templating during the content creation process to improve the efficiency of course material development and in turn the teaching and learning experience. The paper and subsequent investigations focus on three aspects that are often found in successful educational systems including institutional direction, instructor resources, and student consumption of educational material. Each of these elements will be referenced while demonstrating the possible benefits of our software solution:

- Institutional Direction: The impact on resource design and development processes.
- Instructor Resources: The usability of templating and functionality of software automation.
- Student Consumption of Educational Material: The quality, quantity, and accessibility of generated content (e.g., presentation slides, quizzes, worksheets, etc.).

We've developed an automated process centered around templating to demonstrate some advantages our solution could have during the course material design and development process. Using principles and concepts such as Standardization, Distribution, and Automation from Clean Code by Robert Cecil Martin as our design benchmarks. With additional inspiration from The Pragmatic Programmer by Andy Hunt and Dave Thomas; we were able to create a system that generates content across various platforms & file types including MP4, PDFs, and other document types within a given timeframe.

Additionally, we demonstrate the flexibility and creative range of the prototype solution by integrating emerging and trending innovations, namely generative AI and short-form content.

We hope to address the challenges of producing high-quality educational resources by demonstrating how the prototype supports content developers. Specifically, shifting effort away from production tasks such as animating, styling, and content presentation; through the use of templating and automation. Allowing creators to focus more on the substance and quality of the material itself.

Prototype

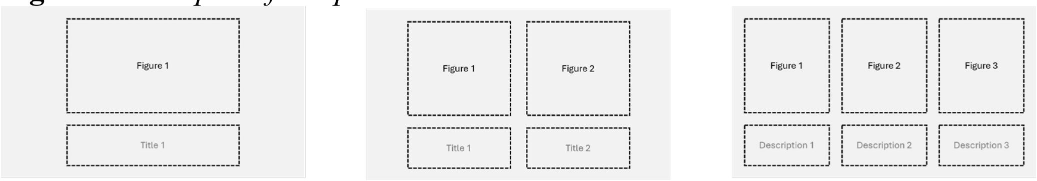
We designed and developed a working prototype to showcase the benefits of a software-driven standardization solution in an academic environment. During the design process, we prioritized templating to relieve course material designers of repetitive tasks such as animating, styling, and content presentation. Diverting that focus to the quality of learning materials. In addition, we used automation to streamline the production process. Maximizing the quantity of learning materials a team can produce.

The prototype uses design templates, a collection of resources, and an instructional document to generate content. With the design templates establishing the rules for organizing and styling information. The resource collection includes images, videos, and other presentable documents. While the instructional document structures the content produced. Details on each of these processes are provided in the sections below. In addition, you can find examples of the content generated at FreeKidsContent.com.

Usability of Templating and Functionality of Automation

Below is an example of three templates that are used to generate a lecture presentation on Topology; the study of shapes and spaces that stay the same under continuous change. These templates include text and resource placeholders that are animated and styled to fit the needs of the user.

Figure 1. *Examples of Templates*



The templates in the figure above were also used to generate a video on the sights and sceneries of Greece. This video combined animations, music, and narration to present the country, its cities, seals, flags, landmarks, and historic sites with short explanations. This exercise was meant to display the tool's flexibility when producing content for different topics, its text-to-speech capabilities, and finally, its consistency in structure across generated content.

Figure 2. *Resource Folder*

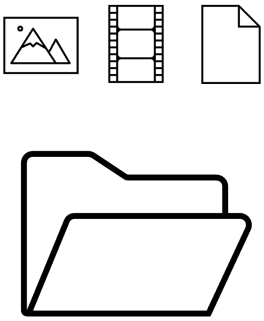
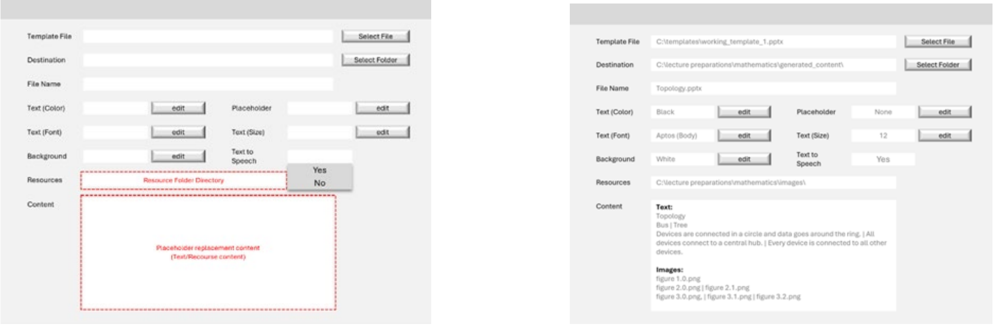


Figure 2 is an illustration of the resource folder and its possible contents. Consisting of images, videos, and other presentable documents used by the prototype's automation protocol.

The prototype has a graphic user interface (GUI) that enables users to interact with the content produced and trigger the automation protocol. In addition to the customizations found in the templates, users can customize styles, colors, and fonts, while also inserting the template placeholders with presentable content, such as text, images, and videos. Flexibility was a key consideration during the development process. Using the prototype, we generated presentation slides, quizzes, worksheets covering a wide range of subjects and topics. Find these at FreeKidsContent.com.

In the first figure below, you'll find an example of the prototype's GUI with all the available specification options. In the second figure you'll find these specifications filled for the Topology lecture presentation export.

Figure 1. *Prototype Graphic User Interface*



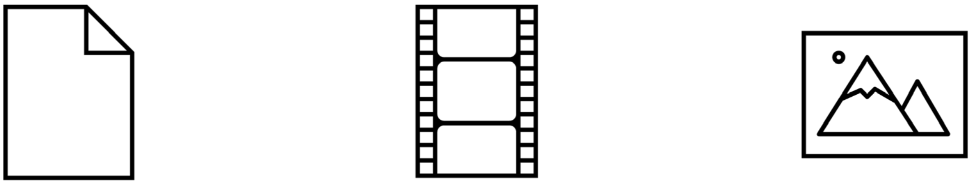
Template File
Destination
File Name
Text (Color)
Placeholder
Text (Font)
Text (Size)
Background
Text to Speech
Yes
No
Resources
Resource Folder Directory
Content
Placeholder replacement content
(Text/Recourse content)

Template File C:\templates\working_template_1.pptx
Destination C:\lecture preparations\mathematics\generated_content\
File Name Topology.pptx
Text (Color) Black
Placeholder None
Text (Font) Aptos (Body)
Text (Size) 12
Background White
Text to Speech Yes
Resources C:\lecture preparations\mathematics\images\
Content
Text:
Topology
Bus | Tree
Devices are connected in a circle and data goes around the ring. | All devices connect to a central hub. | Every device is connected to all other devices.

Images:
figure 1.0.png
figure 2.0.png | figure 2.1.png
figure 3.0.png, | figure 3.1.png | figure 3.2.png

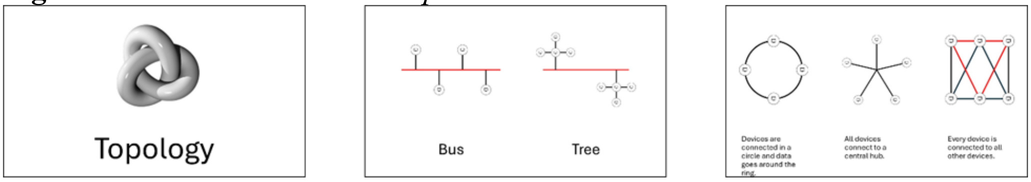
Once all the required inputs are available the automation protocol can be executed. (Note. The template file, content text, and export directory are mandatory inputs.) Our prototype is reliant on the content text. With each line representing a view and the line text separated by a unique character representing placeholders. The program generates views based off a specific template and its associated formatting. Replacing placeholders with text and or presentable file specified in the content text. Once complete, the generated views will be exported to video, image, or simple document formats for consumption.

Figure 2. *Export Formats*



Below is the exported lecture presentation on Topology. Notice how the presentation has characteristics of both the template and the provided content text.

Figure 3. *Lecture Presentation Export*



Topology

Bus
Tree

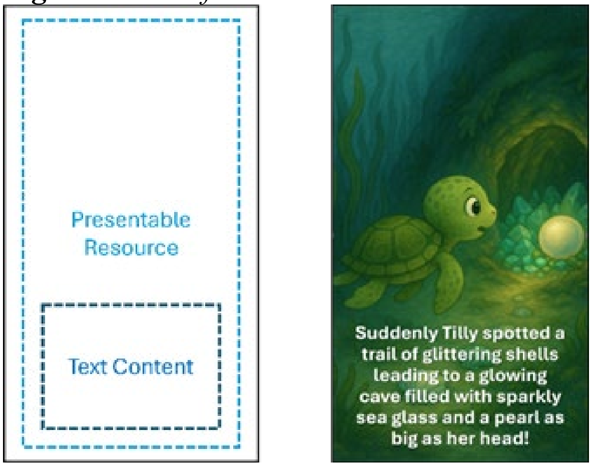
Devices are connected in a circle and data goes around the ring.
All devices connect to a central hub.
Every device is connected to all other devices.

Our solution fosters both standardization, distribution, and automation. It does not fact-check or measure the quality of the content; that responsibility remains with the content design and development team. The prototype ensures that the generated learning materials can be distributed across multiple platforms in a clean, structured, and repeatable manner. We strongly believe that the advantages from the improved content development process will greatly satisfy the needs of our three stakeholders. Institutions: procedural process improvement, instructors: efficiency during content production, and students: accessibility to quality content.

Integration of Trending Innovations

Generative AI and short-form content have been two of the fastest growing trends of the early 2020s. Generative AI is artificial intelligence that creates new content such as: text, images, or music by learning patterns from existing data. While short-form content is concise, engaging material designed for quick consumption. This includes videos under 60 seconds, blog posts under 1,000 words, and quick-hit podcasts. Using the prototype and current generative AI models. We were able to create a short-form video; formatted and styled for platforms such as TikTok, Instagram Reels, and YouTube Shorts.

Figure 4. *Short-form Content*



Suddenly Tilly spotted a trail of glittering shells leading to a glowing cave filled with sparkly sea glass and a pearl as big as her head!

We started by creating a storyline template that included text and image placeholders each with styles and appearance animations. Then using OpenAI's GPT-4o model, we generated a short story about a cartoon character that would appeal to children. Finally, we created the required imagery with 4o Image Generation also by OpenAI. With the content text and image resources obtained. We were able to generate multiple short-from videos each with unique characters, scenic backgrounds, and personalized stories. Refer to the Short-form Content figure above for an example and other generated material at FreeKidsContent.com.

This section highlights how standardization does not have to be rigid; it can integrate current practices in the pursuit of unique and informative content.

Discussion

To successfully integrate technology within any educational ecosystem, we need to address the needs of an institution, its educators and faculty members, as well as the students that make up the student body. By addressing each of these stakeholders, we can mitigate individual challenges, allowing the technology to flourish and enhance the experience for everybody involved.

After closely examining our stakeholders, we decided to prioritize the standardization, distribution, and automation of the learning material creation process. With a focus on the individual or team tasked with developing, maintaining, and delivering content. These priorities further derived from Clean Code by Robert Cecil Martin (Martin, 2008). A book about writing software that is easy to read, maintain, and improve. Supplemented by The Pragmatic Programmer by Andy Hunt and Dave Thomas (Hunt & Thomas, 2019). Which is about becoming a better and more adaptable developer. It gives tips and practices to help programmers write flexible and effective software.

Standardization

In the prototype section of this paper, we introduce templates. These offer a wide range of customizations, including but not limited to content placement, fonts, styles, and colors. Allowing content developers to set the standard of instructional videos, practice worksheets, and other materials used inside and outside of the classroom. Elements such as an institution's name, logos, seals, and colors can be incorporated to solidify an identity within any community and across the internet.

Just as Clean Code (Martin, 2008) and The Pragmatic Programmer (Hunt & Thomas, 2019) stress the importance of consistent, reusable standards in software for clarity and longevity, the template-based standardization creates a flexible framework that ensures identity, usability, and sustainability in instructional materials.

Distribution

Our prototype solution exports to video (MP4), image (JPEG, PNG, GIF, and SVG), and simple document formats such as PDF. This variety does not only allow for easier distribution of content but also increases the range of possible platforms that can host said content.

In the same way that Clean Code and The Pragmatic Programmer highlight practices that keep software versatile and accessible, the prototype's multi-format export capabilities extend the reach of instructional materials, making them easier to share with students, across a wide range of platforms.

Automation

The automation protocol is also introduced in the prototype section of the paper. This python-based script gets templates, specifications, and presentable resources from users, which it uses to generates videos, images, or simple documents.

By aligning with the principles in Clean Code and The Pragmatic Programmer, this automation solution does not only minimize human error but also reinforces repeatability and scalability. Ensuring instructors that the content generated maintains a consistent quality across formats.

By emphasizing standardization, distribution, and automation in the creation of learning materials, the prototype demonstrates how technology can streamline content development while ensuring consistency and accessibility. These practices not only support educators and institutions in maintaining quality but also create scalable solutions that enhance the overall learning experience for students.

Conclusion

In this paper we looked at how software-driven standardization could enhance the design, development, and distribution of educational materials. With the goal of encouraging automation and templating during the content creation process to help course material developers be more efficient. Ultimately improving the teaching and learning experience. We hoped to satisfy the following shareholder needs. Improved resource development processes for institutions, provide a flexible tool for instructors, and quality learning materials for students. Each of these needs were addressed and kept in mind during the prototype's development and within the paper.

Using a python-based prototype we demonstrated how templating can relieve course material developers of repetitive tasks such as animating, styling, and content presentation so that they can focus on the quality of the information presented. While automation streamlined the production of the learning material improving the quantity and distribution of resources available to students. We combined knowledge from Clean Code by Robert Cecil Martin and The Pragmatic Programmer by Andy Hunt and Dave Thomas to develop design benchmarks. These benchmarks currently allow us to generate content across various platforms & media types within a given timeframe. In addition, we integrated emerging technologies and trending content formats into our design process. Namely, generative AI and the short form content format. This was to demonstrate the importance of staying current with the next generation of students while displaying the adaptability and range of the prototype.

At the start of this project, we hoped to tackle the challenges of adopting new technology while producing high-quality educational resources. We strongly believe that assisting resource designers and developers with a tool that shifts effort away from production tasks and allows them to focus on the substance of content is the solution. A solution that tackles the problem of poorly designed educational resources, which are known to reduce the effectiveness of teaching and learning.

As academic institutions continue to explore the future of educational practices, we advocate for standardization with software automation and templating. Adopting software-driven standardization solutions can lead to more consistent, effective, and engaging learning materials. We encourage further research and implementation of these technologies to enhance the academic ecosystem and foster a more impactful educational experience for both instructors and students.

References

Hunt, A., & Thomas, D. (2019). The Pragmatic Programmer. In A. Hunt, & D. Thomas, *The Pragmatic Programmer: From Journeyman to Master.* Boston, MA: Addison-Wesley Professional.

Martin, R. C. (2008). Clean Code. In R. C. Martin, *Clean Code: A Handbook of Agile Software Craftsmanship.* Upper Saddle River, NJ: Prentice Hall.