

DOI: <https://doi.org/10.15276/ict.03.2026.42>

УДК 004.91:004.41

Синхронна семантична проєкція структурованого тексту на слайдову модель

Буц Олександр Володимирович¹⁾

ORCID: <https://orcid.org/0009-0004-4765-3102>; 8404423@stud.op.edu.ua

Оніщенко Тетяна Вікторівна¹⁾

ORCID: <https://orcid.org/0000-0003-0717-6191>; tonishenko@op.edu.ua

¹⁾ Національний університет «Одеська політехніка», пр. Шевченка, 1. Одеса, 65044, Україна

АНОТАЦІЯ

У статті розглянуто науково-прикладну задачу потенційного зниження когнітивного навантаження під час створення презентацій за допомогою перетворення лінійно-структурованого тексту на слайдову модель. Обґрунтовано, що панівна парадигма WYSIWYG змушує автора одночасно формувати зміст і просторове оформлення, тоді як декларативні генератори здебільшого працюють у «сліпому» режимі компіляції та ускладнюють подальше редагування результату. Запропоновано метод синхронної семантичної проєкції, який розглядає вхідний текст як інформаційну модель і безперервно відображає її на просторову модель слайда. Побудовано систему слайдових ролей на основі атомарних синтаксичних елементів і їхніх допустимих комбінацій. Формалізовано функцію семантичної проєкції P для відображення текстових блоків на об'єктну модель PresentationML та функцію F для зв'язку позиції курсора в редакторі з відповідним регіоном попереднього перегляду. Реалізовано двоконтурну архітектуру: клієнтський контур забезпечує введення тексту та синхронну візуалізацію, серверний контур виконує лексичний аналіз, типізацію макетів і генерацію нативного документа PPTX. Порівняльний аналіз із WYSIWYG-редакторами та декларативними генераторами підтвердив переваги розробленої системи за критеріями порогу входження, потенційного зниження когнітивного навантаження та можливості подальшого редагування об'єктного PPTX. Окреслено межі застосування методу: академічні та корпоративні доповіді зі сталою логічною структурою; складна інфографіка та анімація залишаються поза поточною моделлю. У перспективі створена інформаційна модель може слугувати проміжним шаром для автономної генерації звітів за допомогою алгоритмів штучного інтелекту.

Ключові слова: слайдова модель; система слайдових ролей; семантична проєкція; PresentationML; двоконтурна архітектура; когнітивне навантаження; WYSIWYM; генерація PPTX

Для цитування: Буц О. В., Оніщенко Т. В. «Синхронна семантична проєкція структурованого тексту на слайдову модель». *Інформатика. Культура. Техніка.* 2026; Том 3 № 2 (4): 567–576. DOI: [DOI: https://doi.org/10.15276/ict.03.2026.42](https://doi.org/10.15276/ict.03.2026.42)

Актуальність. В умовах сучасного інформаційного середовища – автоматизація створення презентацій на основі структурованого тексту виходить далеко за межі чергового «зручного редактора». Це стає вирішенням проблеми когнітивної ергономіки та продуктивності. Сучасні процеси обміну знаннями охоплюють мільйони фахівців, а це означає, що навіть невелике прискорення циклу «від ідеї до слайда» дає колосальний ефект.

Стрімкий розвиток генеративних моделей штучного інтелекту та інших інформаційних ресурсів – виявляє потребу у новому підході до адаптації контенту. Оскільки домінуючою парадигмою залишаються редактори на основі WYSIWYG (What You See Is What You Get), такі як Microsoft PowerPoint, Google Presentations, проте вони провокують значне розсіювання уваги. Необхідність одночасно формувати зміст презентації та керувати просторовим оформленням збільшує побічне когнітивне навантаження автора й відволікає ресурс робочої пам'яті від самої структури повідомлення [1].

Існуючі альтернативні інструменти, які базуються на основі декларативних мов розмітки, частіше дозволяють працювати лише у «сліпому» режимі, тому користувачі не завжди отримують бажаний результат. Після кожної генерації користувач буде витрачати багато часу на редагування презентації. Розв'язання проблеми просторової розсинхронізації, а саме розриву між введеним контентом та його оформленням – є ключем сильного пришвидшення роботи. Редактор, який надає користувачу попередній перегляд презентації у реальному часі, дозволив би уникнути ітераційних правок. Також деякі з таких інструментів мають великий поріг входження, тому деякі користувачі просто не будуть використовувати такі інструменти.

Аналіз розвитку таких систем дозволяє виділити два майбутні вектори:

- зміщення фокуса від ручного візуального форматування (WYSIWYG) до парадигми семантичної розмітки, коли увага приділяється не дизайну, а самому контенту та його логічній структурі;

- сучасні нейромережі зможуть одразу ідеально формувати структурований текст для адаптації його у слайдову модель. Тобто розроблена система може стати шаром для штучного інтелекту та фізичним документом, відкриється шлях до автономної генерації звітів.

Створення слайдів з наукових текстів досліджується переважно як задача відбору речень і компоновання тез: від навчання на парах у форматі «стаття–презентація» до екстрактивних і абстрактивних моделей SlideGen [8], [9]. Новітні агентні підходи генерують програмний код PresentationML безпосередньо з інструкції [10], проте працюють у режимі разової компіляції без неперервного зв'язку між курсором автора та моделлю слайда. Парадигма WYSIWYM задає принципово інший інтерфейсний контракт. Автор оперує змістом – система підтримує відповідність між інформаційною та просторовою моделями.

Метою дослідження є створення інформаційної моделі та оптимізованої програмної архітектури, яка дозволить користувачу легко та швидко перетворювати лінійно-структурований текст у слайдову модель. Досягнення цієї мети базується на побудові моделі слайдових ролей та формалізації правил семантичної проєкції тексту на візуальні елементи, Практична реалізація охоплює розроблення функції просторової синхронізації, де необхідно пов'язати курсор у текстовому редакторі з відповідним регіоном на клієнтському контурі прев'ю. Крім того, архітектура має передбачати створення серверного контуру для повноцінної генерації документів формату PPTX на базі сформованої моделі. Завершальним етапом роботи є порівняльний аналіз розробленого рішення з класичними та альтернативними інструментами.

Отже, виникає потреба в новому класі систем, які б поєднували простоту написання звичайного лінійно-структурованого тексту з миттєвим відображенням готового слайда. Для навчальних і доповідних матеріалів ефективність визначається не декоративним навантаженням слайда, а узгодженістю вербального ряду зі структурованим візуальним поданням; зайве просторове форматування створює стороннє опрацювання й заважає побудові ментальної моделі змісту [3].

Щоб реалізувати такий механізм – необхідний перехід до концепції синхронної семантичної проєкції. Цей підхід вимагає розроблення спеціальної архітектури, де лінійний текст сприймався як інформаційна модель, що безперервно проєктується на просторову модель слайда.

Загальна архітектура системи. Двоконтурну архітектуру спроектовано в логіці людиноорієнтованого розроблення інтерактивних систем: ітерація «намір автора → видимий стан слайда → уточнення тексту» є основним циклом використання, а не допоміжною операцією експорту [7]. Для забезпечення ефективної роботи механізму синхронної семантичної проєкції, запропоновано чітко розділити відповідальність.

Систему реалізовано як веб-додаток, що функціонує на базі двоконтурної архітектурної моделі (структурну схему взаємодії контурів наведено на Рис. 1):

- Клієнтський контур відповідає за безпосередню взаємодію з користувачем на базі бібліотеки React. Завдяки компонентному підходу можливо ізолювати логіку редактора від попереднього перегляду. Практична реалізація Функції Ф для просторової синхронізації складається з використання механізму реактивного управління станом. Під час кожної зміни курсору, або модифікації тексту – система локально обробляє дані та перемальовує відповідний просторовий регіон без перезавантаження сторінки. Таким чином забезпечується ефект безшовної взаємодії та миттєвий відгук.

- Серверний контур виконує роль обчислювального ядра і розроблений мовою Python з використанням фреймворку FastAPI. Цей контур приймає текстову модель від клієнта, здійснює лексичний аналіз, застосовує правила типізації слайдів та виконує фізичну

генерацію документа. Застосування парадигми асинхронного програмування дозволяє серверу обробляти множинні паралельні запити на генерацію презентацій без блокування основного потоку виконання. Взаємодія між клієнтським та серверним контурами реалізується через архітектурний стиль REST API. Основним комунікаційним вузлом виступає маршрут «/generate», який приймає HTTP POST-запит. Для формалізації вхідної інформаційної моделі на етапі запиту клієнта використано бібліотеку Pydantic – за допомогою якої можна виокремити вхідні дані (назва презентації та сам контент). Після перетворення вхідних даних у форматі JSON, сервер поверне потік з презентацією у тлі HTTP-відповіді, що дозволить клієнтському браузеру миттєво ініціювати завантаження згенерованого PPTX-файлу.

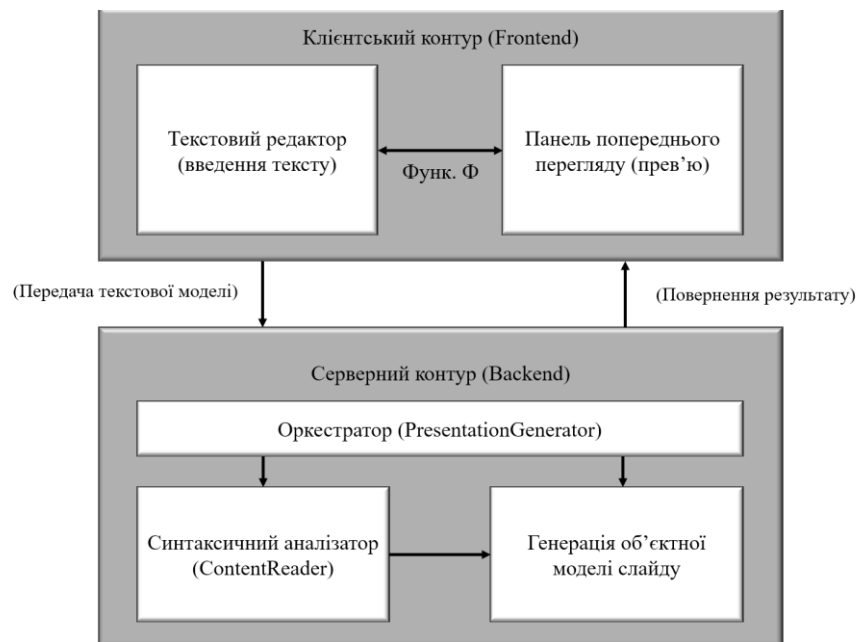


Рис. 1. Загальна двоконтурна архітектура системи семантичної проєкції

Джерело: розроблено авторами

Таке архітектурне рішення забезпечує повну кросплатформність: користувач може створювати презентації на будь-якому пристрої (ПК, планшеті чи смартфоні) незалежно від операційної системи, потребуючи для роботи лише сучасний веб-браузер.

Формат інформаційної моделі та система слайдових ролей. Для реалізації механізму семантичної проєкції вимагається формалізація вхідних даних. У запропонованій архітектурі вхідний текст декомпонується на цільні блоки, де блоки складаються з базових атомарних елементів. Слід уточнити, що термін «семантична проєкція», в контексті даного дослідження, не передбачає автоматичного семантичного аналізу природної мови. Натомість семантична роль кожного елемента (заголовок, підзаголовок, текст) явно задається автором через структурований (синтаксичний) маркер. Сама ж проєкція полягає у збереженні цієї ролі під час переходу від лінійного текстового подання до об'єктної слайдової моделі (макетів PresentationML). Кожен елемент розпізнається системою за відповідним синтаксичним маркером. Визначення основних мінімальних елементів зазначено у Таблиці 1.

Але окремі атомарні елементи не утворюють самостійно повну модель слайда. Наступним рівнем формалізації є визначення комбінацій. Тільки згруповані набори примітивів формують кінцеву систему слайдових ролей. Залежно від того, яка комбінація елементів зчитана з поточного блоку тексту, система визначає тип майбутнього слайда (наприклад, титульний слайд складається з заголовку та підзаголовку).

Перелік комбінацій базових елементів і відповідні типи слайдових моделей наведено у Таблиці 2.

Таблиця 1. Визначення основних мінімальних елементів

№	Елемент	Короткий опис (семантична роль)	Синтаксичний маркер (префікс)
1	Заголовок	Головний заголовок презентації і, одночасно, є заголовком першого рівня.	#
2	Підзаголовок	Підзаголовок на слайді – є заголовком другого рівня.	##
3	Звичайний текст	Базовий інформаційний блок, який обробляється як стандартний абзац	префікс відсутній
4	Елемент списку	Рядок контенту, що визначає елемент маркованого або нумерованого списку на поточному слайді	- для маркованого списку 1. для нумерованого списку
5	Зображення	Медійний блок, що містить інформацію про зображення.	![підпис](url/base64)

Джерело: розроблено авторами

Таблиця 2. Модель слайдових ролей та їх компонентний склад

Тип слайду	Обов'язкові елементи
Титульний слайд (Title Slide)	Заголовок + Підзаголовок
Слайд-роздільник (Section Header)	Заголовок
Текстовий слайд (Bullet Slide)	Підзаголовок + Текст
Медіа слайд (Image Slide)	Підзаголовок + Зображення
Комбінований слайд (Split Layout)	Підзаголовок + Текст + Зображення
Слайд-перелік (List Slide)	Підзаголовок + Елементи списку

Джерело: розроблено авторами

Правила семантичної проєкції та методи генерації слайдових структур. Для процесу семантичної проєкції необхідне розуміння архітектури цільового документа. Згенерована інформаційна модель не відповідає за відображення пікселів на екрані, а формує складну ієрархічну структуру на базі стандарту PresentationML.

У реалізації серверного контуру генерації документів було використано бібліотеку python-pptx. Цей інструмент надає API для створення та модифікації презентацій на рівні ООП, таким чином немає необхідності у низькорівневій роботі з сирими XML-файлами та ZIP-архівами.

Цільовий артефакт формується як документ Office Open XML, де презентація описується ієрархією розмітки PresentationML: зразок слайда, макет і примірник слайда [6]. Цій архітектурі відповідає трирівневий граф, зображений на Рис. 2. Проєкція текстових блоків на слайд відбувається за рахунок звернення до існуючих елементів макету та модифікації їхніх текстових фреймів.

Незважаючи на працездатність цього API, пряме використання у бізнес-логіці програми є недоцільним, бо це призводить до високої зв'язності. Щоб парсер не залежав від індексів макетів PresentationML – логіку винесено в ієрархію адаптерів: кожен тип слайда інкапсулює відображення своєї ролі на відповідний макет, що відповідає патерну Adapter [5] та реалізується через поліморфний клас Slide і його спадкоємців (TitleSlide, BulletSlide та інші).

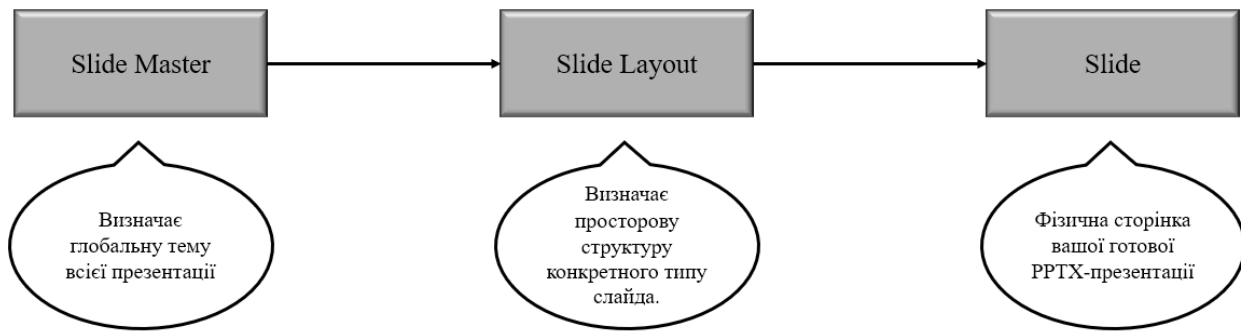


Рис. 2. Трирівнева архітектура об'єктної моделі формату PresentationML

Джерело: розроблено авторами

Управлінням всім життєвим циклом створення документа на серверному боці займається модуль-оркестратор (клас PresentationGenerator). Його завданням є ініціалізація базової просторової моделі.

Оркестратор звертається до модуля синтаксичного аналізу (ContentReader) для обробки вхідного потоку текстових даних, саме цей компонент виконує розбиття вхідного рядка: спочатку весь масив розбивається на окремі слайди за допомогою роздільника «---». Далі алгоритм виконує аналіз кожного блоку, ідентифікуючи атомарні елементи за допомогою префіксів та групуючи їх у відповідні масиви. На основі цих масивів система повертає типізований об'єкт-адаптер потрібного слайда.

Така архітектура забезпечує жорстке розділення відповідальності. Модуль синтаксичного аналізу відповідає лише за виокремленням моделей за типом, а шар Slide – за правильне розміщення даних, повністю приховуючи складність роботи з OpenXML.

Коли оркестратор ітеративно обійде усі слайди та викличе для кожного свої семантичні правила проєкції настане етап віддачі результату. Замість збереження файлу на фізичний носій, модуль збереже зібрану структуру в оперативну пам'ять у вигляді байтового потоку. Такий підхід зменшить витрати часу на операції збереження та зчитування дискової підсистеми, і в результаті підвищиться пропускна здатність серверного контуру і зменшиться затримка при віддачі готової презентації користувачу.

Наглядним буде математичний опис для семантичної проєкції тексту в об'єктну модель. Нехай T – це вхідний текст користувача, який необхідно опрацювати. Процес лексичного аналізу сформулює на основі T послідовність текстових блоків B , де b_i – це блок тексту, який відповідає одному слайду, n – загальна кількість слайдів.

$$B = (b_1, b_2, \dots, b_n) \quad (1)$$

Кожен текстовий блок b_i проходить порядковий аналіз через функцію типізації f_{type} . Ця функція аналізує наявність синтаксичних префіксів та повертає тип макету з доступної множини підтримуваних ролей слайдів:

$$L(s_i) = f_{type}(b_i) \in \{TitleSlide, BulletSlide, \dots, ListSlide\}. \quad (2)$$

Фінальна модель презентації S – це послідовність слайдів на основі сформульованих об'єктів-адаптерів:

$$S = (s_1, s_2, \dots, s_n). \quad (3)$$

Загальний процес перетворення, зводиться до функції семантичної проєкції Π . Де Π – це комплексна операція маршрутизації даних, яка гарантує, що кожен i -тий текстовий блок b_i буде спроектований у відповідний контейнер-заповнювач i -го слайду s_i .

$$\Pi: B \rightarrow S, \Pi(b_i) = s_i. \quad (4)$$

З алгоритмічної точки зору складність моделі синтаксичного аналізу становить $O(N)$, де N – це кількість символів у вхідному тексті. Текст обробляється за один прохід, без використання рекурсії.

Синхронна візуалізація та механізм зворотного зв'язку. Серверна генерація та мапінг на об'єктну модель вирішують проблему формування валідного документа. Проте також необхідно знизити когнітивне навантаження на автора – тому треба розробити ергономічний клієнтський інтерфейс. Принцип безпосереднього маніпулювання передбачає видимість об'єкта інтересу, швидкі оборотні дії та заміну складної командної мови на маніпуляцію змістом; синхронізація курсора з регіоном слайда реалізує саме цей клас зворотного зв'язку [2]. Тому необхідним є створення механізму синхронної проєкції, який формалізується через Функцію Φ .

Функція Φ представляє алгоритм встановлення зв'язку між масивом тексту в редакторі та просторовою моделлю у вікні попереднього перегляду.

Цей механізм працює на основі трьох етапів:

1. Відстеження контексту за допомогою використання слухачів подій у клієнтському редакторі для визначення позиції курсору.
2. Локальна декомпозиція у спрощеній версії на стороні клієнта. Система обчислює індекс слайду та зіставляє його зі слайдом, відокремленим за допомогою «---».
3. Відображення виявленого слайду.

Якщо сформулювати математичний опис для Функції Φ , то аргументами функції була б множина B , а також курсор c . А результатом функції був би кортеж зі згенерованого слайду s_i , та r – просторового регіону всередині слайда s_i (наприклад, $r \in \{TitleSlide, BulletSlide, \dots, ListSlide\}$), який редагується на той момент часу.

$$\Phi(B, c) = (s_i, r). \quad (5)$$

Отримавши кортеж (s_i, r) , клієнтський інтерфейс здійснював навігацію до об'єкта s_i . Таким чином би створювався ефект безшовної взаємодії, повністю усуваючий розрив між лінійним кодом та його візуальним поданням.

Практичну реалізацію клієнтського інтерфейсу, який поєднує зону введення розмітки та панель синхронного попереднього перегляду, наведено на Рис. 3.

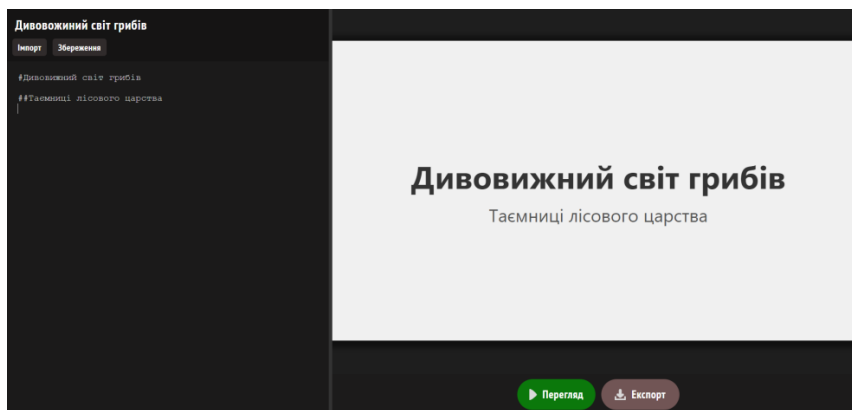


Рис. 3. Клієнтський інтерфейс розробленої системи
Джерело: розроблено авторами

Порівняльний аналіз створеної системи. Зважаючи на наведені архітектурні особливості, доцільно узагальнити загальний баланс сильних та слабких сторін запропонованого програмного забезпечення.

Неявною перевагою розробленої системи над візуальними редакторами є легкість наскрізного аналізу контенту. Замість того, щоб продивлятися кожен слайд та перераховувати інформацію – можна одразу побачити весь текст у редакторі одночасно. Також з'являється можливість легкого редагування помилок (на етапі до серіалізації у PPTX) за допомогою зовнішніх інструментів, наприклад, штучного інтелекту.

Перевагою системи над декларативними генераторами є формат видачі файлу. Найпопулярніші інструменти для створення презентації найчастіше використовують експорт

до формату PDF чи HTML. Це створює бар'єр до подальшої роботи над презентацією, оскільки, потрібно буде повертатися до перекомпіляції вихідного коду, наприклад, такими є Reveal.js, Marp. Розроблена система генерує у форматі PresentationML, що вирішує дану проблему, і кінцевий споживач може редагувати у будь якому редакторі свою презентацію.

Згідно з виконанням загального порівняння було сформовано Таблицю 3.

Таблиця 3. Аналіз візуальних та текстових редакторів з розробленою системою

Критерій оцінювання	WYSIWYG-редактори (PowerPoint)	Декларативні компілятори (Marp, Sliddev, Pandoc);	Розроблена система
Принцип форматування	Одночасне ручне позиціонування та написання тексту	Суворе відокремлення дизайну від тексту через код	Суворе відокремлення дизайну від тексту через розмітку
Вимоги до навичок користувача	Базова комп'ютерна грамотність	Знання спеціального синтаксису розмітки або програмування	Базова комп'ютерна грамотність та знання базових префіксів
Фокус уваги автора	Розподілений (зміст + візуальне оформлення)	Зосереджений виключно на коді (без прев'ю)	Зосереджений на змісті (із синхронним прев'ю)
Синхронний візуальний зв'язок	Наявний	Відсутній (потребує перекомпіляції)	Наявний (миттєве оновлення)
Позиціонування візуальних елементів	Абсолютне (вільне малювання та розміщення)	Програмне (задається параметрами у коді)	Шаблонне (жорстко прив'язане до макетів PresentationML)
Редагування кінцевого файлу	Пряме візуальне редагування об'єктів	Ускладнене (потребує редагування вихідного коду)	Пряме візуальне редагування об'єктів (у PPTX)
Цільовий формат експорту	Нативний (.pptx)	Статичний (.pdf, .html)	Нативний (.pptx)

Джерело: розроблено авторами

Аналіз функціональних характеристик, наведених у таблиці, підтверджує, що розроблена система поєднує мінімальні вимоги до навичок користувача із зосередженням фокусу уваги виключно на змісті. Це створює передумови для очікуваного зниження когнітивного навантаження порівняно з класичними візуальними редакторами. У сучасній редакції теорії когнітивного навантаження розрізняють внутрішнє, зовнішнє та доречне навантаження; інтерфейс, який знімає з автора операції компонування, зменшує зовнішнє навантаження й залишає ресурс для опрацювання самої структури повідомлення [4].

Також було виконано пілотне експериментальне випробування: як вхідні дані було сформульовано тестовий текст за допомогою штучного інтелекту (ШІ використовувався виключно для генерації тестового матеріалу, а не для обробки результатів чи розрахунків, текст та відео з тестуванням розміщено у репозиторії проєкту). А також створено дві презентації на основі одного тексту і виконано заміри часу. Результати замірів часу є такими: при створенні презентації в ручному форматі у WYSIWYG-редакторі вийшло 1 хвилину 34 секунди, а при генерації через розроблений редактор створення зайняло 1 хвилину 10 секунд. Отримані дані фіксують скорочення операційного часу приблизно на 25% вже на етапі базового прототипування. Але слід зазначити, що отриманий результат є попереднім і потребує подальшої перевірки на більшій кількості користувачів та різноманітних тестових презентаціях.

Можна виділити два ключових фактори, які забезпечили таке прискорення:

- Мінімізовано контекстні перемикання. Достатньо було один раз скопіювати весь текст, і далі його редагувати. Необхідно було лише розставити префікси для виокремлення заголовків, а також розставити між слайдами роздільник «---».

- Автоматизовано просторове форматування. Робота у візуальному редакторі супроводжувалася бічними подіями: створенням слайдів, обранням макету, на відміну від створеного редактору, де необхідно було лише розставити префікси у тексті.

Але, незважаючи на доведену ефективність у створенні презентацій – розроблена система має певні архітектурні обмеження на відміну від класичних WYSIWYG-редакторів. Оскільки проєкція базується на жорстко заданій системі макетів – система не призначена для створення складної інфографіки, вільного малювання, абсолютного позиціонування об'єктів. Використання новоствореного інструменту буде доцільним лише у академічних/корпоративних доповідях, де необхідна логічна структура та контент, а не унікальний художній дизайн кожного слайду. Також бібліотека `python-pptx` не підтримує створення анімацій, що також є обмеженням для системи.

Розроблена система не є фінальним рішенням і має потенціал для подальшого вдосконалення.

Можна виокремити декілька ключових векторів.

По-перше, шляхом проведення глибокого аналізу роботи бібліотеки `python-pptx`, а також розширення функціоналу, можливе впровадження опціональних стилів форматування. Це охоплюватиме розширене форматування тексту (зміна розміру, кольору, шрифту та його виділення), зміна теми слайда (зміна заднього фону, основних кольорів) та введення механізмів анімації.

По-друге, вагомим кроком стане інтеграція методів обробки природної мови (NLP – Natural Language Processing). Застосування даного підходу дозволить системі автоматично аналізувати текст, виокремлюючи елементи.

По-третє, найбільш актуальним та перспективним напрямом є вбудування систем штучного інтелекту у розроблену архітектуру, що дозволить користувачу швидко вносити зміни у контент, змінюючи стиль і виправляючи помилки.

Висновки. Отже, у роботі було вирішено актуальну науково-прикладну задачу мінімізації рутинних операцій, що створює передумови для зниження когнітивного навантаження під час створення презентаційних матеріалів за допомогою розробки оптимізованої двоконтурної архітектури для генерації слайдів на основі структурованого тексту. Була побудована система слайдових ролей та було сформовано правила семантичної проєкції тексту. Реалізовано алгоритм візуального зворотного зв'язку на основі Функції Ф для створення зв'язку між текстовим редактором та попереднім переглядом слайда. В результаті спроектовано системний контур, який формує презентацію формату PPTX

Було проведено порівняльний аналіз системи з існуючими рішеннями, за результатами якого виявлено переваги у низькому порозі входження, теоретичному зниженні когнітивного навантаження, можливості вільного подальшого редагування, а також перелічено недоліки через обмеження бібліотеки та загальне спрощення системи. Під час пілотного експериментального випробування з замірами витраченого часу зафіксовано скорочення часу створення презентації на 25 %, однак ці результати потребують перевірки на більшій вибірці користувачів. Окреслено архітектурні обмеження системи, що позиціонує її як інструмент для академічних та корпоративних доповідей зі сталою логічною структурою.

Конфлікт інтересів: Автори декларують, що не мають конфлікту інтересів, зокрема фінансового, особистого, авторського чи будь-якого іншого характеру, який міг би вплинути на дослідження, а також на результати, опубліковані в цій статті

Фінансування: Дослідження проводилося без фінансової підтримки

Доступність даних: Код моделювання, використаний для одержання наведених результатів, подано у відкритому репозиторії [<https://github.com/MrTvistYT/md2pptx>]

Використання засобів штучного інтелекту: Під час підготовки рукопису використовувався інструмент Gemini 3.1 Про штучного інтелекту (ШІ) для граматичної та стилістичної перевірки тексту. Також ШІ було використано виключно для генерації тестового вхідного матеріалу (тестового тексту) під час проведення експерименту. Усі фрагменти, опрацьовані за його допомогою, були перевірені й відредаговані автором. ШІ не використовувався для генерації тексту, формул і таблиць, формування наукових положень, отримання результатів або формулювання висновків. Наукові положення, результати та висновки є власним інтелектуальним внеском авторів

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Sweller, J. “Cognitive load during problem solving: Effects on learning”. *Cognitive Science*. 1988; 12 (2): 257–285. DOI: https://doi.org/10.1207/s15516709cog1202_4.
2. Shneiderman, B. “Direct manipulation: a step beyond programming languages”. *Computer*. 1983; 16 (8): 57–69. DOI: <https://doi.org/10.1109/MC.1983.1654471>.
3. Mayer, R. E. “Multimedia learning”. 3rd ed. *Cambridge University Press*. 2020. DOI: <https://doi.org/10.1017/9781108894333>.
4. Sweller, J., van Merriënboer, J. J. G. & Paas, F. “Cognitive architecture and instructional design: 20 years later”. *Educational Psychology Review*. 2019; 31: 261–292. DOI: <https://doi.org/10.1007/s10648-019-09465-5>.
5. Gamma, E., Helm, R., Johnson, R. & Vlissides, J. “Design patterns: elements of reusable object-oriented software”. *Addison-Wesley*. 1995. DOI: <https://doi.org/10.1007/978-3-642-59412-0>.
6. “ISO/IEC 29500-1:2016. Information technology – Document description and processing languages – Office Open XML File Formats – Part 1: Fundamentals and Markup Language Reference”. *ISO*. 2016. DOI: <https://doi.org/10.3403/30283512>.
7. ISO. “ISO 9241-210:2019. Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems”. *ISO*. 2019. DOI: <https://doi.org/10.3403/30388991u>.
8. Sefid, A., Mitra, P. & Giles, C. L. “SlideGen: an abstractive section-based slide generator for scholarly documents”. *Proceedings of the 21st ACM Symposium on Document Engineering (DocEng '21)*. 2021. p. 1–4. DOI: <https://doi.org/10.1145/3469096.3474939>.
9. Sefid, A., Wu, J., Mitra, P. & Giles, C. L. “Extractive research slide generation using windowed labeling ranking”. *Proceedings of the Second Workshop on Scholarly Document Processing*. 2021. p. 91–96. DOI: <https://doi.org/10.18653/v1/2021.sdp-1.11>.
10. Ge, J., Wang, Z. Z., Zhou, X., Peng, Y.-H., Subramanian, S., Tan, Q., Sap, M., Suhr, A., Fried, D., Neubig, G. & Darrell, T. “AutoPresent: designing structured visuals from scratch”. *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2025. p. 2902–2911. DOI: <https://doi.org/10.1109/CVPR52734.2025.00276>.

Отримано 03.09.2026

Отримано після перегляду 10.10.2026

Прийнято 13.10.2026

DOI: <https://doi.org/10.15276/ict.03.2026.42>

UDC 004.91:004.41

Synchronous semantic projection of structured text onto a slide model

Oleksandr V. Buts¹⁾

ORCID: <https://orcid.org/0009-0004-4765-3102>; 8404423@stud.op.edu.ua

Tetyana V. Onishenko¹⁾

ORCID: <https://orcid.org/0000-0003-0717-6191>; tonishenko@op.edu.ua

¹⁾ Odesa Polytechnic National University, 1, Shevchenko Av. Odesa, 65044, Ukraine

ABSTRACT

This paper focuses on the applied scientific problem of mitigating potential cognitive load during presentation design by transforming linearly structured text into a slide-based model. It is justified that the prevailing WYSIWYG paradigm forces authors to simultaneously manage content formulation and spatial layout, whereas declarative generators typically operate in a “blind” compilation mode complicating the subsequent editing of the output. A method of synchronous semantic projection is proposed. This method treats the input text as an informational model and continuously maps it onto the spatial structure of a slide. A system of slide roles has been constructed based on atomic syntactic elements and their permissible combinations. The semantic projection function “П” is formalized to map text blocks onto the PresentationML object model, along with the function “Ф” for linking the editor cursor position to the corresponding preview region. A two-tier architecture was implemented, comprising a client layer responsible for text entry and synchronous rendering, and a server layer dedicated to lexical analysis, layout typing, and native PPTX document generation. A comparative analysis against traditional WYSIWYG editors and declarative generators demonstrates the superiority of the developed system in terms of entry threshold, potential cognitive load reduction, and post-editing flexibility of the object-level PPTX. The boundaries of the proposed method are outlined, confining its application primarily to academic and corporate presentations with a stable logical structure; complex infographics and advanced animations remain outside the current model scope. Looking ahead, this informational model can serve as an intermediate semantic layer for autonomous report generation driven by artificial intelligence algorithms.

Keywords: Slide model; system of slide roles; semantic projection; PresentationML; dual-loop architecture; cognitive load; WYSIWYM; PPTX generation

ІНФОРМАЦІЯ ПРО АВТОРІВ



Буц Олександр Володимирович – здобувач вищої освіти, кафедра Інженерії програмного забезпечення. Національний університет «Одеська політехніка», пр. Шевченка, 1. Одеса, 65044, Україна
Галузь досліджень: програмна інженерія

Oleksandr V. Buts – student, Software Engineering Department. Odesa Polytechnic National University, 1, Shevchenko Av. Odesa, 65044, Ukraine
ORCID: <https://orcid.org/0009-0004-4765-3102>; 8404423@stud.op.edu.ua
Research field: software engineering



Оніщенко Тетяна Вікторівна - старший викладач, кафедра Інженерії програмного забезпечення. Національний університет «Одеська політехніка», пр. Шевченка, 1. Одеса, 65044, Україна
Галузь досліджень: програмна інженерія

Tetyana V. Onishenko - Senior Lecturer, Software Engineering Department. Odesa Polytechnic National University, 1, Shevchenko Av. Odesa, 65044, Ukraine
ORCID: <https://orcid.org/0000-0003-0717-6191>; tonishenko@op.edu.ua
Research field: software engineering