УДК 004.8

ПОШУКОВА СИСТЕМА НАУКОВИХ МАТЕРІАЛІВ УКРАЇНСЬКОЮ МОВОЮ ІЗ СОЦІАЛЬНОЮ СКЛАДОВОЮ

А.М. Глибовець

Національний університет «Києво-Могилянська академія»,
04655, Київ, вул. Г. Сковороди, 2.
Тел.: 425 0245, факс.: 425-02-45.
E-mail: andriy@glybovets.com.ua

Описується архітектура пошукової системи наукових матеріалів українською мовою із соціальною складовою. Розглянуто та обгрунтовано вибір технологій для створення такої системи, описано її функціональність. Представлено алгоритм для виділення з PDF документів наукових статей логічних частин.

The paper describes the architecture of search engine scientific materials in Ukrainian language with the social component. Considered and the choice of technologies to develop a system described by its functionality. The algorithm for the selection scientific articles logical parts in the PDF documents.

Вступ

За умов інформаційного перенасичення сучасного суспільства надзвичайно важливими є фільтрування та пошук релевантної інформації відповідно до потреб особи. Саме тому пошукові системи (ПС) набули такої популярності. Однак ПС загального призначення нині уже недостатньо [1]. Для вирішення певних задач до наявних систем додаються уточнюючі функції, або ж створюються спеціалізовані системи: для пошуку за форматом файлів, за певною категорією знань тощо. Одним з видів таких систем є ПС наукових матеріалів.

На сьогодні існує декілька потужних систем пошуку цього типу: Google Scholar, Scirus, ScienceDirect. Однак, більшість з них не підтримує пошуку українською мовою, а інші не мають соціальної складової, яка б дозволила користувачам системи спілкуватись між собою, обмінюватись інформацією, додавати нові матеріали, рецензії тощо.

Зважаючи на ці зауваження, поставлено задачу розробки багатофункціональної моделі ПС наукових матеріалів українською мовою з соціальною складовою (ПСНМУМ), а також реалізація алгоритму поділу документу на логічні зони (заголовок, автор, використана література та УДК), що дозволить перейти до створення робочого прототипу такої ПС.

Уточнимо постановку задачі. Окрім моделі потрібно: підібрати технології реалізації та обгрунтувати переваги і недоліки вибору; визначити список необхідних функцій системи, а також описати логіку їх реалізації; розробити та реалізувати алгоритм для розбору матеріалів наукових статей у форматі PDF на логічні зони.

Архітектура та вибір технологій реалізації

Пошукова система наукових матеріалів складається з серверної та клієнтської частини.

Серверна частина містить спеціалізовані модулі, кожен з яких відповідає за окрему функцію пошукової системи, а саме: індексації, класифікації, збору посилань, клієнтський модуль (front end).

Клієнтська частина, разом із клієнтським модулем на сервері, відповідає за взаємодію з користувачем на стороні клієнта. Вона надає доступ до функцій пошуку, обміну повідомленнями, соціальних функцій тощо і містить інтерфейс для реєстрації та аутентифікації користувачів у системі.

При виборі технологій для реалізації проекту враховувалось кілька основних вимог до майбутньої системи: масштабованість, ефективність, надійність. З огляду на вищезазначені вимоги, обрано такий стек технологій.

Серверна частина буде реалізована на мові Java. Серед переваг можна видылити: легкість у підтримці та розвитку проекту; наявність рішень з відкритим кодом для організації пошукової системи (Hadoop, Lucene, Nutch, Solr); наявність готових модулів, реалізованих силами автора роботи та інших студентів НаУКМА, що будуть використані для майбутнього проекту. Очевидним недоліком бачиться відносно низька швидкодія.

Для хостингу ми обираємо Google App Engine [2], а індексацію та основні функції інформаційного пошуку будемо робити на базі бібілотеки Lucene [3]. Основними чинниками обрання останньої були висока швидкодія, можливий пошук за полями (заголовок, автори), відкритий код і здатність індексувати PDF документи (більшість наукових матеріалів саме в цьому форматі).

Проведення фронт-енду на сервері нам бачиться на базі JSF (JavaServer Faces) за можливості прямого доступу до сервера (збільшує швидкодію) та наявності детальної документації.

© А.М. Глибовець, 2014

Багата функціональність, велика кількість готових рішень широка спільнота розробників javascript/jQuery визначила її обрання для клієнтської частини.

Функціональність системи

Початкова база документів для майбутньої пошукової системи буде відібрана з найпотужнішої на сьогодні онлайн бібліотеки наукових матеріалів українською мовою – Національної бібліотеки України імені І. В. Вернадського [4]. Надалі ця база поповнюватиметься за допомогою спеціального модуля (робота). Він діятиме за таким алгоритмом:

1) подібно до звичайних Web-браузерів, робот відвідує сторінку за посиланням і копіює її вміст;

2) в отриманому html-коді знаходить внутрішні та зовнішні посилання, шляхом пошуку входжень «*http*://», «*href*=» тощо. Отримані посилання робот вносить в базу даних;

3) виділяє серед html-коду власне текст, шляхом видалення усіх тегів та іншого не релевантного «сміття». Отриманий текст зберігається для подальшого використання.

Окрім, робот має проводити циклічну перевірку уже доданих документів на наявність змін. З певним часовим інтервалом, він має повторно відвідувати сторінки-джерела кожного документа. Якщо сторінка більше не доступна, або матеріал за посиланням був видалений, робот має через деякий час видаляти посилання з бази системи. Або ж, якщо контент сторінки змінився, він має проводити його повторну обробку, тобто знову копіювати вміст, знаходити посилання та обробляти текст.

Очевидно, що для роботи даного модуля, адміністратор має створити початкову базу посилань на популярні наукові ресурси. Варто зазначити, що даний пункт не є обов'язковим для прототипу системи і може бути реалізований пізніше, під час розширення системи.

Завдання індексації документів ми перекладаємо на бібліотеку Lucene. Вона надає широкий набір засобів для створення інвертованого індексу. Ця бібліотека допускає розширення функціональності через перевизначення функціональних одиниць. Розширюючи функціональність бібліотеки, розроблено клас ScientificWorkIndexator, що відповідає за процес індексації [5].

Для створення індексу за зонами, необхідно мати інструмент для виділення окремих полів у документах (в першу чергу імена авторів та назви статті). Основна проблема полягає у тому, що не існує єдиного стандарту для оформлення документів, а тому така задача не є тривіальною. Алгоритм розбиття документа на логічні зони, а також його реалізацію представлено далі.

Завданням майбутньої пошукової системи є пошук саме наукових матеріалів. Тому перед індексацією документа, він має бути перевірений на науковість і за результатами цієї перевірки або доданий до бази, або відкинутий як такий, що не відповідає вимогам системи і потребам користувача. Для цього буде використаний алгоритм, описаний у [6].

Також кожен документ має бути віднесений до однієї або кількох категорій знань (зокрема для реалізації функції пошуку за категоріями). Для вирішення цієї задачі буде використано підхід із [6].

Бібліотека Lucene здатна проводити зважене зонне ранжування, що і буде використовуватись у алгоритмі пошуку майбутньої системи.

Система має надавати можливість пошуку статей за автором. Логіка реалізації даної функції відносно тривіальна. Користувач повинен мати змогу при пошуку задавати ім'я автора/авторів, і отримувати у результатах публікації вказаного автора/авторів. Також, при перегляді кожної статті, система повинна рекомендувати інші статті того ж автора/авторів, якщо такі є. Оскільки імена авторів – це окрема зона індексу системи, ця функція є частково реалізованою у бібліотеці Lucene.

У системі має бути реалізована функція пошуку документів за УДК та категорією/категоріями знань. Для цього база системи має містити повний список категорій (зокрема, для виведення підказок під час пошуку). Також кожен документ має бути віднесений до однієї або більше категорій під час класифікації. Як і пошук за автором, ця функція є частиною зонного пошуку в Lucene.

Під час перегляду результатів пошуку та статей, користувачеві має надаватись рекомендації щодо схожих матеріалів. Схожими вважатимуться матеріали, що мають спільного автора, спільні терміни в заголовку, спільні джерела або ж належать до однієї категорії знань. Також має бути реалізована окрема функція пошуку, що дозволить переглядати усі схожі статті до даної.

При перегляді певної статті користувач має мати можливість перейти до статей, що зазначені у списку посилань до даної (якщо вони наявні у базі системи). Також має бути реалізована можливість переходу до статей авторів, що зазначені у цьому списку.

Бувають випадки, коли різні автори працюють над однією темою, не здогадуючись про це. Система має допомогти таким користувачам знайти одне одного для подальшої співпраці. Для цього має бути реалізований модуль обміну повідомленнями між користувачами. Має бути можливість вказування поточних тем діяльності користувачів (і відповідно їх перегляду). Роботодавцям або інвесторам, що шукають спеціалістів для виконання того чи іншого завдання, система має запропонувати авторів, що працювали або працюють над відповідними темами. Має бути реалізований пошук користувачів, що працюють за певною (або схожою) темою.

Якщо система набуде потрібної популярності, необхідно буде реалізувати можливість самостійного додавання своїх робіт користувачами системи. Такий спосіб поповнення колекції має ряд переваг. Автор зможе чітко зазначити атрибути своєї роботи при додаванні (назву, дату, використану літературу, категорію знань тощо), водночас як система могла зробити це з помилками.

Зареєстрованим користувачам системи має надаватись можливість оцінювати роботи інших користувачів. До того ж, рейтинг статей автора має впливати на вагу його голосу при оцінюванні чужих статей. За допомогою таких оцінок, має визначатись якість матеріалу, яка має враховуватись при видачі результатів на пошукові запити (при ранжуванні документів).

Користувачі мають мати змогу визначати актуальність та цікавість обраної ними теми для інших користувачів. Зокрема, система надаватиме функціональність для зручного рецензування статей. Рецензії мають відображатися поряд із самими статтями у результатах пошуку, що допоможе користувачам оцінити якість, повноту та відповідність статті їх потребам ще до переходу на сторінку.

Розбір PDF-документів

Важливою частиною індексації та зваженого зонного ранжування наукових матеріалів у форматі PDF є їх розбиття на логічні частини (зони). Для дослідження обрано саме формат PDF, оскільки на даний момент він є найпопулярнішим у сфері наукових публікацій. Крім того, база документів, що буде використовуватись у прототипі системи, складається саме з матеріалів цього формату.

Основною проблемою розбору PDF документів є те, що цей формат не розрахований на програмну обробку. Він орієнтований на відображення, а не на збереження та передачу текстової інформації. Крім того, не існує єдиного стандарту для оформлення наукових публікацій, і навіть тих стандартів, що розроблені, рідко дотримуються.

Наш алгоритм має виділяти у PDF документі такі зони: заголовок, УДК (універсальний десятковий класифікатор), автори та використана література (список посилань). Однак, є і інші важливі частини такі як: анотація, видавництво, дата публікації, вступ тощо (ці зони в даній роботі не розглядатимуться).

Формат PDF (Portable Document Format) створено для представлення документів у незалежному від пристрою виведення та роздільної здатності вигляді. Доступ до тексту та легкість програмного розбору не входили до списку пріоритетів його розробки. Тому під час програмної обробки файлів цього формату виникають певні труднощі. Файли формату PDF складаються з директив для відображення контенту, що не групуються за розташуванням. Одне слово, або навіть буква, яку потрібно відобразити, може задаватися кількома директивами. Саме тому першочерговою задачею під час програмної обробки PDF документів є виділення тексту та групування його за абзацами.

Існує декілька готових бібліотек, що здатні виділяти текст з документів PDF (iText, PDFBox, PDF Clown), але вони здебільшого подають текст сторінками і без форматування. Тому прийнято рішення розширити функціональність однієї з бібліотек (iText) [7] таким чином, щоб текст виділявся абзацами з однаковим або схожим форматуванням (розмір і жирність шрифтів, вирівнювання тексту тощо).

Реалізовано такий алгоритм:

- за допомогою бібліотеки iText виділяються окремі прямокутники з текстом.
- визначаються та об'єднуються прямокутники, що лежать в одному рядку. Два прямокутники вважаються такими, що лежать в одному рядку, якщо їх вертикальний перетин складає не менше 80% висоти найнижчого з них. Крім цього має виконуватись одна з умов: а) горизонтальна відстань між ними не має бути більшою, ніж висота найнижчого з них; б) простір між цими прямокутниками рядком нижче повинен містити текст.
- визначаються та об'єднуються рядки, що належать до одного абзацу. Два рядки вважаються такими, що належать до одного абзацу, якщо виконуються такі умови:

1) жирність шрифтів в обох рядках однакова;

2) регістр символів цих рядків однаковий;

3) нехай:

- l – горизонтальна відстань між лівими краями рядків,
- r – горизонтальна відстань між правими краями рядків,
- c – горизонтальна відстань між центральними точками рядків,
- w – максимальна ширина пробілу шрифтів обох рядків (white space width),
- α, β, γ – деякі додатні константи.

Тоді виконується твердження:

$$(l < w \bullet \alpha) \vee (r < w \bullet \beta) \vee (c < w \bullet \gamma).$$

Експериментальним чином було підібрано: $\alpha = 1$, $\beta = 1.5$, $\gamma = 4.5$;

4) Вертикальна відстань між рядками не перевищує половину висоти лінії (line height) верхнього рядка.

Алгоритм поділу на логічні зони включає виділення заголовку, УДК, авторів та посилань.

Нехай X – множина абзаців першої сторінки документу; $font_x$ – нормалізований розмір шрифта абзацу x (тут і надалі $x \in X$); $lineheight_x$ – нормалізована висота лінії абзацу x; $centered_x$ – 1, якщо текст абзацу x відцентрований, 0 – якщо ні; $bold_x$ – 1, якщо шрифт абзацу x жирний, 0 – якщо ні; $uppercase_x$ – 1, якщо абзац x верхнього регістру, 0 – якщо ні;

Визначимо функцію:

$$h(x) = font_x \cdot fc + lineheight_x \cdot lc + centerd_x \cdot cc + bold_x \cdot bc + uppercase_x \cdot uc,$$

де fc, lc, cc, bc, uc – певні константи.

Заголовком статті вважатимемо текст того абзацу, для якого функція h прийматиме найбільше значення. Експериментальним шляхом підібрано такі значення констант:

$$fc = 1,\ lc = 1,\ cc = 0.75,\ uc = 1,\ bc = 0.25.$$

Пошук УДК зводиться до знаходження першого (за розташуванням зверху вниз, зліва направо) абзацу першої сторінки документа, текст якого починається з літер “УДК” та містить крім них лише цифри та знаки “-” і “.”.

Визначення авторів статті зводиться до знаходження абзацу, найближчого за розташуванням до абзацу заголовку, що містить текст формату “І.І. П” або “П І.І.”, де І – ініціали, П – прізвище. Для цього створено окремі регулярні вирази для кожного формату.

Пошук списку посилань полягає у знаходженні нумерованого списку, що починається з певних ключових слів (“Використана література”, “Список посилань” тощо) на одній із останніх сторінок документа. Алгоритм проходить по черзі сторінки документа, починаючи з останньої, доки не знайде абзац, що містить одне з ключових слів. Знайшовши його, алгоритм за допомогою регулярних виразів виділяє нумерований список, що розташований нижче цього абзацу. Якщо такого списку немає, продовжується пошук абзацу з ключовими словами.

Далі приведемо UML діаграму класів реалізації алгоритму (рис. 1, 2, 3).

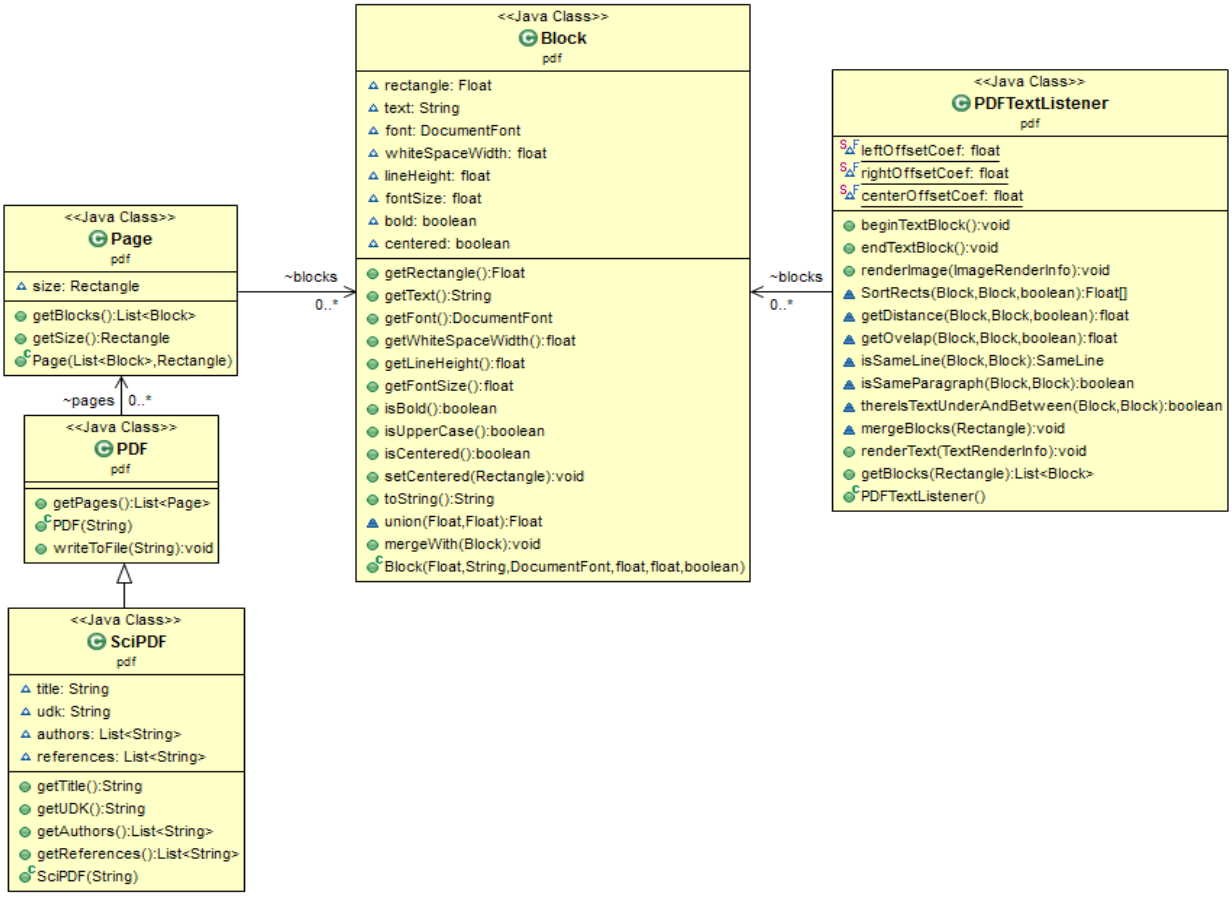

Рис.1. діаграма класів: package guess

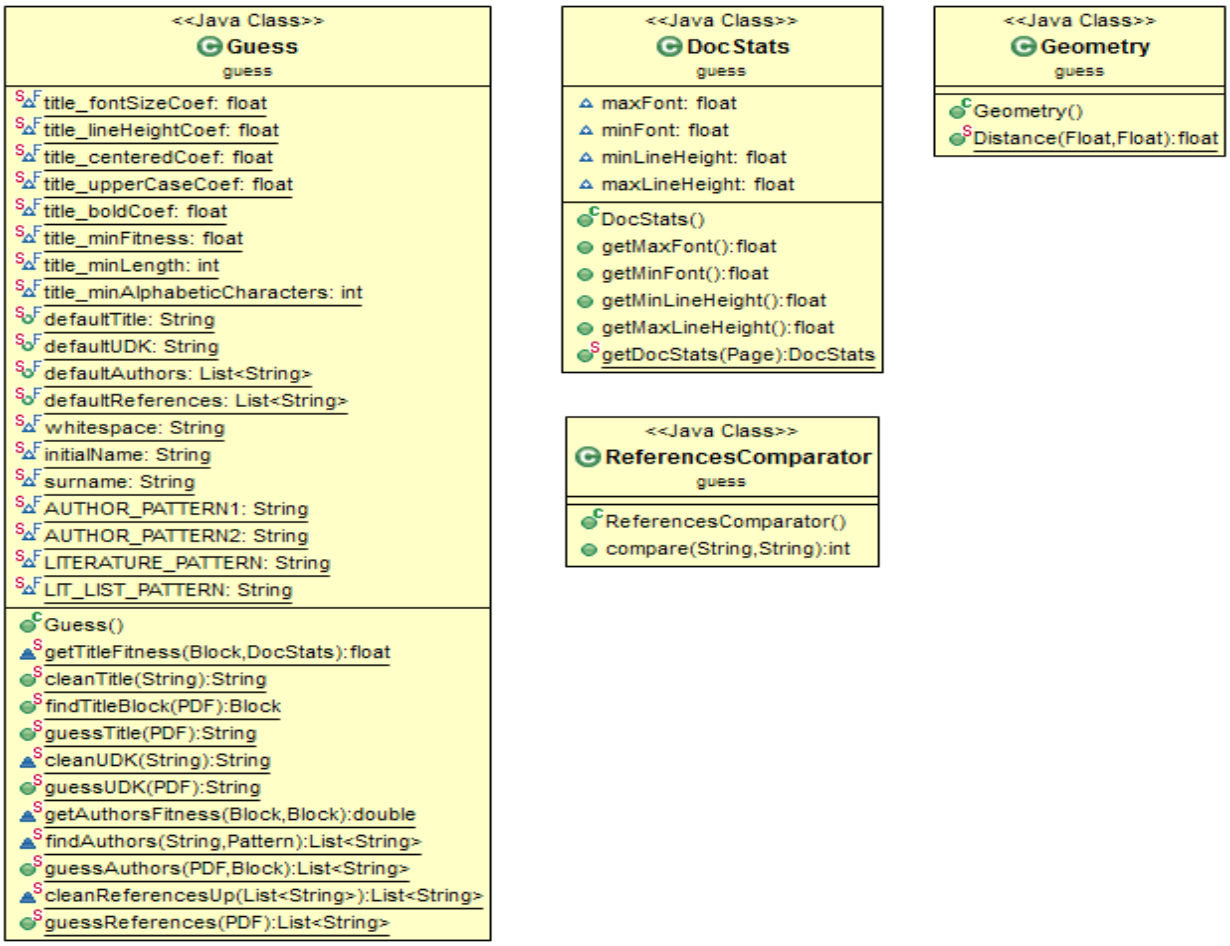

Рис. 2. Діаграма класів: package guess

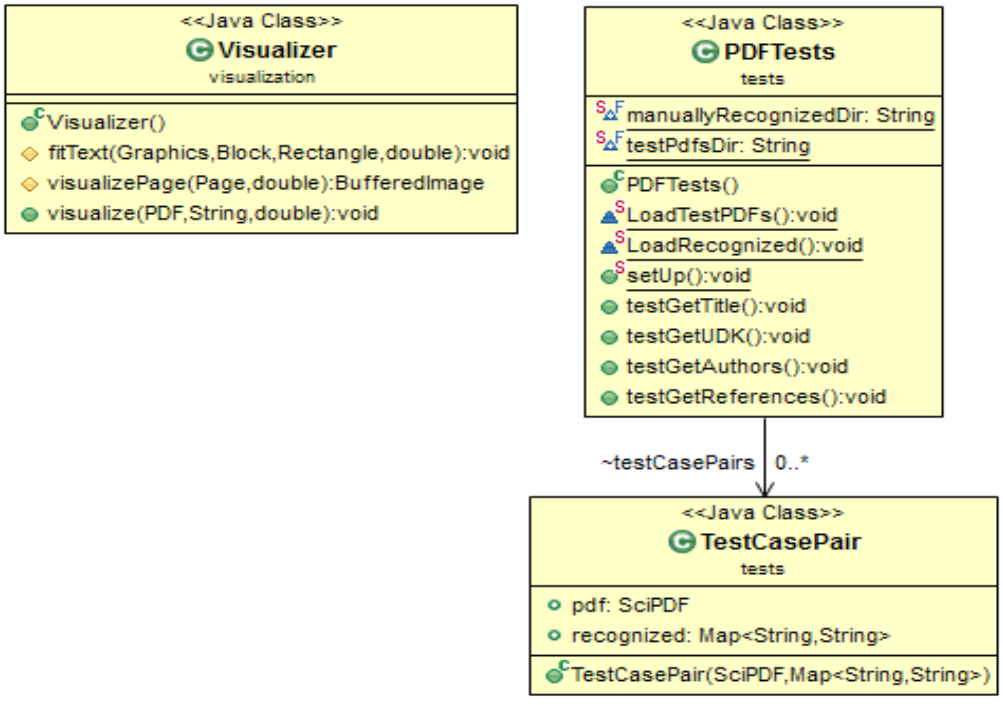

Рис. 3. Діаграма класів: package tests, visualization

Висновки

В даній роботі представлено архітектуру пошукової системи наукових матеріалів українською мовою з соціальною складовою. Запропоновано таку структуру та технології для реалізації, за допомогою яких можна розробити масштабовану, ефективну та надійну систему.

Запропоновано та реалізовано алгоритм виділення у PDF документах наукових статей чотирьох основних логічних зон. Описано сильні та слабкі сторони алгоритму. Проведений статистичний аналіз експериментальних випробувань реалізації алгоритму показали його практичну придатність, хоча втручання експертів для перевірки даних на перших етапах буде необхідно.

1. *Глибовець М.М., Жигмановський А.А., Заболотний Р.І., Захоженко П.О.* Веб сервіси оброблення документів, Національний університет "Києво-Могилянська академія". – К.: НаУКМА, – 2012. – 212с.
2. *Google* App Engine https://developers.google.com/appengine/
3. *Apache* Lucene http://lucene.apache.org/core/
4. *Національна* бібліотека України імені В.І. Вернадського http://nbuv.gov.ua/
5. *Глибовець А.М., Сітмамбетов Н.* Создание специализированной поисковой системы на базе облачных технологий // Пр. міжнар. конф. KDS 2012 “Knowledge-Dialog-Solution” 10-14 вересня 2012.
6. *Глибовець А.М., Шабінський А.С., Ольшевський Р.Я.* Побудова пошукового робота україномовних наукових матеріалів // Наукові праці МДУ ім. Петра Могили. Комп’ютерні технології. – Випуск 130. – Том 143, – 2010. – С. 81–87.
7. *ITEXT* Programmable PDF Software http://itextpdf.com/