https://orcid.org/0000-0003-4431-1989
https://orcid.org/0000-0002-5451-626X
https://orcid.org/0000-0001-5568-7629

УДК 621.37:637.142

В.О. БРАЦЬКИЙ[*], О.М. М'ЯКШИЛО[*], В.А. ЛИТВИНОВ[**]

ДІАГНОСТИЧНА СИСТЕМА АНАЛІЗУ LOG-ФАЙЛІВ ІЗ ВІДДАЛЕНИХ ВУЗЛІВ ОБРОБКИ ДАНИХ

[*]Національний університет харчових технологій, м. Київ, Україна
[**]Інститут проблем математичних машин і систем НАН України, м. Київ, Україна

Анотація. *Для супроводження та підтримки належного функціонування розподілених систем використовують реєстрацію подій у log-файлах із метою подальшого аналізу і виявлення обставин та причин відхилень від нормальної роботи. Інструментальні засоби для досягнення цієї мети мають в першу чергу забезпечувати: 1) швидкий пошук і обробку log-файлів у процесі аналізу і виявлення причин збою; 2) формування пропозицій щодо вирішення виявленої проблеми. У статті наведено результати аналізу існуючих засобів журналізації та обробки log-файлів і обґрунтовано доцільність розробки діагностичної системи, що пропонується. Описана загальна структура розробленої діагностичної системи, її функціональна модель. Розглянуті функції основних програмних модулів – парсера, що обробляє log-файл і збирає частини повідомлень у єдиний log-об'єкт, модуля діагностики, що обробляє інформацію про помилку в об'єкті log-файла, а також модуля накопичення і використання знань про відомі помилки функціонування вузла, способи їх усунення і про проблеми, не виявлені раніше. Обґрунтовано вибір нереляційної СКБД MongoDB для збереження даних із log-файлів і накопичення знань, розроблено уніфіковану структуру об'єктів log-файла в середовищі MongoDB. Управління діагностичною системою – сумісну роботу модулів та їх взаємодію з базою даних і базою знань у середовищі MongoDB – забезпечує головна програма LogHelper, що надає користувачу можливості конструювати запити по log-об'єктах, переглядати результати запиту, розглянути усі відомі і невідомі помилки, обробити невідому помилку і задати для неї статус, запропонувати існуючі рішення для користувача. Результати проведеного дослідження і апробації показали, що розроблена діагностична система є дієвим засобом підтримки функціонування віддалених вузлів обробки даних завдяки швидкому виявленню причин відмов і формування пропозицій щодо їх усунення.*

Ключові слова: *діагностична система, log-файл, віддалений вузол, MongoDB.*

Abstract. *To monitor and maintain the proper functioning of distributed systems, there are used log events to further analyze and identify the circumstances and causes of deviations from normal operation. Tools to achieve this goal should primarily provide: 1) rapid search and processing of log files in the process of analysis and identification of failure causes; 2) generation of offers for solving the identified problem. This article presents the results of the analysis of the existing tools for logging and log file processing and substantiates the feasibility of the development of the proposed diagnostic system. The general structure of the developed diagnostic system and its functional model are described. In the paper, there are considered the functions of the main software modules: a parser that processes a log file and collects parts of messages into a single log object; a diagnostic module that processes information about an error in a log file object; and a module for accumulating and using knowledge about the known node malfunctions, ways to fix them and problems that were not previously identified. The choice of MongoDB NoSQL for saving the log file data and knowledge base storage DBMS is substantiated. A unified structure for log file objects has been developed for the MongoDB environment. Diagnostics system control – interoperability of modules and their interaction with the database and knowledge base in the MongoDB environment – is provided by the main program LogHelper which allows the user to perform different queries on log objects, monitor their results, inspect known and not previously identified errors, work with unknown malfunctions, manually set their statuses and find a solution from an existing set of solutions. The results of the conducted research show that the developed diagnostic system is highly effective as a maintenance solu-*

tion for remote data processing nodes and it supports rapid identification of the malfunction and generation of proposed solutions.
Keywords: *diagnostic system, log file, remote node, MongoDB.*

DOI: 10.34121/1028-9763-2022-1-62-70

1. Вступ

Розподілені системи широко використовуються в різних сферах суспільної діяльності. Для супроводження та підтримки належного функціонування розподілених систем використовують реєстрацію подій у log-файлах із метою подальшого аналізу і виявлення обставин та причин відхилень від нормальної роботи. Інструментальні засоби для досягнення цієї мети мають в першу чергу забезпечувати: 1) швидкий пошук і обробку log-файлів у процесі аналізу і виявлення причин збою; 2) формування пропозицій щодо вирішення виявленої проблеми. Одним із основних шляхів належного виконання цих вимог є наявність у складі інструментальної системи засобів швидкого пошуку і обробки даних щодо log-файлів та їх об'єктів, а також засобів накопичення і використання знань про раніше виявлені проблеми.

Додатковими загальними «технологічними» вимогами є підтримка різних форматів log-файлів, невисока ресурсоємність, зручний інтерфейс користувача тощо.

У теперішній час існує декілька комерційних інструментальних продуктів лог-менеджменту для створення, організації, накопичення, пошуку і обробки log-файлів. Це, в першу чергу, LogParser [1], Logstash [2], WebLog Explorer [3].

Як видно з аналізу описів розглянутих продуктів, основною особливістю їх функціонала є широке призначення і, відповідно, деяка надлишковість функцій по відношенню саме до задачі обробки й аналізу log-файлів, для виявлення причин проблемної ситуації і шляхів її виправлення. З іншої сторони, в них відсутні можливості і відповідні засоби накопичення і повторного використання результатів попередніх аналізів. База даних, що обробляється, зокрема, для пришвидшення їх пошуку, передбачена лише у продукті LogParser.

Отже, *мета статті* полягає у викладенні результатів досліджень та розробки діагностичної системи на основі аналізу log-файлів, орієнтованої саме на конкретну задачу пришвидшення пошуку і обробки log-файлів та накопичення інформації щодо результатів попередніх аналізів.

2. Структура системи і функції основних компонентів

На рис. 1 наведено структуру діагностичної системи, яка складається з комплексної програми обробки і аналізу log-файлів (управляючий модуль LogHelper**)** у складі парсера, модуля діагностики, модуля набуття знань, бази даних і бази знань.

База даних слугує для збереження масиву log-файлів, що обробляються, і забезпечення їх швидкого пошуку за заданими критеріями. В базі знань накопичується інформація щодо відомих помилок і проблемних ситуацій. Користувачами системи виступають адміністратор, який здійснює завантаження log-файлів, та експерт-аналітик, який доповнює базу знань новими знаннями про помилки на вузлах обробки даних.

Процес функціонування системи ілюструє функціональна модель у нотації IDF0.

Функції парсера і модуля діагностики представлені діяльністю «Аналіз log-файлів», в яку входить зчитування log-файла, його перевірка та збирання у log-об'єкт, занесення у колекцію log-об'єктів.

Модулю набуття знань відповідає діяльність «Аналіз log-об'єктів», в якій проводиться ідентифікація тих log-об'єктів, даних про які немає в базі знань. По визначених критеріях log-об'єкт відноситься до певного класу подібних об'єктів і експерт-аналітик має визначити його статус, щоб ідентифікувати і помістити в базу знань.

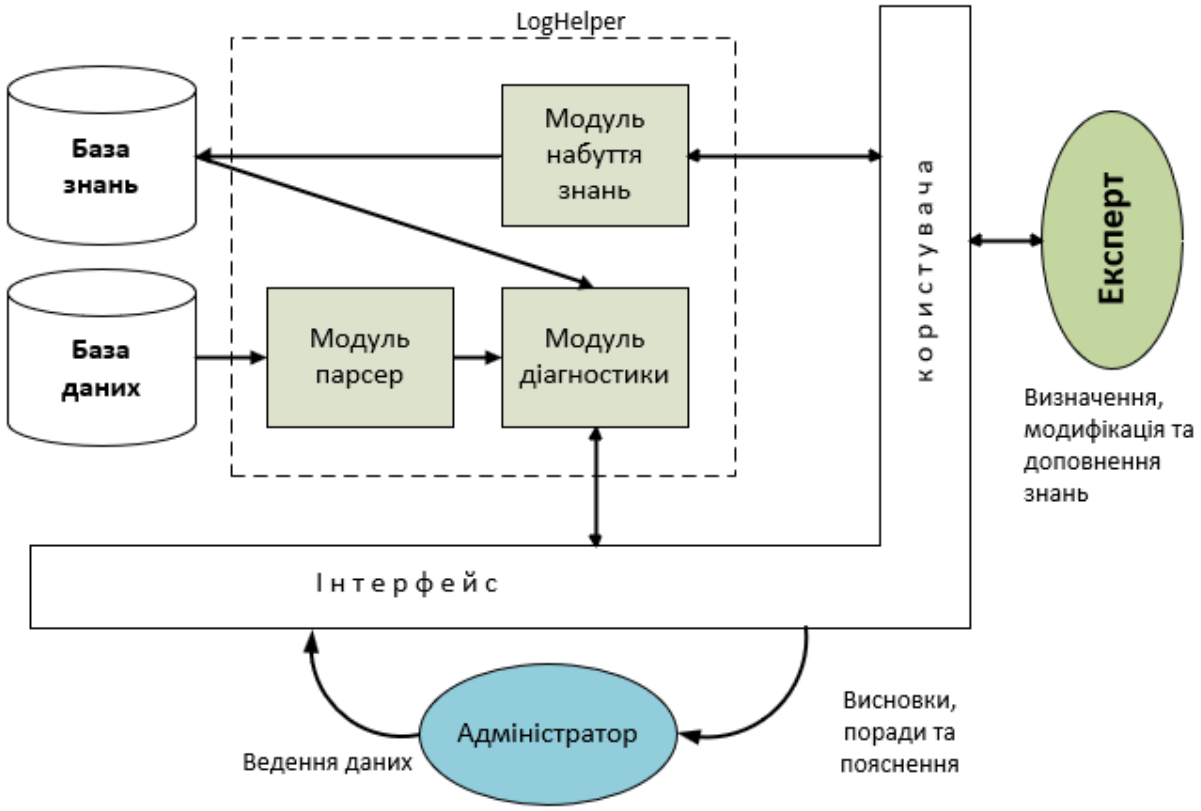

Рисунок 1 – Структурна схема діагностичної системи

Рисунок 2 – Функціональна діаграма роботи системи

3. Реалізація компонентів системи

3.1. Вибір СКБД

Для управління базою даних обрана нереляційна СКБД MongoDB. Вибір обумовлений такими перевагами MongoDB для виконання функцій системи у порівнянні з реляційними СУБД:

– зручність організації і значно краща швидкість обробки слабкоструктурованої інформації;

– краща, ніж у реляційних баз даних, масштабованість, що є важливим при обробці великих обсягів даних.

Так, експерименти показали [4, 5], що вставка 1 млн тестових записів на SQL Server виконується за 3040 секунд, а на MongoDB за 214 секунд, тобто в 14 разів скоріше. Типовий зважений набір вибірок, вставок, видалень при обробці тестових записів у середовищі MongoDB виконується за 1 секунду при однопотоковій роботі і за 3 секунди при багатопотоковій (50 потоків). Відповідні цифри для SQL Server склали 17 і 211 секунд.

MongoDB зберігає дані у форматі BSON (BinaryScriptObjectNotation), а подає користувачеві у форматі JSON – (JavaScriptObjectNotation). За рахунок своєї лаконічності в порівнянні з XML формат JSON більше підходить для серіалізації складних структур. Якщо говорити про веб-додатки, він доречний у задачах обміну даними як між браузером і сервером (AJAX), так і між самими серверами (програмні HTTP-сполучення).

3.2. Структура log-файла як об’єкта БД

Об’єкти MongoDB, у загальному вигляді, описуються такою дворівневою структурою:

$$\begin{cases} D = \{F0,\ F1 : e1, F2 : e2, ..., Fn : en, d1, ..., d2, ... dl\}, \\ Dr = \{\ fr0,\ fr1 : k1,\ fr2 : k2, ...,\ frm : km\}, \end{cases}$$

де $F0$ та $f0$ – ідентифікатори батьківського та дочірніх документів відповідно;

$F1 ... Fn$ та $fr1 ... frm$ – назви атрибутів батьківського та дочірніх документів;

$e1 ... en$ та $k1 ... km$ – атомарні значення атрибутів батьківського та дочірнього документів;

$d1 ... dr$ – посилання на інші документи або вкладені документи.

У реалізації представлення вихідних об’єктів log-файла змінні структури (1) мають такі значення:

$F0$ – «Дата і інформація про запит»;

$F1$ – «Тіло запиту»;

$F2$ – «Дата і інформація про відповідь»;

$F3$ – «Тіло відповіді»;

$e0 - e3$ – атомарні значення атрибутів $F0 - F3$;

$F4 - Fn$ – додаткові атрибути повідомлення;

$e4 - en$ – атомарні значення додаткових атрибутів повідомлення.

Для збереження сформованих парсером log-об’єктів змінні структури (1) приймають значення:

$F0$ – «Ідентифікатор»;

$F1$ – «Дата запиту»;

$F2$ – «Тіло запиту»;

$F3$ – «Тіло відповіді»;

$F3$ – «Дата відповіді»;

$F5$ – «Ідентифікатор зв’язку між запитом і відповіддю»;

$e0-e5$ – атомарні значення атрибутів $F0-F5$;

$fr0-frm$ – назви атрибутів вкладеного документа;

$k1...km$ – атомарні значення атрибутів вкладеного документа.

У вкладений документ вміщуються відомості про log-файл, з якого отримали інформацію.

3.3. Програма LogHelper

Програма LogHelper [6, 7] забезпечує управління діагностичною системою – сумісну роботу модулів та їх взаємодію з базою даних і базою знань у середовищі MongoDB.

На рис. 3 наведено об'єктну модель даних, які використовуються і створюються у процесі функціонування діагностичної системи.

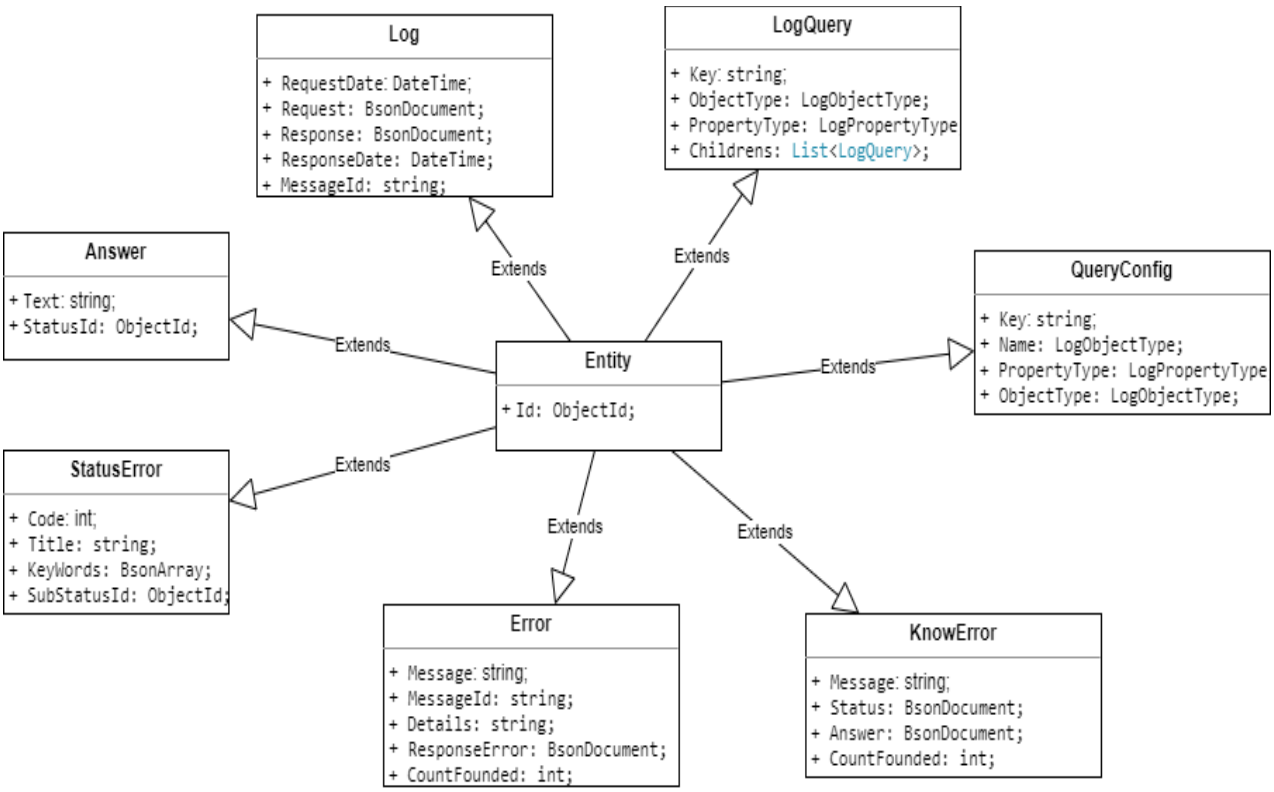

Рисунок 3 – Об'єктна модель даних системи

Об'єктна модель складається з колекцій об'єктів: Log, Error, StatusError, KnownError, Answer, LogQuery, QueryConfig.

У коллекції Error зберігаються усі невідомі для бази знань помилки, які в подальшому будуть опрацьовані адміністратором, експертом-аналітиком або програмістом.

У коллекції StatusError зберігаються статуси для оброблених помилок, за допомогою яких відбувається класифікація помилок.

У колекції KnownError зберігаються всі відомі для бази знань помилки, які були опрацьовані адміністратором, аналітиком або програмістом.

У колекції Answer – зберігаються варіанти рішень для опрацьованих помилок і помилок, які знаходяться у колекції KnownError;

У колекції LogQuery зберігаються варіанти фільтрів, за властивостями, для пошуку потрібних log-об'єктів.

У колекції QueryConfig зберігаються вже обрані користувачами варіанти фільтрації.

Після запуску програми LogHelper система пропонує обрати log-файли для аналізу або користувач може обрати з верхнього рядка меню розділ «Робота над log-об'єктами», де є можливість:

– конструювати запити по log-об'єктах;
– переглядати результати запиту;
– розглянути усі відомі і невідомі помилки;
– обробити невідому помилку і задати для неї статус;
– запропонувати існуючі рішення для користувача.

3.4. Парсер

Log-файл – це файл текстового формату, в який заносяться всі дані про будь-які дії відвідувачів і користувачів серверів розподіленої системи. Файл має послідовну структуру, в якій логічно пов'язані записи є розпорошеними. Наприклад, дані для конкретного користувача на конкретну дату про запит до сервера вузла і відповідь від нього можуть зберігатися в різних частинах log-файла, між ними можуть бути інші звернення та відповіді. Для проведення аналізу повідомлень слід всю пов'язану інформацію зібрати в одному документі. Парсер обробляє log-файл і збирає частини повідомлення (як запит, так і відповідь) в єдиний log-об'єкт, придатний для зберігання в базі даних, і подальшого аналізу.

Після завершення перебору записів log-файла усі повні документи зі структурою, наведеною в розд. 3.2, заносяться в базу даних MongoDb.

Для завантаження log-файла в базу даних використовується методика GridFS. Це специфікація MongoDB для зберігання та виведення великих файлів, розмір яких перевищує обмеження на розмір документа BSON, що дорівнює 16 МБ. Технологія GridFS розбиває великі файли на менші фрагменти. Фрагменти зберігаються в одній колекції (fs.chunks), а метадані про ці фрагменти в іншій колекції (fs.files).

3.5. Модулі діагностики і набуття знань

У процесі роботи модуля діагностики відбувається пошук (по повному співпадінню або по ключових словах) інформації про помилку в об'єкті log-файла, обробленому парсером. Якщо в базі знань така інформація присутня, то користувачу відображається повідомлення з такими даними (рис. 4):

– статус помилки, в якому описано її код і назва;
– відповідь або рішення даної проблеми;
– кількість знайдених аналогічних помилок.

Якщо інформація відсутня, то об'єкти, оброблені парсером, порівнюються з уже існуючими повідомленнями у базі даних із помилками, схожими із знайденою помилкою, і зберігаються для подальшої обробки. В повідомленні про незнайдену помилку відзначається і кількість помилок такого типу.

Експерт-аналітик має можливість обробити помилку, задавши для неї статус і варіант або варіанти рішень (рис. 5). Система привертає увагу аналітика-експерта до невідомої помилки, вказуючи на частоту її появи. В активному вікні наведено перелік близьких за змістом, до невідомої помилки, подій та вказано їх статус.

При визначенні статусу для невідомої помилки є можливість створити новий статус або обрати існуючий. За допомогою статусу здійснюються класифікація помилок і фільтрація даних для подальшого аналізу правильності роботи вузла розподіленої системи.

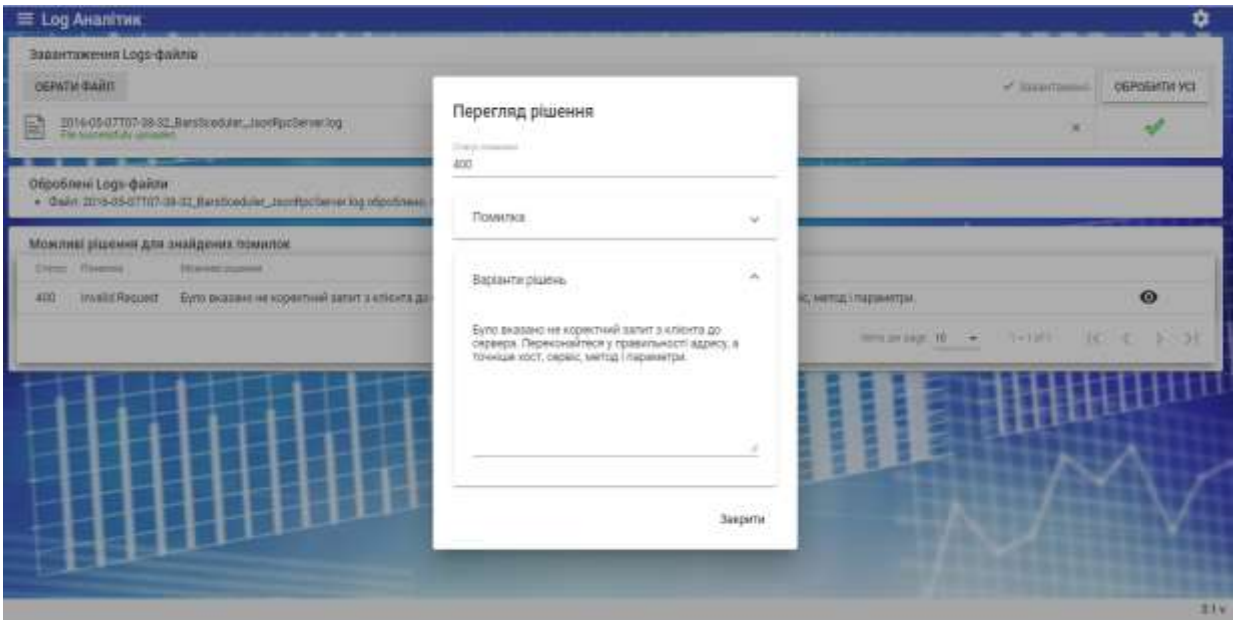

Рисунок 4 – Інтерфейс модуля діагностики

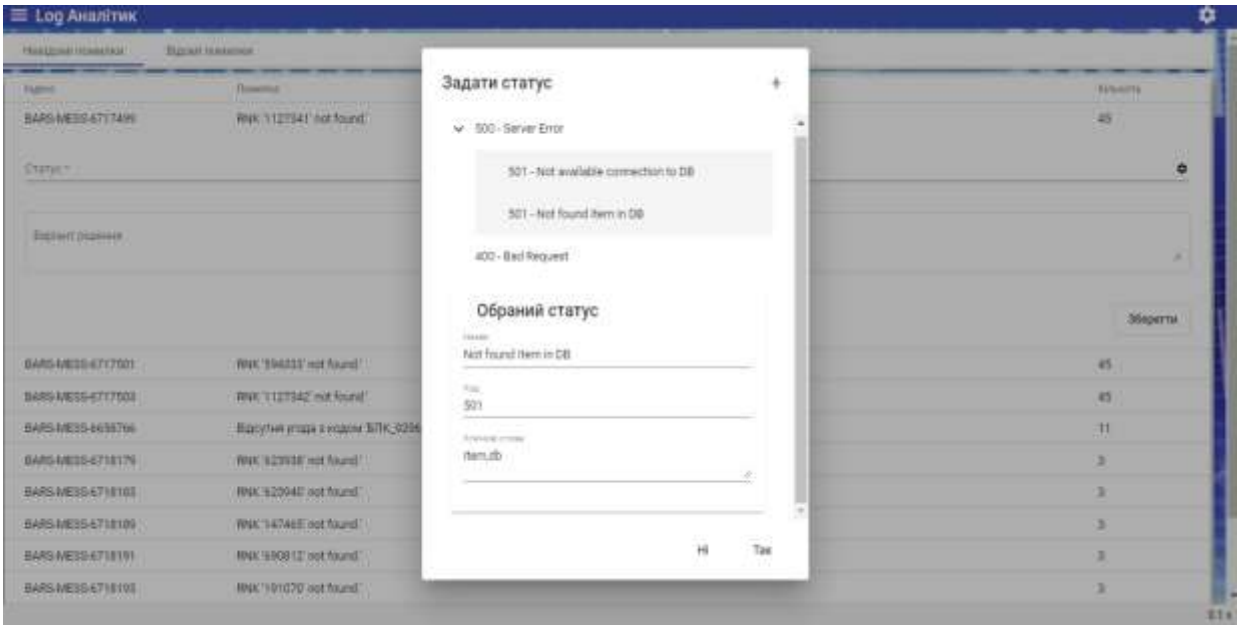

Рисунок 5 – Інтерфейс модуля набуття знань

При визначенні статусу для невідомої помилки є можливість створити новий статус або обрати існуючий. За допомогою статусу здійснюються класифікація помилок і фільтрація даних для подальшого аналізу правильності роботи вузла розподіленої системи.

4. Тестування роботи системи

Робота діагностичної системи апробована і протестована в корпоративній банківській системі. При роботі працівника банку з інформаційною системою усі його дії, а точніше запити до сервера вузла і відповіді сервера, записуються у log-файли, які є першоджерелом інформації про можливі помилки і позаштатні ситуації. Дані файли створюються кожного дня нові і в назві містять дату створення. Адміністратор, працівник банку, має доступ до log-файлів і для детального розбору позаштатної ситуації проводить аналіз обраних файлів (наприклад, за датою) за допомогою описаної діагностичної системи.

Результати роботи системи на прикладі фрагмента log-файла, в якому зберігаються запити до сервера і відповіді, ілюструє рис. 6.

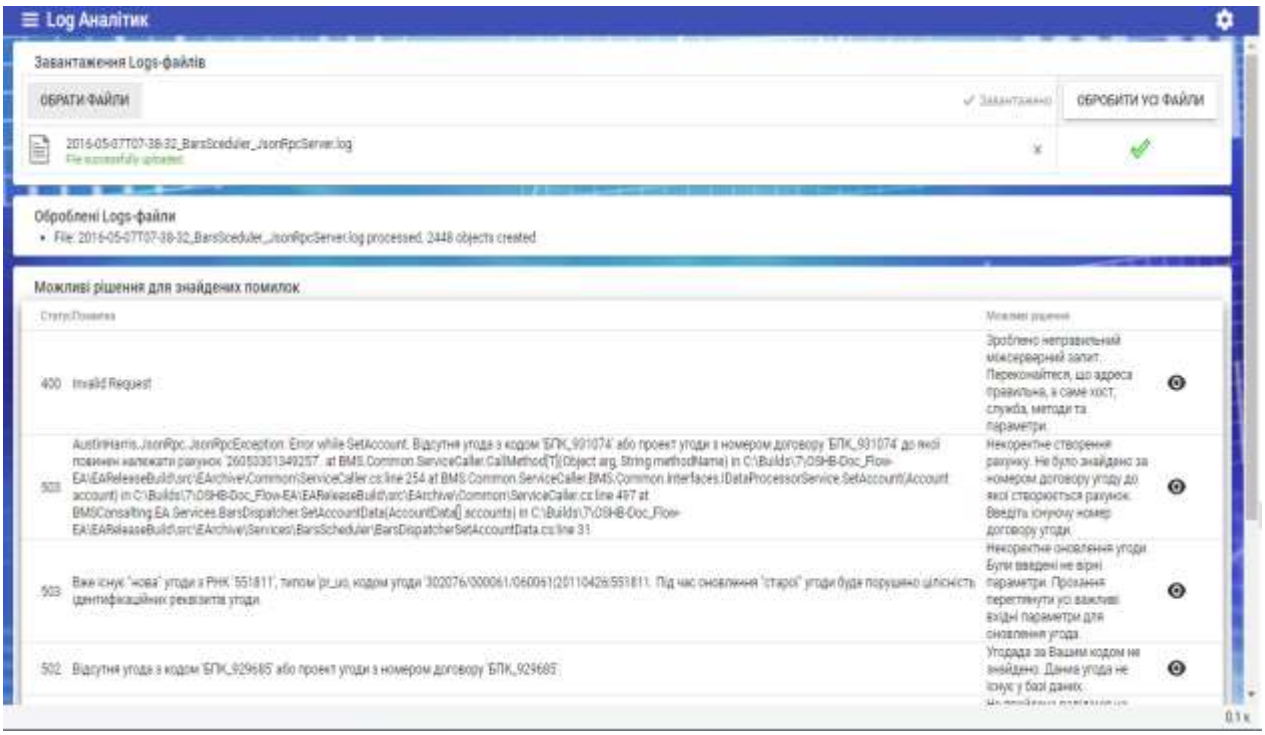

Рисунок 6 – Результати роботи модуля діагностики

Після проведеного аналізу система інформує користувача про відомі помилки для системи і варіанти рішень для усунення даних помилок. Зокрема, система виявила такі помилки:

– некоректно сформований запит до сервера вузла;
– помилка бізнес-логіки при створенні нового рахунку;
– помилка бізнес-логіки при оновленні угоди;
– не знайдено угоди за параметрами у базі даних;
– не пройдена валідація на стороні сервера.

У наведеному прикладі відбувалися завантаження і аналіз log-файлів у MongoDb із віддаленого вузла обробки даних. Загальний розмір 43-ох файлів займає 504 Мб. Вставка тривала 10,05 секунди. Розмір колекції з об’єктами займає 265 Мб. Як можна побачити, MongoDb стискає колекцію майже вдвічі, швидкість при цьому є прийнятною.

Системою діагностики за секунду обробляється 1400 рядків і створюється 344 повних об’єкти, які готові для збереження у базі даних. Після занесення log-файла до БД у MongoDb можна легко побудувати JSON-запити і проводити аналіз даних, наприклад, за такими параметрами, як дата, місце розташування, статус виконання тощо.

5. Висновки

Результати проведеного дослідження і апробації показали, що розроблена діагностична система є дієвим засобом аналізу функціонування клієнтських додатків на віддалених вузлах обробки даних. Використання LogHelper дає змогу побудувати чіткі і точні запити для аналізу log-файлів та знаходження рішень для усунення помилок.

Діагностичну систему було впроваджено в підрозділі «Розробки і підтримки систем компанії ТОВ «ЄЕА»» та використано у банківській інформаційній системі «Централізо-

ваний електронний архів». В акті впровадження зазначено, що використання діагностичної системи значно прискорило виявлення і вирішення проблем у корпоративній системі банку. Витрати часу компанії для аналізу помилок зменшилися з годин на хвилини і привели до зменшення негативних проявів помилок у банківській системі.

Поряд з цим запропоновані підходи і рішення, які визначають властивості системи, створюють, на думку авторів, підстави для обґрунтування її включення у програмне забезпечення різних корпоративних систем, зокрема, наприклад, спеціалізованих систем управління контентом типу [8].

СПИСОК ДЖЕРЕЛ

1. Парсер журналов LogParser 2.2. URL: https://www.microsoft.com/en-us/download/details.aspx?id=24659.
2. Logstash: Collect, Parse, Transform Logs | Elastic. URL: https://www.elastic.co/logstash/.
3. Web Log Explorer – Web Log Analyzer. URL: https://www.exacttrend.com/WebLogExplorer/.
4. Брацький В.О., М'якшило О.М. Дослідження особливостей застосування реляційних і нереляційних баз даних на прикладі SQL SERVER та MONGODB. *Наукові праці НУХТ*. 2016. Т. 22, № 5. С. 18–24.
5. Брацький В.О., М'якшило О.М. Дослідження та розробка методу обробки log-файлів у розподіленій інформаційній системі з використанням нереляційної бази даних MongoDB. *Наукові праці НУХТ*. 2018. Т. 24, № 1. С. 17–25.
6. Брацький В.О. Оброблення й аналіз log-файлів за допомогою програмного продукту LogHelper. *Наукові праці Таврійського національного університету ім. В.І. Вернадського*. 2018. Т. 29 (68), № 5, Ч. 1. С. 34–41.
7. А.с. Міністерство розвитку економіки, торгівлі та сільського господарства України. Комп'ютерна програма «Аналіз log-файлів» / В.О. Брацький, О.М. М'якшило. № 95824 від 06.02.2020.
8. Грибков С.В., Литвинов В.А., Олійник Г.В. Інструментальна модель веб-орієнтованої програмної реалізації підсистеми підтримки прийняття рішень у складі програмного комплексу ситуаційного центру. *Математичні машини і системи*. 2020. № 1. С. 73–81.

Стаття надійшла до редакції 28.01.2022