

ТЕХНИЧЕСКИЕ НАУКИ



УДК 004.45

Использование асинхронной парадигмы для OSINT приложений

Д.П. Ковалев, А.А. Лёза, Н.В. Гапон

Донской государственный технический университет, г. Ростов-на-Дону, Российская Федерация

Аннотация

Статья посвящена применению асинхронной архитектуры при программной реализации модуля для инструментария OSINT (Open Source Intelligence). Представлена схема основных функций, приведены результаты тестирования, а также рассмотрен пример реализации асинхронного подхода. Выполнено сравнение разработанного решения с существующими аналогами. Результаты работы представляют интерес для специалистов в области кибербезопасности, журналистики, социологии и государственного управления.

Ключевые слова: OSINT, открытая разведка, большие данные, машинное обучение, верификация информации, социальные сети, кибербезопасность, цифровой след

Для цитирования. Ковалев Д.П., Лёза А.А., Гапон Н.В. Использование асинхронной парадигмы для OSINT приложений. *Молодой исследователь Дона*. 2026;11(5):31–36.

Using Asynchronous Pattern for OSINT Applications

Danil P. Kovalev, Aleksei A. Leza, Nikolai V. Gapon

Don State Technical University, Rostov-on-Don, Russian Federation

Abstract

The article studies implementation of the asynchronous architecture in programming an OSINT (Open Source Intelligence) framework module. The article provides a diagram of the main functions, presents testing results, and studies an example of asynchronous approach implementation. The developed solution has been compared to the existing analogues. The research results could be of interest to the experts in the fields of cybersecurity, journalism, sociology, and public governance.

Keywords: OSINT, open source intelligence, big data, machine learning, information verification, social networks, cybersecurity, digital footprint

For Citation. Kovalev DP, Leza AA, Gapon NV. Using Asynchronous Pattern for OSINT Applications. *Young Researcher of Don*. 2026;11(5):31–36.

Введение. Информационное пространство Интернета стало крупнейшим в истории источником открытых данных. Социальные сети, форумы, государственные реестры, новостные порталы и блоги содержат значительный объем сведений, представляющих операционный, аналитический и стратегический интерес. Методология OSINT, под которой понимается целенаправленный сбор и анализ данных из открытых источников для решения конкретных задач, приобрела критически важное значение в деятельности журналистов, правоохранительных органов, аналитиков и специалистов по безопасности.

В рамках данной статьи разработан алгоритм, направленный на устранение отдельных недостатков при работе с инструментами OSINT, в частности синхронной обработки данных. Повышение производительности обеспечивается за счет применения асинхронной парадигмы программирования. Функциональная схема примера использования алгоритма представлена на рис. 1.

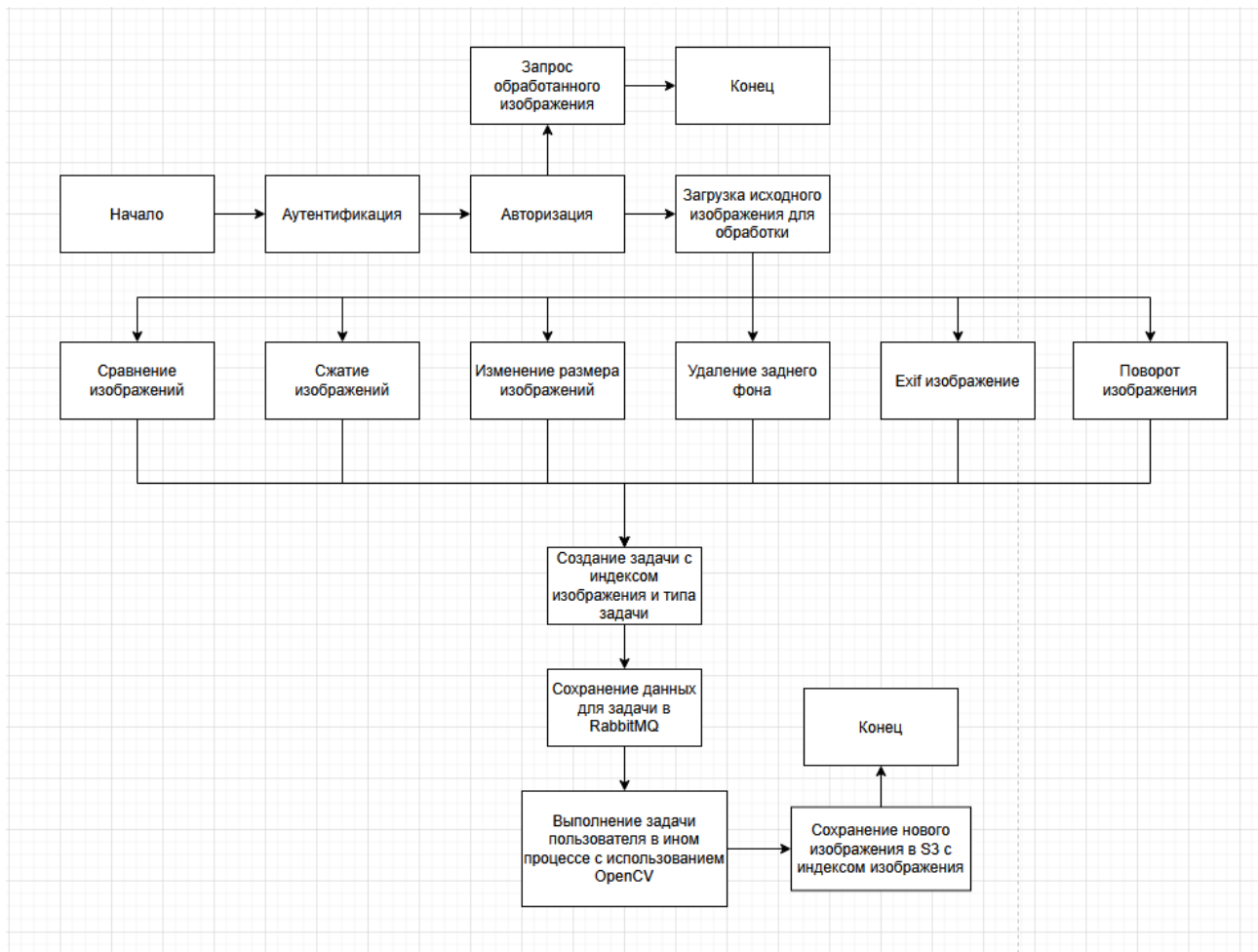


Рис. 1. Функциональная схема

На основе данного алгоритма разработан программный продукт для OSINT, демонстрирующий различия между традиционной синхронной и асинхронной обработкой данных. Ниже рассмотрен один из алгоритмов, который может применяться совместно с асинхронной архитектурой.

Сжатие изображения — архитектурная реализация. После получения пользовательского запроса на сжатие изображения сервис выполняет проверку прав доступа и загружает исходный файл, предоставленный пользователем. Далее формируется детерминированный TaskID, представляющий собой значение имени `compress_image` и идентификатор изображения, предназначенного для обработки. Задача передается в планировщик TaskIQ [1], при этом HTTP-обработчик сразу возвращает TaskID клиенту и не блокируется выполнением вычислений. Такой подход обеспечивает асинхронный режим работы и позволяет масштабировать процесс сравнения независимо от API-процесса.

На уровне инфраструктуры TaskIQ [1] использует RabbitMQ [2] в качестве брокера очередей и Redis [3] как хранилище результатов и данных о ходе выполнения, поэтому состояние задачи может отслеживаться по TaskID. В worker-процессе TaskIQ [1] получает сообщение из RabbitMQ [2], переводит задачу в состояние STARTED с фиксацией прогресса в Redis [3], повторно загружает изображения из хранилища и выполняет сравнение посредством доменного сервиса. Результат сравнения сохраняется в реляционной базе данных, после чего состояние задачи изменяется на SUCCESS. При возникновении ошибок, например, в случае отсутствия файла, задача переводится в состояние FAILURE и отклоняется, что позволяет корректно зафиксировать проблему и не блокировать очередь. Таким образом, с точки зрения программной реализации реализуется асинхронная обработка сжатия изображения.

Сжатие изображения – математическая модель. Сам процесс сжатия можно объяснить сквозь анализ математической составляющей для сжатия изображения. В случае разработки сервиса использовался механизм квантования, который встроен в библиотеку OpenCV.

Квантование — это отображение значений сигнала из непрерывного (или высокоточного дискретного) пространства в конечное множество уровней представления, то есть отображение $Q: R \rightarrow \{y_k\}$, где каждому интервалу решений R_k сопоставляется уровень y_k [4]. В скалярном равномерном случае квантование описывается следующим образом:

Квантование — это отображение значений сигнала из непрерывного (или высокоточного дискретного) пространства в конечное множество уровней представления, то есть отображение $Q:R \rightarrow \{y_k\}$, где каждому интервалу решений R_k сопоставляется уровень y_k [4]. В скалярном равномерном случае квантование описывается следующим образом:

$$Q(x) = \Delta \cdot \text{round}\left(\frac{x}{\Delta}\right), \quad (1)$$

где Δ — шаг квантования. В JPEG квантование применяется к коэффициентам DCT и является основным источником потери информации [5, 6].

Кодирование, выполняемое в OpenCV с помощью `cv2.imencode("jpg", ..., IMWRITE_JPEG_QUALITY)`, соответствует стандартной цепочке JPEG [5, 6]. В начале выполняется разбиение на блоки 8×8 и применяется DCT для каждого блока, как представлено ниже по формуле:

$$(u, v) = \frac{1}{4} a(u) a(v) \sum_{x=0}^7 \sum_{y=0}^7 f(x, y) \cos\left(\frac{(2x+1)u\pi}{16}\right) \cos\left(\frac{(2y+1)v\pi}{16}\right), \quad (2)$$

где $u, v \in \{0, \dots, 7\}$, $a(0) = \frac{1}{\sqrt{2}}$, $a(k) = 1$ при $k > 0$ [0,5]

Следующий этап — квантование процессов DCT (основная стадия потерь). В данном случае используется формула, которая представлена ниже:

$$\psi(u, v) = \text{round}\left(\frac{F(u, v)}{Q(u, v)}\right), \quad (2)$$

где $Q(u, v)$ — элемент матрицы квантования. Чем больше $Q(u, v)$, тем грубее квантование и выше степень сжатия [5, 6].

Можно сделать наблюдение, что для этапа сжатия выбран JPEG-кодировщик, реализованный в OpenCV (`cv2.imencode`), поскольку он обеспечивает устойчивый компромисс между степенью сжатия, скоростью и качеством при типичных фотографических изображениях. JPEG является международным стандартом (ISO/IEC 10918-1), имеет повсеместную поддержку и детерминированное поведение, что упрощает воспроизводимость экспериментов и интеграцию в прикладные системы.

Тестирование программы — ручное тестирование. Перед началом использования сервиса «PixErase» администратор выполняет развертывание в контейнеризированной среде. Для воспроизводимого запуска инфраструктуры используется Docker-compose [7].

Для проверки корректности работы прикладного функционала используется тестовое изображение, представленное на рис. 2. Данное изображение служит входным объектом для последующих операций обработки и позволяет воспроизводимо проверять результативность сценариев сервиса.

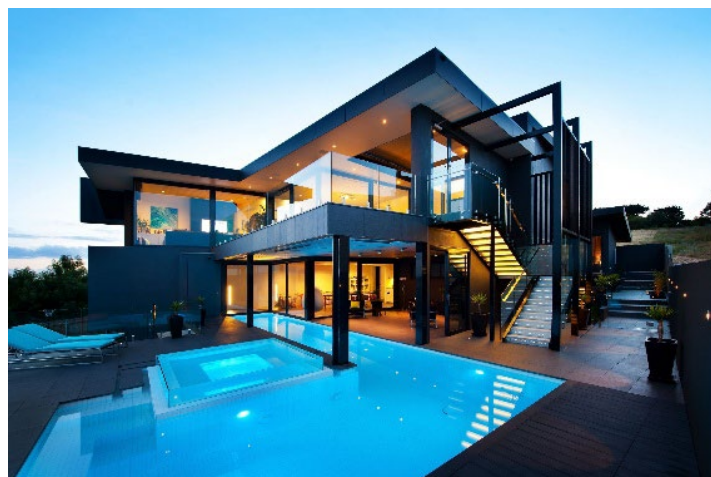


Рис. 2. Тестовое изображение для обработки

В результате выполнения запроса на загрузку изображения в backend формируется ответ сервера с присвоенным идентификатором изображения. Пример запроса и ответа приведен на рис. 3. Полученный `image_id` далее используется в последующих операциях обработки.

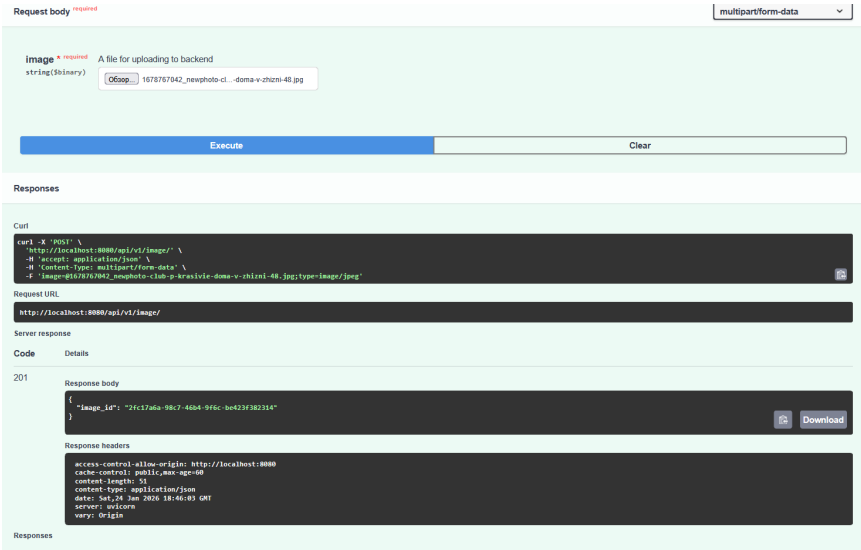


Рис. 3. Загрузка изображения на backend

Для применения алгоритма сжатия используется запрос PATCH /v1/image/id/{image_id}/compress с параметром качества quality в теле запроса, пример показан на рис. 4.

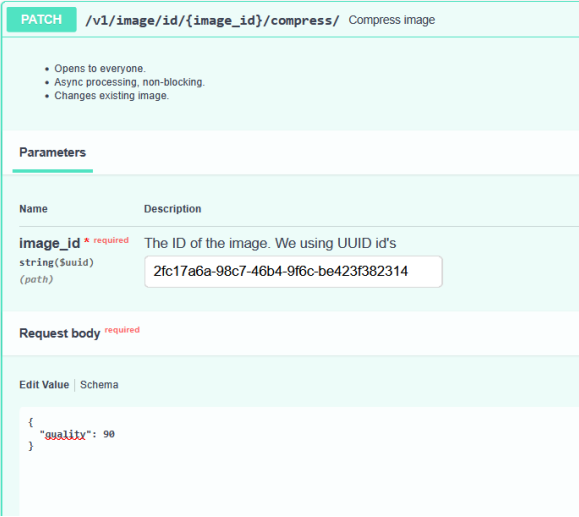


Рис. 4. Сжатие изображение до 90 % качества

После завершения обработки выполняется скачивание исходного изображения, что позволяет провести сравнительный анализ размеров файлов.

Сопоставление размеров до и после сжатия представлено на рис. 5 и демонстрирует уменьшение объема данных при сохранении приемлемого визуального качества, что подтверждает работоспособность выбранного режима компрессии.

1678767042_newphoto-club-p-krasvie-d...	24.01.2026 21:57	Файл "JPG"	591 КБ
1678767042_newphoto-club-p-krasvie-d...	24.01.2026 21:51	Файл "JPG"	791 КБ

Рис. 5. Сравнение размеров изображений

Нагрузочное тестирование. Для оценки производительности «PixErase» было проведено сравнение синхронной и асинхронной обработки изображений на базе алгоритмов OpenCV. Тестирование выполнялось на стенде: AMD Ryzen 7 7800X3D (5.0 ГГц), NVIDIA RTX 4080, 32 ГБ DDR5-6000. В качестве тестовых данных использовался датасет Pistachio Image Dataset [8]. Изображения выбирались случайным образом, для асинхронного режима задействовано 4 worker-процесса. Сравнение проводилось по трём метрикам: время выполнения операций, утилизация CPU/GPU и общая пропускная способность конвейера.

Результаты подтвердили, что асинхронная архитектура обеспечивает существенный прирост производительности при пакетной обработке за счёт неблокирующего выполнения задач и эффективного распределения нагрузки между процессами. Результаты сравнения представлены в таблице 1.

Время выполнения операций обработки изображений (секунды)

Операция	Количество изображений	Синхронная обработка	Асинхронная обработка	Ускорение, %
Сжатие (JPEG, quality = 90)	1	0,045	0,048	–6,7
Сжатие (JPEG, quality = 90)	10	0,450	0,125	72,2
Сжатие (JPEG, quality = 90)	50	2,250	0,485	78,4
Сжатие (JPEG, quality = 90)	100	4,500	0,920	79,6

Сравнение с существующими решениями. В ходе разработки «PixErase» был проведён сравнительный анализ с аналогом Osintimage.ai [9]. Ключевое различие подходов: если Osintimage.ai представляет собой монолитное решение для AI-геолокации, то «PixErase» — это модульная сервисная платформа для комплексной обработки и подготовки визуальных данных. Основное архитектурное преимущество «PixErase» — полностью асинхронный конвейер обработки на базе RabbitMQ. Это обеспечивает:

- масштабируемость — возможность обработки больших пакетов файлов в фоновом режиме без блокировки интерфейса;
- эффективность — кэширование результатов в S3-совместимом хранилище (MinIO) исключает дублирование вычислений;
- надёжность — полная контейнеризация (Docker Compose) и документированный API упрощают интеграцию в промышленные OSINT-конвейеры.

Таким образом, «PixErase» ориентирован на задачи, требующие высокой пропускной способности и гибкости, тогда как аналог [9] оптимален для легковесного развёртывания под узкоспециализированные задачи.

Заключение. Проведён анализ проблем обработки открытых данных в OSINT. Разработан модуль «PixErase», использующий асинхронную архитектуру для ускорения анализа изображений. Ключевыми преимуществами решения являются модульность, поддержка кэширования и высокая производительность по сравнению с аналогами. Инструмент готов к внедрению в системы мониторинга и расследований. Дальнейшие исследования будут направлены на интеграцию алгоритмов верификации контента на базе машинного обучения и работу с распределёнными источниками в реальном времени.

Список литературы

1. *TaskIQ Python. Асинхронный диспетчер распределённых задач.* URL: <https://taskiq-python.github.io> (дата обращения: 10.12.2025).
2. *RabbitMQ 4.3 Documentation.* URL: <https://www.rabbitmq.com/docs> (дата обращения: 10.12.2025).
3. *Redis документация к базе данных.* URL: <https://redis.io> (дата обращения: 10.12.2025).
4. Gray RM, Neuhoff DL. Quantization. *IEEE Transactions on Information Theory.* 1998;44(6):2325–2383. <https://doi.org/10.1109/18.720541>
5. ITU–T Recommendation T.81 (09/92). ISO/IEC 10918–1:1992. *Information Technology — Digital Compression and Coding of Continuous–Tone Still Images: Requirements and Guidelines (JPEG).* Geneva, 1992. 186 p. URL: <https://www.itu.int/rec/T-REC-T.81> (дата обращения: 10.12.2025).
6. Wallace GK. The JPEG Still Picture Compression Standard. *Communications of the ACM.* 1991;34(4):30–44. <https://doi.org/10.1145/103085.103089>
7. *Docker Compose.* URL: <https://docs.docker.com/compose> (дата обращения: 10.12.2025).
8. Murat Koklu. *Pistachio Image Dataset.* URL: <https://www.kaggle.com/datasets/muratkokludataset/pistachio-image-dataset> (дата обращения: 10.12.2025).
9. Ronantakizawa. *Osintimage.ai (репозиторий программного кода).* URL: <https://github.com/ronantakizawa/osintimage.ai> (дата обращения: 10.12.2025).

Об авторах:

Данил Петрович Ковалев, студент кафедры «Кибербезопасность информационных систем» Донского государственного технического университета (344003, Российская Федерация, г. Ростов-на-Дону, пл. Гагарина, 1), dan.kovalev2013@gmail.com

Алексей Александрович Лёза, студент кафедры «Кибербезопасность информационных систем» Донского государственного технического университета (344003, Российская Федерация, г. Ростов-на-Дону, пл. Гагарина, 1), ll_leza@mail.ru

Николай Валерьевич Гапон, кандидат технических наук, старший преподаватель кафедры «Кибербезопасность информационных систем» Донского государственного технического университета (344003, Российская Федерация, г. Ростов-на-Дону, пл. Гагарина, 1), ngapon@donstu.ru

Конфликт интересов: авторы заявляют об отсутствии конфликта интересов.

Все авторы прочитали и одобрили окончательный вариант рукописи.

About the Authors:

Danil P. Kovalev, Student of the Cybersecurity of Information Systems Department, Don State Technical University (1, Gagarin Sq., Rostov-on-Don, 344003, Russian Federation), dan.kovalev2013@gmail.com

Aleksei A. Leza, Student of the Cybersecurity of Information Systems Department, Don State Technical University (1, Gagarin Sq., Rostov-on-Don, 344003, Russian Federation), ll_leza@mail.ru

Nikolai V. Gapon, Cand.Sci. (Engineering), Senior Lecturer of the Cybersecurity of Information Systems Department, Don State Technical University (1, Gagarin Sq., Rostov-on-Don, 344003, Russian Federation), ngapon@donstu.ru

Conflict of Interest Statement: the authors declare no conflict of interest.

The authors have read and approved the final manuscript.