РАЗРАБОТКА ПРОГРАММНОГО КОМПЛЕКСА ЦЕНТРАЛИЗОВАННОГО ЛОГИРОВАНИЯ СЛОЖНЫХ РАСПРЕДЕЛЕННЫХ СИСТЕМ

Винничек Екатерина Васильевна
К.п.н., доцент, Пензенский государственный университет архитектуры и строительства, г. Пенза
katjushika@yandex.ru

Аннотация. Рассмотрены проблемы логирования функционирования компонентов сложных распределенных вычислительных систем. Выполнена задача проектирования централизованного логирования информационной системы. Разработан программный комплекс, позволяющий вести централизованное логирование и мониторинг функционирования ПО.

Ключевые слова: вычислительные системы, логирование, мониторинг вычислительных систем, уровни логирования, централизованное логирование, логирование в сложных распределенных системах, система оповещения.

BUILDING A CENTRALIZED LOGGING SOFTWARE FOR COMPLEX DISTRIBUTED SYSTEMS

E. Vinnichek

Summary. Considered the problems of logging the functioning of complex distributed system components. Completed the designing of centralized logging for information system. Developed centralized logging and monitoring software.

Keywords: computing systems, logging, computing system monitoring, logging levels, centralized logging, logging for complex distributed systems, alerting system.

Введение

Современные требования к уровню автоматизации технологических процессов очень высоки [1], что вызывает определенные сложности в их реализации. Построение и запуск нетривиальных процессов влечет за собой проектирование и разработку сложных распределенных вычислительных систем. Важнейшим вопросом, который необходимо решить уже на этапе проектирования — это разработка системы логирования функционирования компонентов вычислительной системы. К сожалению, для большого числа информационных систем до сих пор вопрос логирования рассматривается только как задача отладки отдельных приложений и сервисов на этапе разработки.

Грамотно реализованный процесс логирования позволяет получать не только информацию об ошибках, он позволяет получать параметризированные данные логики работы приложения — те данные, которые невозможно получить другим способом. Систематизированное хранение и анализ таких данных, позволяет построить систему логирования вычислительной системы. Система логирования может функционировать отдельно, либо являться подсистемой или источником данных для полноценной системы мониторинга.

Современный рынок IT-продуктов предоставляет целый класс систем мониторинга и мониторинговых сервисов [2], агрегирующий метрики работы сетей, серверов, баз данных и приложений. Эти системы на основании полученных данных способны формировать комплексное представление инфраструктуры, обновляемое в реальном времени. Как правило, это сложные, дорогостоящие системы, требующие настройки и сопровождения квалифицированных специалистов. Целесообразность использования таких продуктов мониторинга определяется сложностью и финансовыми возможностями реализуемых проектов. В то время, как задача реализации системы логирования — более простая задача, которая может быть реализована самостоятельно. При этом грамотно построенная система логирования хорошо масштабируется и может легко взаимодействовать с мониторинговыми сервисами.

Задачами исследования являются:

- изучение и анализ требований к процессу логирования событий;
- проектирование системы централизованного логирования;
- программная реализация системы логирования, которая способна эксплуатироваться отдельно, либо предоставлять данные для существующих

систем мониторинга или мониторинговых сервисов.

1. Анализ требований к процессу логирования

Процесс логирования (журналирования) — автоматическая запись событий, происходящих в рамках определенного процесса. Логирование позволяет оценивать состояние процесса и регистрировать ошибки. Главное преимущество использования логирования заключается в предоставлении возможности контролировать процесс выполнения бизнес-логики как отдельного процесса, так и системы в целом. Непосредственными потребителями данных, предоставляемых системой логирования, могут быть:

- разработчики и служба сопровождения ПО;
- системные администраторы и администраторы безопасности;
- аналитики информационной системы.

Основными требованиями к процессу логирования событий в распределенных системах являются:

1) наличие централизованного хранилища данных;
2) отображение текущего состояния системы и её отдельных процессов; доступ к истории состояний;
3) регистрация событий на любых компьютерах и серверах вычислительной системы от любых программ и сервисов;
4) наличие функциональной системы поиска и фильтрации зарегистрированных в системе событий;
5) возможность централизованного администрирования и мониторинга системы;
6) наличие системы оповещения (email, SMS, мессенджеры) о возникновении определенных событий;
7) оперативное изменение уровня и детализации логирования любого приложения вычислительной системы;

8) процесс фиксирования событий не должен сказываться на работоспособности приложений;

2. Проектирование и программная реализация системы централизованного логирования

Процесс логирования можно логически поделить на несколько этапов:

- регистрация событий и передача информации о них в хранилище данных;
- анализ и структуризация полученных данных и запись их в хранилище;
- мониторинг данных хранилища и управление процессом логирования.

Сопоставив каждому логически выделенному этапу свой уровень, получим трехуровневую систему:

1) Клиентский уровень;
2) Уровень БД;
3) Уровень администрирования.

2.1. Клиентский уровень системы логирования

Любая информационная система на нижнем уровне представляет собой набор приложений и сервисов. Процесс работы каждого приложения может характеризоваться набором логических действий и состояниями, которые могут быть детерминированы и зафиксированы. Фиксация события возможна двумя способами:

- оперативная передача информации в хранилище;
- запись информации на локальном ПК для дальнейшей передачи в хранилище.

Оба варианта имеют свои преимущества и недостатки. Первый вариант не требует дополнительных средств и ресурсов для локального хранения, но влечет за собой потерю данных в случае отсутствия связи с хранилищем. Минусы второго варианта — это небольшая задержка в передаче данных в хранилище, необходимость в наличии ресурсов для хранения временных данных, разработка программного решения асинхронной передачи данных в хранилище. Основное преимущество второго варианта — это надежность передачи данных в хранилище.

При проектировании системы логирования будем ориентироваться на второй способ (промежуточное локальное хранение информации с её дальнейшей передачей в хранилище). Реализовать такой функционал логичнее всего в виде сервиса/службы операционной системы. В дальнейшем для такой службы будем использовать наименование Watcher-сервис.

При программной реализации в состав Watcher-сервиса целесообразно включить функционал простого сервера (например, сокет-сервер или http-сервер) и реализовать интерфейс обмена. Это позволит управлять работой сервиса как локально, так и удаленно с помощью утилиты/диспетчера управления (Рис. 1).

В проектируемой модели все приложения и службы информационной системы должны фиксировать свою работу в журналы (структурированные лог файлы). Запись событий целесообразно выполнять в отдельных потоках в асинхронном режиме, чтобы это не сказывалось на производительности самих приложений.

Watcher-сервис будет анализировать лог-файлы приложений и в случае обнаружения “новых” данных передавать их в хранилище. Временное отсутствие связи

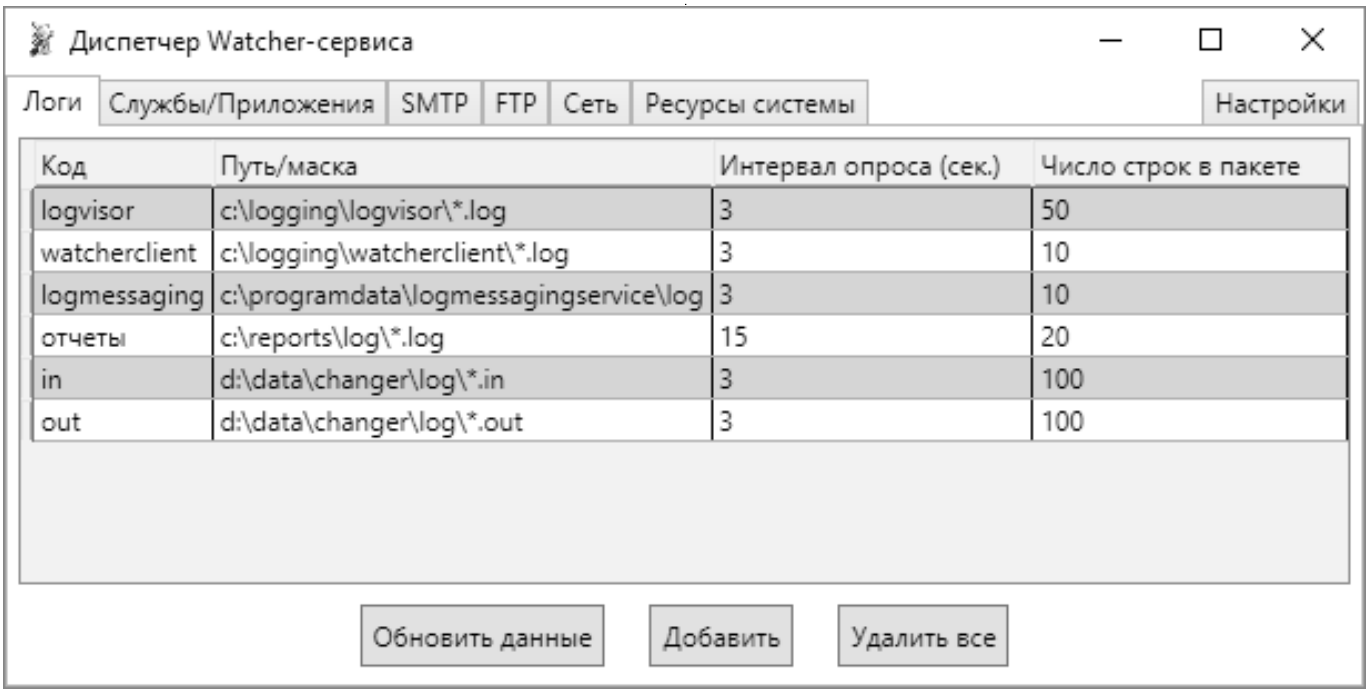

Рис. 1. Утилита управления Watcher-сервисом

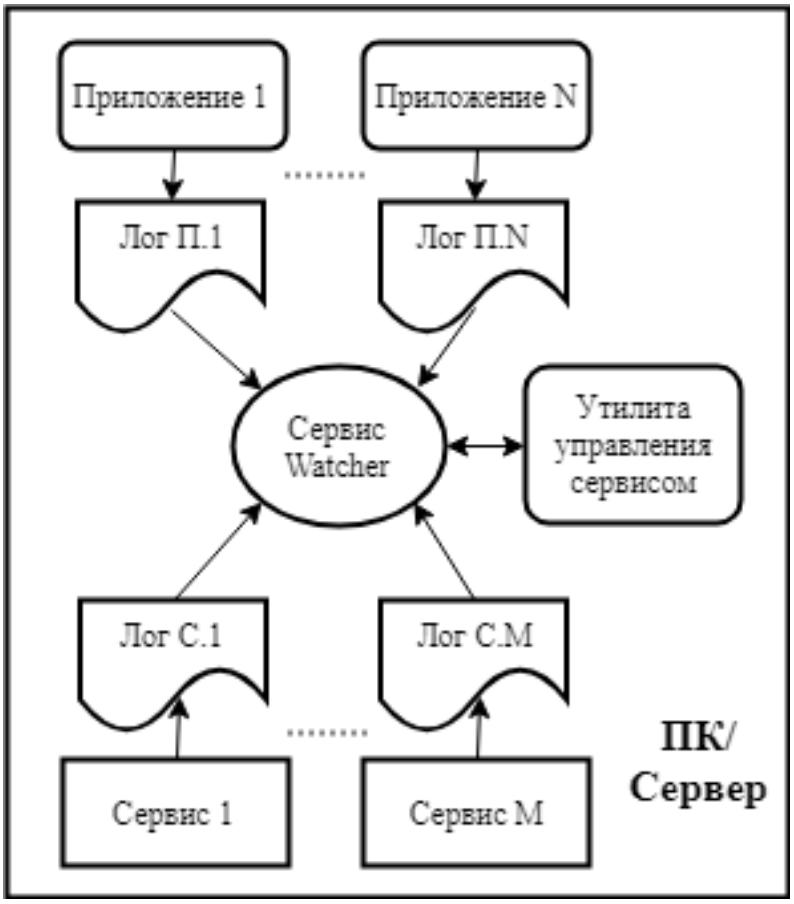

Рис. 2. Модель клиентского уровня.

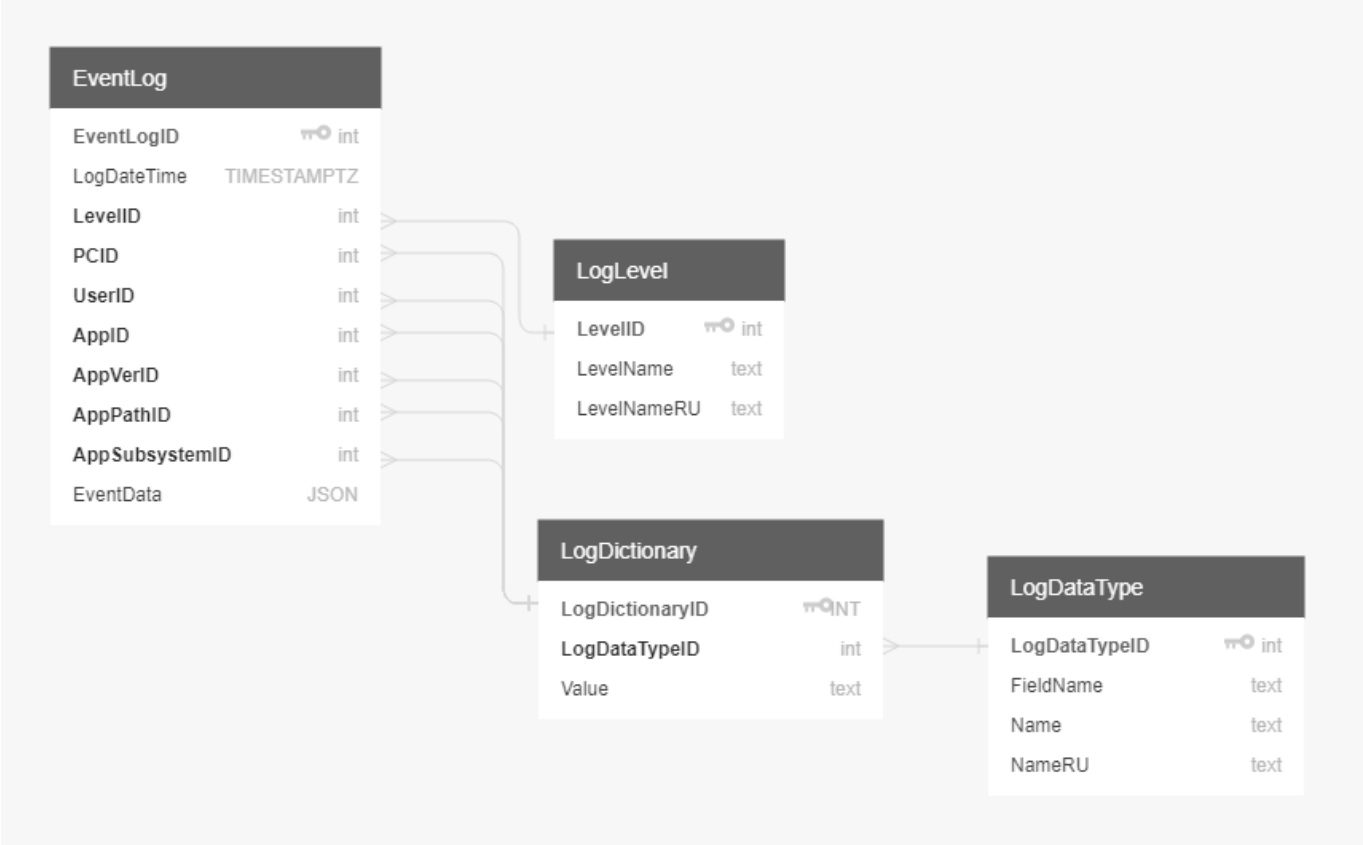

Рис. 3. Модель данных, описывающая события системы

с хранилищем в данном случае лишь отложит передачу информации в БД, но не приведет к её потере.

Схема модели клиентского уровня приведена на рисунке 2.

При регистрации событий необходимо их типизировать, чтобы в дальнейшем рационально обрабатывать данные. Для этого принято выделять несколько уровней событий. В программной реализации проектируемой системы будем использовать библиотеку.NET Serilog [3], которая использует следующие регистрируемые уровни событий:

Verbose — используется для отслеживания информации и отладки мелочей на этапе разработки/проектирования.

Debug — уровень используется для регистрации сообщений отладки или профилирования. В production системе сообщения этого уровня, как правило, включают при первоначальном запуске системы или для временной отладки узких мест.

Information — к этому уровню относят события, представляющие интерес или имеющие отношения к внешним наблюдателям. По умолчанию этот уровень является минимальным при ведении журнала логирования.

Warning — указывает на возможные проблемы или ухудшение работы/функциональности. Сообщения этого уровня требуют анализа, поскольку они могут соответствовать как уровню Information, так и ошибке. Возможно, это регистрация нового типа ситуации, ещё не известного системе или не предусмотренного для дальнейшей чёткой типизации уровня.

Error — указывает на сбои в работе приложения или ошибки взаимодействия с внешними системами. Ошибки этого уровня и выше требуют обязательной фиксации в системах логирования и дальнейшего решения.

Fatal — критические ошибки, приводящие к полному сбою приложения или неработоспособности системы в целом. Ошибки данного уровня требуют немедленной реакции.

Для унификации необходимо разработать единый формат записи в журнал лога. В качестве универсального варианта можно предложить использовать построчную запись в текстовый файл, где одна строковая запись

соответствует одному событию. Строковая запись может включать в себя следующую информацию:

- Дата/время события;
- Важность события (уровень);
- Имя ПК;
- Имя пользователя;
- Имя приложения/службы/библиотеки;
- Версия ПО;
- Путь к приложению;
- Подсистема (для многопотоковых приложений);
- Описание события.

2.2 Уровень БД

На уровне БД организуется централизованное хранилище данных. Задача рационального выбора СУБД для реализации данного уровня ложится на архитекторов системы исходя из запросов и возможностей компании. В нашей реализации системы логирования выбор был сделал в пользу PostgreSQL.

Все события можно хранить в одной таблице. Для хранения параметров событий логично создать справочники/словари для каждого типа параметров. Можно объединить их в единый справочник (LogDictionary), как показано на диаграмме (Рис. 3).

Такая реализация потребует больше ресурсов при добавлении в БД информации, но позволит рационально использовать дисковое пространство сервера СУБД и будет иметь высокую скорость поиска/выбора данных о событиях системы.

Программная реализация других структур данных для полноценной работы системы (например, для рассылки уведомлений системы оповещения) в данной работе рассматриваться не будет.

2.3 Уровень администрирования

На уровне администрирования решаются три задачи:

- мониторинг данных системы;
- удаленное управление процессом логирования;
- оповещение (рассылка уведомлений о событиях системы).

2.3.1 Мониторинг системы

Объем информации, характеризующий состояние информационной системы и отдельных её составляющих огромен. Для обработки этой информации нужен инструмент — средство мониторинга системы. Основная цель мониторинга — предоставление оперативного информационного обеспечения, которое способствует надежному функционированию системы.

Основными требованиями к системе мониторинга являются:

- определение важных рабочих параметров/показателей для отслеживания;
- оперативное выявление отклонений;
- периодическая и своевременная отчетность;
- осуществление фильтрации событий;
- создание правил мониторинга;
- настройка средств оповещения;
- построение графической отчетности.

При реализации системы мониторинга необходимо определиться с объектами для мониторинга. Для проектируемой системы централизованного логирования такими объектами будут являться:

- наличие связи с централизованным хранилищем;
- состояние сервисов логирования (Watcher-сервисов) на контролируемых объектах системы;
- логическое состояние отслеживаемых приложений и служб;
- системные и аппаратные характеристики объектов;
- доступность ресурсов.

Для мониторинга системных и аппаратных характеристик объектов и доступность на объектах ресурсов можно использовать средства сторонних разработчиков. Но наиболее оптимальным решением будет расширение функционала сервиса логирования. Целесообразно использовать Watcher-сервис не только для передачи логов в хранилище данных, но и для передачи в БД аппаратного состояния (наличие свободной памяти, потребляемые ресурсы процессоров и т.п.) и состояния доступности сетевых ресурсов, FTP и пр.

На рисунке 4 приведен пример программной реализации такого мониторинга:

Для определения параметров отслеживания в системе мониторинга целесообразно использовать фильтры (Рис. 5). Это позволит контролировать события с важными параметрами.

Первоначальная настройка (создание фильтров) может выполняться системными администраторами высокой квалификации. Дальнейшее сопровождение системы и мониторинг можно возложить на менее профессиональных специалистов.

Важно понимать, что любая система мониторинга не способна заменить собой квалификацию пользователей и администраторов. Система мониторинга является лишь инструментом, предоставляющим необходимые сведения для принятия верного решения.

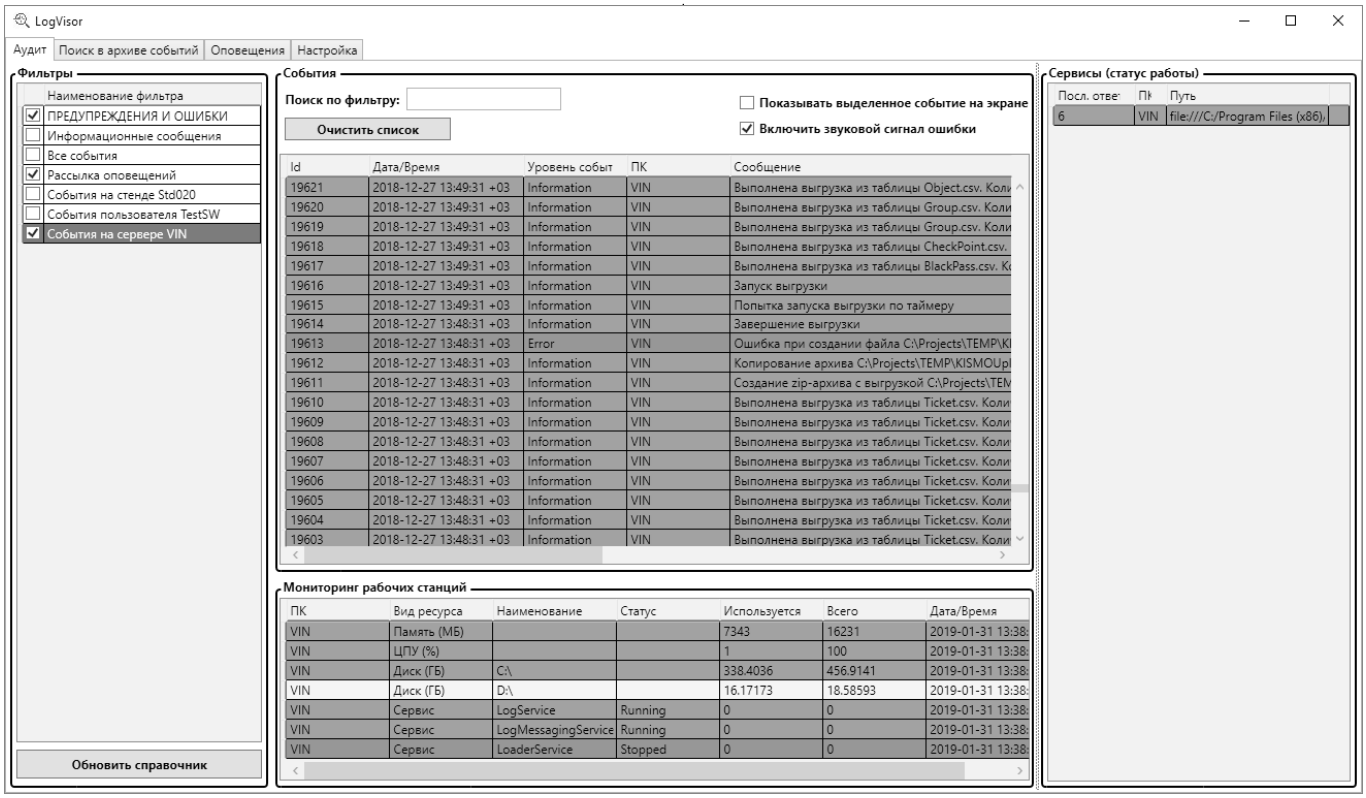

Рис. 4. Приложение мониторинга информационной системы

2.3.2 Удаленное управление процессом логирования

Программная реализация сервиса логирования в виде сервера (сокет-сервер, http-сервер) предоставляет инструмент централизованного управления процессом сбора и передачи информации в хранилище данных. В этом случае удаленно с рабочего места администратора можно менять все настройки одновременно или, например, указать, что с определенного рабочего места не передавать в хранилище события “с уровнем ниже Warning”.

2.3.3 Система уведомлений

Система визуального мониторинга является основным инструментом, предоставляющим информацию о состоянии распределенной информационной системы. К недостаткам системы мониторинга стоит отнести необходимость в наличии постоянного доступа. Отсутствие оператора системы мониторинга на рабочем месте или отсутствие удаленного доступа к панели мониторинга лишает систему надежности функционирования. Эти факторы являются одной из причин в необходимости дополнительной реализации системы уведомлений.

В качестве системы уведомлений можно использовать push-технологию [4] либо самостоятельно реализовать простую систему оповещения клиентов по существующим каналам (SMS, email, мессенджеры). Программная реализация — сервис операционной системы с доступом к централизованному хранилищу данных и каналам связи.

Принцип работы системы уведомлений аналогичен системе мониторинга. Фильтры событий в данном случае соответствуют подпискам. Получатели уведомлений являются подписчиками. В случае возникновения событий в определенных подписках всем их подписчикам будут рассылаться соответствующие уведомления.

Использование системы уведомления позволяет контролировать работу информационных систем удаленно. На практике часто разработчики сложных систем после передачи комплекса программ в эксплуатацию используют системы оповещения, подписываясь на критически важные события.

2.4 Схема системы логирования

Схема спроектированной системы логирования отображена на рис. 6.

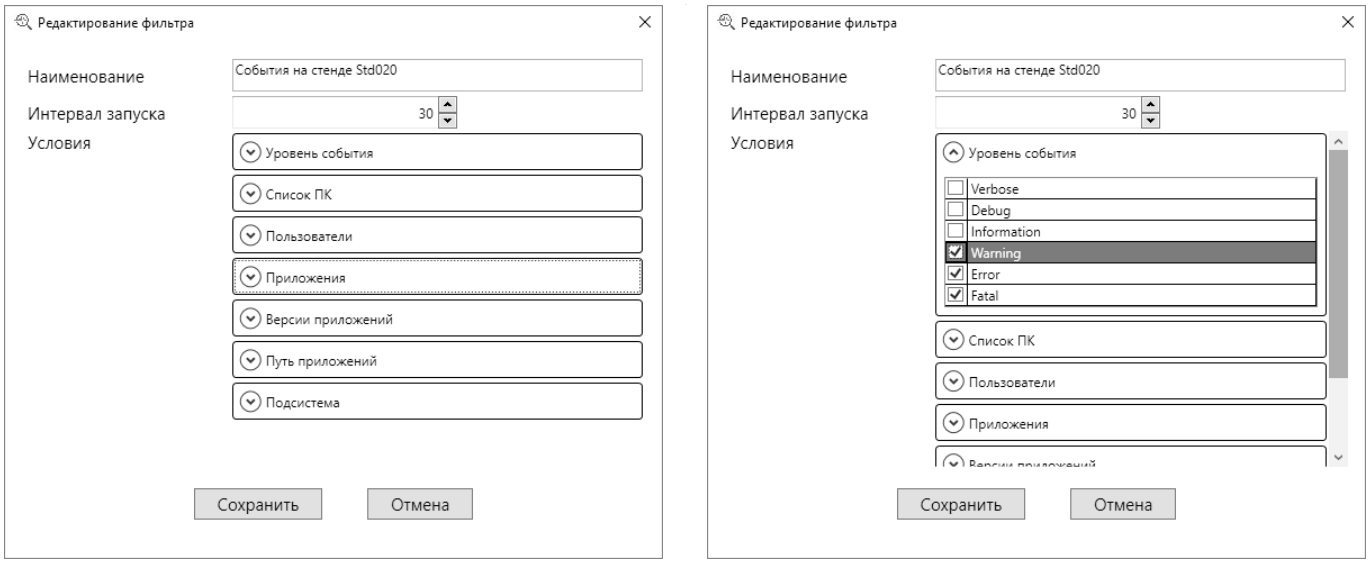

Рис. 5. Использование фильтров в мониторинге

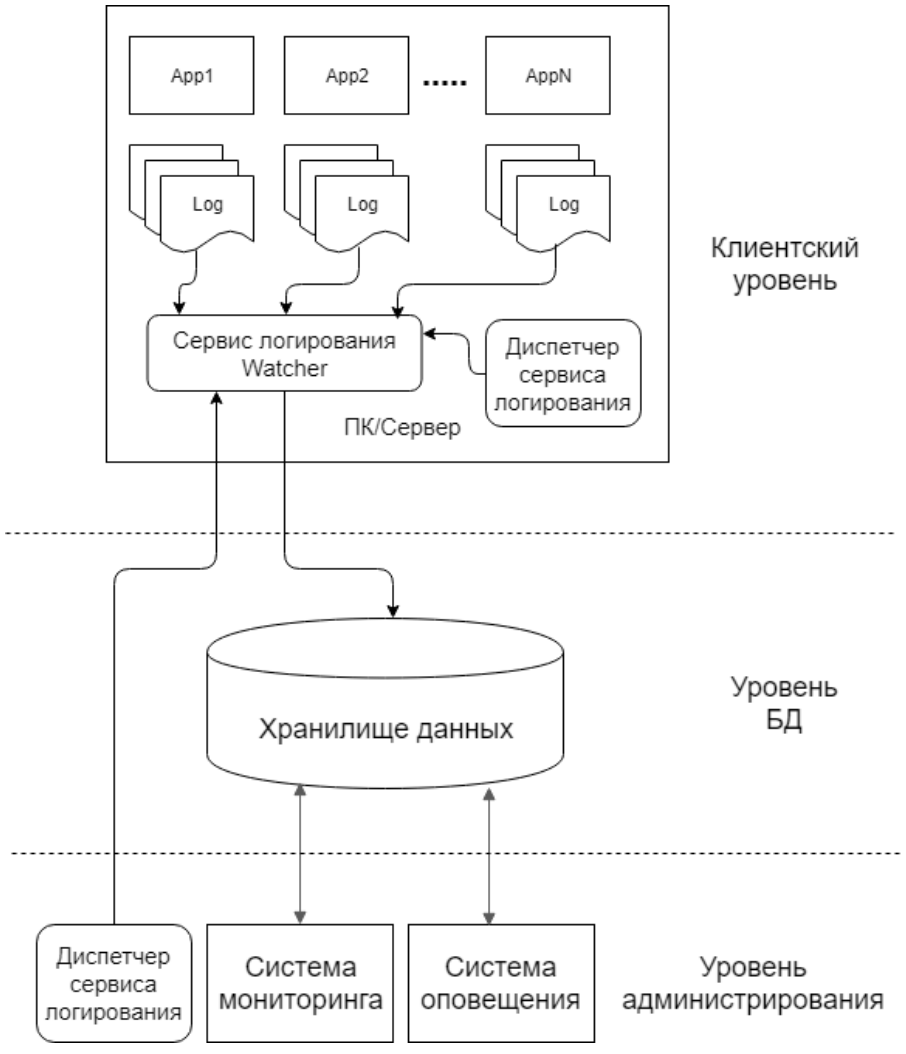

Рис. 6. Схема системы логирования

Заключение

Использование централизованного логирования является важным вопросом при проектировании и внедрении сложных распределенных информационных систем. Система логирования предоставляет возможность получать уникальные параметризированные данные логики работы приложений. Это позволяет процессу логирования функционировать отдельно, либо являться подсистемой или источником данных для полноценной системы мониторинга.

В данной работе выполнен анализ требований, предъявляемых к процессу логирования. По результатам анализа была спроектирована структурная схема системы логирования для сложных распределенных систем. В предложенной схеме процесс логирования логически разделен на три уровня. Рассмотренная схема может иметь гибкую программную реализацию, позволяющую менять программные модули каждого уровня независимо.

На основе этой схемы была выполнена разработка программного комплекса централизованного логирования успешно внедренного на практике.

Полученные результаты определили новые направления для дальнейшего исследования. Одним из перспективных направлений является задача прогнозирования состояния системы на основе событий, зарегистрированных в системе логирования.

ЛИТЕРАТУРА

1. Картамышева Е. С., Иванченко Д. С. Промышленная автоматизация в России: проблемы и их решения // Молодой ученый.— 2016.— № 28.— С. 93–95.
2. Гайфулин Т.А., Костомаров Д. С. Анализ современных систем мониторинга // Известия Тульского государственного университета. Технические науки.— 2013, Вып. 9., Ч. 2,— С. 51–55.
3. Serilog is a diagnostic logging library for.NET applications [Электронный ресурс] // github.com, 2018, URL: https://github.com/serilog/serilog
4. Павлов А.Д., Намиот Д. Е. Информационные системы на основе push-уведомлений // International Journal of Open Information Technologies, издательство Лаборатория Открытых Информационных Технологий факультета ВМК МГУ им. М. В. Ломоносова (Москва), том 2, № 8, 2014, с. 11–19

Журнал «Современная наука: актуальные проблемы теории и практики»

Пензенский государственный университет архитектуры и строительства