Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления» (ИУ)
КАФЕДРА «Информационная безопасность» (ИУ8)

Рубежный контроль №2 на тему "Проектирование предметной области"

по дисциплине «Объектно-ориентированный анализ и проектирование»

Вариант 7

Студент ИУ8-114 (Группа) Н.В. Железцов (И. О. Фамилия) _____ (Подпись, дата)

Преподаватель Ю. В. Молодцова (И. О. Фамилия) _____ (Подпись, дата)

Москва, 2025 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель задания — сформировать предметную модель информационной системы в соответствии с требованиями системной инженерии (ISO/IEC/IEEE 15288, 42010) и современными процессами жизненного цикла (ГОСТ Р 59793–2021).

Необходимо спроектировать предметную область для следующих систем:

- система мониторинга журналов безопасности (лог-менеджмент);
- система контроля исполнения политик безопасности рабочих станций.

ОСНОВНАЯ ЧАСТЬ

1 Система мониторинга журналов безопасности (лог-менеджмент)

Система мониторинга журналов безопасности (лог-менеджмент) — это инструмент, который централизованно собирает, хранит и анализирует журналы событий из разных источников, помогает выявлять инциденты, отслеживать активность, обеспечивать соответствие требованиям и ускорять расследование угроз.

Данная система является важной частью инфраструктуры информационной безопасности организации, поскольку обеспечивает прозрачность процессов, выявление аномалий, фиксацию действий пользователей и служб, а также служит основой для оперативного и ретроспективного анализа происходящих событий. Контекст предметной области включает управление информационными рисками, обеспечение непрерывности работы, поддержание соответствия нормативным требованиям и повышение уровня защищённости.

1.1 Границы и контекст предметной области

Границы системы определяют, какие функции и компоненты входят в лог-менеджмент и за что система отвечает непосредственно. Контекст предметной области показывает окружение, в рамках которого эта система функционирует, а также описывает связи с внешними элементами — источниками данных, пользователями, службами реагирования и регуляторами.

1.1.1 Заинтересованные стороны

В таблице 1 представлены ключевые заинтересованные стороны, которые взаимодействуют с системой, а также их основные потребности и цели.

Таблица 1 – Заинтересованные стороны системы лог-менеджмента

Заинтересованная сторона	Потребности	Цели
Руководство	Краткие отчёты, метрики рисков, видимость инцидентов	Управленческие решения, снижение рисков
ИТ / Системные администраторы	Централизованный сбор логов, простая интеграция, доступность данных	Стабильная работа инфраструктуры, быстрая диагностика
Служба информационной безопасности	Полные журналы, корреляция событий, алерты	Раннее выявление угроз, расследование инцидентов
SOC / Операционные аналитики	Данные в реальном времени, приоритизация, удобные панели	Сокращение времени реакции на инциденты
Разработчики	Доступ к логам приложений, структурированные данные	Быстрое устранение ошибок, повышение качества ПО
Аудит / Комплаенс	Контроль целостности логов, долгосрочное хранение, отчётность	Соответствие стандартам и требованиям регуляторов

1.1.2 Границы системы

Границы определяют, какие функциональные возможности предоставляет система, а какие задачи выполняются внешними элементами или другими системами. Внутренние компоненты лог-менеджмента включают:

- Модуль сбора логов (агенты, коннекторы).
- Централизованное хранилище журналов.
- Механизм нормализации и корреляции событий.
- Подсистема оповещений и отчётности.

– Интерфейсы визуализации (дашборды, поиск по логам).
– Механизмы аутентификации и авторизации пользователей.
– Подсистема резервного копирования и архивирования логов.

За пределами системы остаются те элементы, которые взаимодействуют с лог-менеджментом, но не управляются им напрямую:

– Источники логов (серверы, приложения, сетевое оборудование, СЗИ).
– Внешние системы реагирования и тикетирования.
– Пользователи и персонал (администраторы, SOC, аудиторы).
– Внешние регуляторы и требования стандартизации.

Таким образом формируются функциональные и организационные границы, в рамках которых система обеспечивает свою работу.

1.1.3 Внешние интерфейсы

Внешние интерфейсы определяют способы взаимодействия лог-менеджмента с окружением, обеспечивая передачу данных, получение уведомлений, формирование отчётов и интеграцию с другими сервисами. Они приведены в таблице 2.

Таблица 2 – Основные внешние интерфейсы системы лог-менеджмента

Внешний объект / система	Тип интерфейса	Назначение
Источники логов (серверы, сетевые устройства, приложения)	Syslog, API, агенты, файловые коллекторы	Передача событий и журналов в систему лог-менеджмента
Системы ИБ (IDS/IPS, WAF, антивирус, DLP)	Syslog, REST API, интеграционные коннекторы	Получение событий безопасности для анализа и корреляции
Система управления инцидентами (ITSM, SOAR)	REST API, вебхуки	Экспорт алертов и автоматизация реакции
Пользователи (SOC, администраторы, аудиторы)	Веб-интерфейс, ролевой доступ, отчёты	Просмотр логов, расследование, анализ, управление настройками
Внешние регуляторы (при проверках)	Экспорт отчётов, выгрузка архивов	Предоставление доказательной базы и журналов

1.2 Терминологический и объектный анализ предметной области

Терминологический и объектный анализ позволяет формализовать ключевые понятия предметной области лог-менеджмента, определить сущности, их характеристики и взаимосвязи. Это обеспечивает единое понятийное пространство для дальнейшего проектирования системы, уменьшает неоднозначность интерпретаций и служит основой для построения модели данных и бизнес-процессов.

1.2.1 Выделение сущностей и их определения

В соответствии с требованиями стандарта ISO 704 определения приводятся через ближайшее родовое понятие с указанием отличительных признаков. Ниже сформирован перечень сущностей, используемых в системе мониторинга журналов безопасности.

Журнал события (лог) Документированная запись о состоянии, действии или событии, зафиксированная информационной системой в определённый момент времени.

Событие безопасности Факт, отражающий действие или изменение состояния информационной системы, имеющее значение для оценки её защищённости.

Источник логов Система или компонент, генерирующий журналы событий и передающий их в лог-менеджмент.

Агент сбора логов Программный компонент, обеспечивающий получение, преобразование и передачу журналов от источника логов в центральную систему.

Хранилище логов Централизованный компонент, обеспечивающий долговременное хранение, поиск и доступ к журналам событий.

Нормализация событий Процесс приведения записей логов к единому структурированному формату для унификации анализа.

Корреляция событий Процесс установления взаимосвязей между разрозненными событиями с целью выявления аномалий или инцидентов.

Пользователь системы Лицо или роль, имеющая доступ к функциям системы лог-менеджмента.

Инцидент информационной безопасности Событие или совокупность событий, нарушающих или потенциально нарушающих конфиденциальность, целостность или доступность информации.

Оповещение (алерт) Автоматически сформированное уведомление о событиях, требующих внимания.

Отчёт Формализованный документ, описывающий результаты анализа, статистику и выводы на основе журналов событий.

1.2.2 Атрибуты сущностей

Для каждой сущности определены ключевые атрибуты, необходимые для её представления в системе.

- **Журнал события (лог):**
  - время события;
  - идентификатор источника;
  - тип события;
  - уровень важности;
  - текст сообщения;
  - структурированные поля (IP, пользователь, процесс и др.).
- **Событие безопасности:**
  - категория;
  - критичность;
  - контекст (объект, субъект);
  - статус (нормальное, подозрительное, инцидент).
- **Источник логов:**
  - тип (сервер, приложение, устройство);
  - уникальный идентификатор;
  - протокол передачи;
  - частота генерации событий.
- **Агент сбора логов:**
  - версия;
  - поддерживаемые форматы;

  - метод передачи;
  - состояние (активен/недоступен).
- **Хранилище логов:**
  - объём хранения;
  - срок ретенции;
  - механизмы шифрования;
  - политика доступа.
- **Нормализованное событие:**
  - формат записи;
  - значения нормализованных полей;
  - признак полноты.
- **Коррелированное событие:**
  - список исходных событий;
  - правило корреляции;
  - выявленный сценарий;
  - приоритизация.
- **Пользователь системы:**
  - роль;
  - уровень доступа;
  - статус учётной записи.
- **Алерт:**
  - тип триггера;
  - серьёзность;
  - метаданные (время, источник);
  - статус обработки.
- **Отчёт:**
  - тип отчёта;
  - период анализа;
  - ответственный;
  - формат (PDF, HTML, CSV).

1.2.3 Бизнес-правила, регулирующие поведение сущностей

Данный раздел определяет нормативные ограничения и логику работы системы.

а) Каждый источник логов должен иметь уникальный идентификатор и передавать журналы согласно установленному регламенту.

б) Все журналы событий должны содержать корректное время, синхронизированное по NTP.

в) Передача логов от агентов должна выполняться в защищённом виде (шифрование, аутентификация).

г) Все входящие события проходят нормализацию перед помещением в хранилище.

д) Корреляционные правила должны быть документированы, версионированы и регулярно пересматриваться.

е) Каждое коррелированное событие должно содержать ссылки на свои исходные события.

ж) Алерты формируются только при выполнении условий, определённых правилами корреляции или детекции.

з) Пользователь выполняет действия только в пределах своей роли и прав доступа.

и) Хранилище логов обеспечивает неизменность данных в течение срока ретенции, соответствующего требованиям регуляторов.

к) Отчёты формируются только на основе проверенных и нормализованных данных.

л) Любые изменения конфигурации системы фиксируются в журнале административных действий.

1.3 Структурная модель предметной области

1.3.1 UML-диаграмма классов

Ниже приведена UML-диаграмма классов, отражающая основные сущности предметной области лог-менеджмента, связи между ними, кардинальности и ключевые ограничения. Диаграмма предназначена для использования как средство согласования модели данных и границ ответственности компонентов системы.

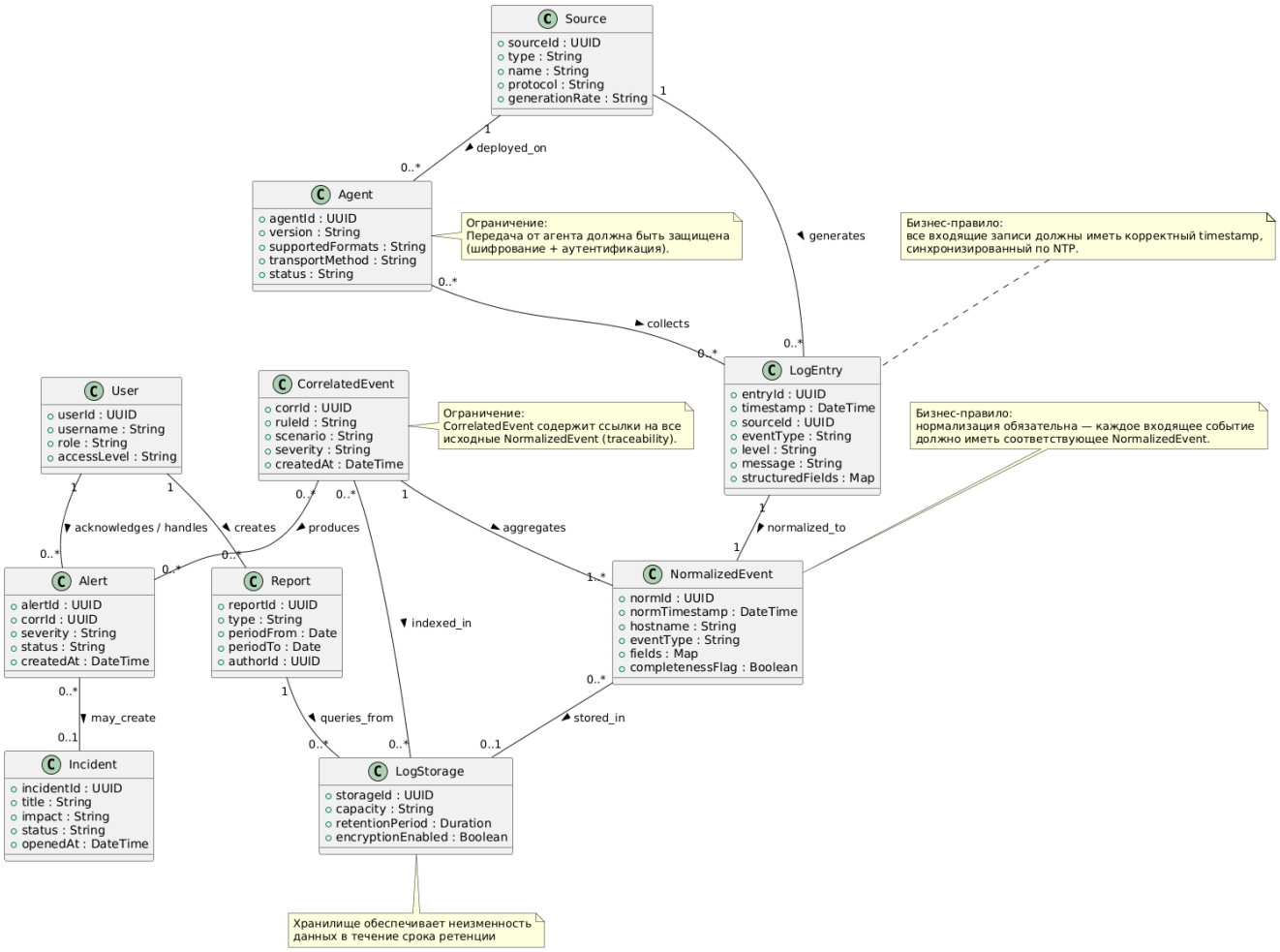

Рисунок 1 – UML-диаграмма классов системы лог-менеджмента

1.3.2 Пояснение к элементам диаграммы

- **Source (Источник логов)** — генерирует `LogEntry`; для каждого источника задан уникальный `sourceId`. Один источник может иметь ноль, одну или несколько установленных сущностей `Agent` (в зависимости от архитектуры — агент на хосте или централизованный коллектор).
- **Agent (Агент сбора)** — собирает записи событий и передаёт их в систему; один агент может собирать множество `LogEntry`.
- **LogEntry (Журнал события)** — исходная запись; согласно бизнес-правилу все входящие записи должны иметь корректный NTP-таймстемп и проходить нормализацию.
- **NormalizedEvent (Нормализованное событие)** — результат приведения `LogEntry` к единому формату; связь 1..1 с `LogEntry` (обязательно) в модели, т.к. нормализация равна требованию.

- **CorrelatedEvent (Коррелированное событие)** — объединяет несколько нормализованных событий по правилу корреляции; одно коррелированное событие агрегирует 1..* `NormalizedEvent`.
- **Alert (Алерт)** — создаётся на основе коррелированных событий; алерты могут приводить к созданию `Incident`.
- **Incident (Инцидент)** — следствие подтверждённых алертов, управляется процессом реагирования.
- **LogStorage (Хранилище)** — хранит исходные и/или нормализованные события, обеспечивает ретеншн, шифрование и неизменность в рамках регламентированных сроков.
- **User (Пользователь)** и **Report (Отчёт)** — пользователи создают отчёты и обрабатывают алерты; права операций ограничены ролями.

1.3.3 Основные ограничения и бизнес-правила

а) **Нормализация обязательна:** каждое поступившее событие должно иметь соответствующий `NormalizedEvent`. (см. диаграмму: `LogEntry 1..1 NormalizedEvent`)

б) **Синхронизация времени:** все записи обязаны иметь корректный временной штамп, синхронизированный по NTP.

в) **Аутентификация и шифрование:** передача от агентов в систему должна выполняться в защищённом виде.

г) **Трассируемость:** каждое `CorrelatedEvent` содержит ссылки на все исходные `NormalizedEvent`.

д) **Неизменность:** данные в `LogStorage` являются неизменяемыми в период ретенции.

е) **Ограничения доступа:** операции над алертами, инцидентами и отчётами доступны в соответствии с ролью и уровнем доступа пользователя.

1.4 Модель поведения системы

Модель поведения системы отражает динамическое взаимодействие сущностей и пользователей системы лог-менеджмента. Она позволяет формализовать сценарии взаимодействия, понять бизнес-процессы и определить жизненные циклы ключевых объектов.

1.4.1 Диаграмма прецедентов (Use Cases)

Диаграмма прецедентов показывает всех акторов системы и основные сценарии их взаимодействия с системой лог-менеджмента. Акторы включают: пользователей (SOC, администраторов, аудиторов), внешние источники логов, системы ИБ, системы управления инцидентами и регуляторов.

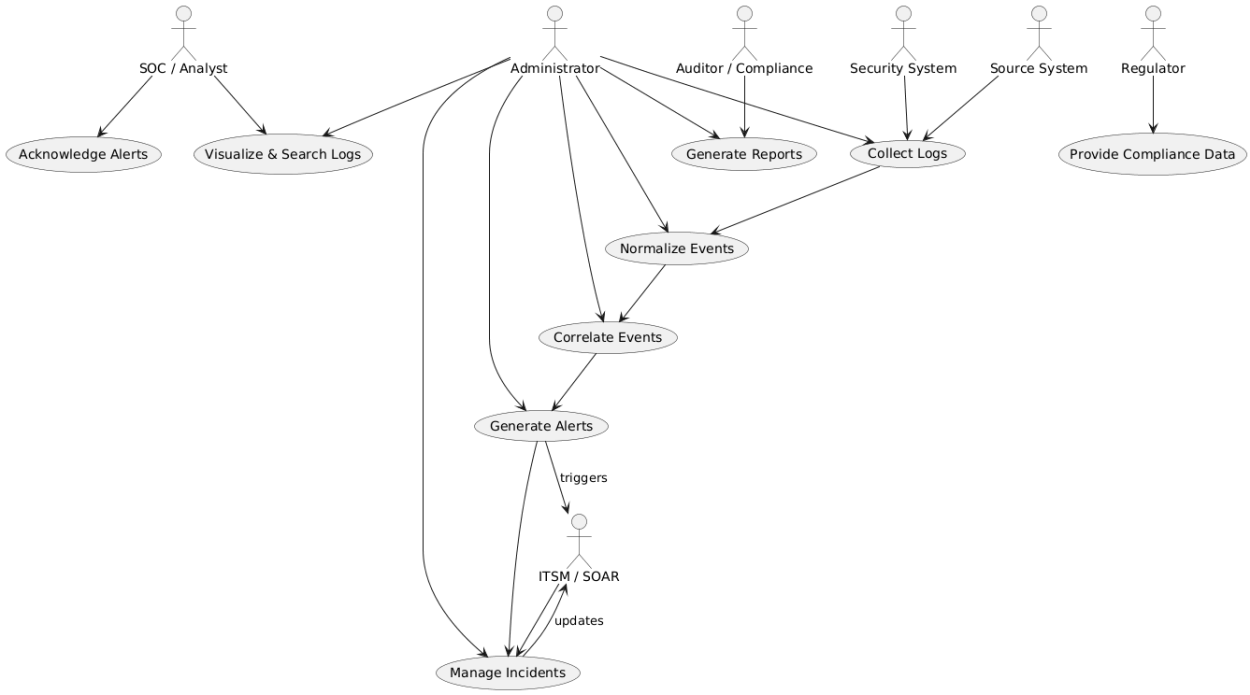

Рисунок 2 – Диаграмма прецедентов системы лог-менеджмента

1.4.2 Диаграмма деятельности (Activity Diagram)

Пример ключевого бизнес-процесса: обработка инцидента на основе коррелированного события.

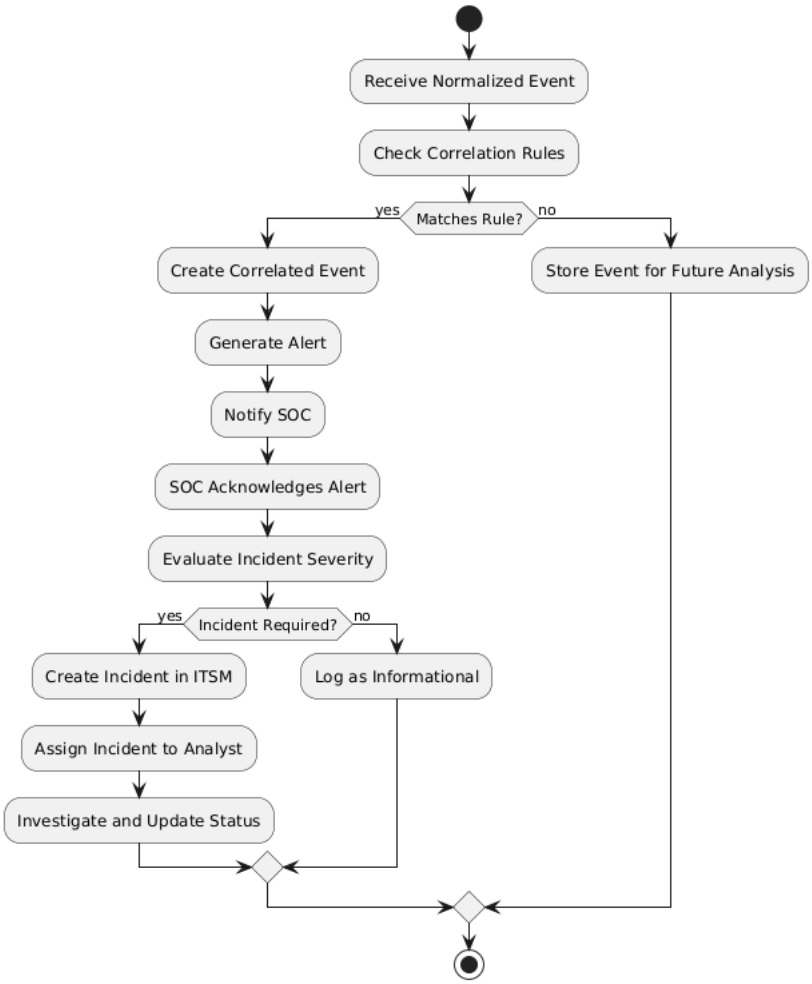

Рисунок 3 – Диаграмма деятельности системы лог-менеджмента

1.4.3 Диаграмма состояний (State Diagram)

Пример состояния для сущности `Incident`, которая имеет сложный жизненный цикл: открытие, обработка, закрытие.

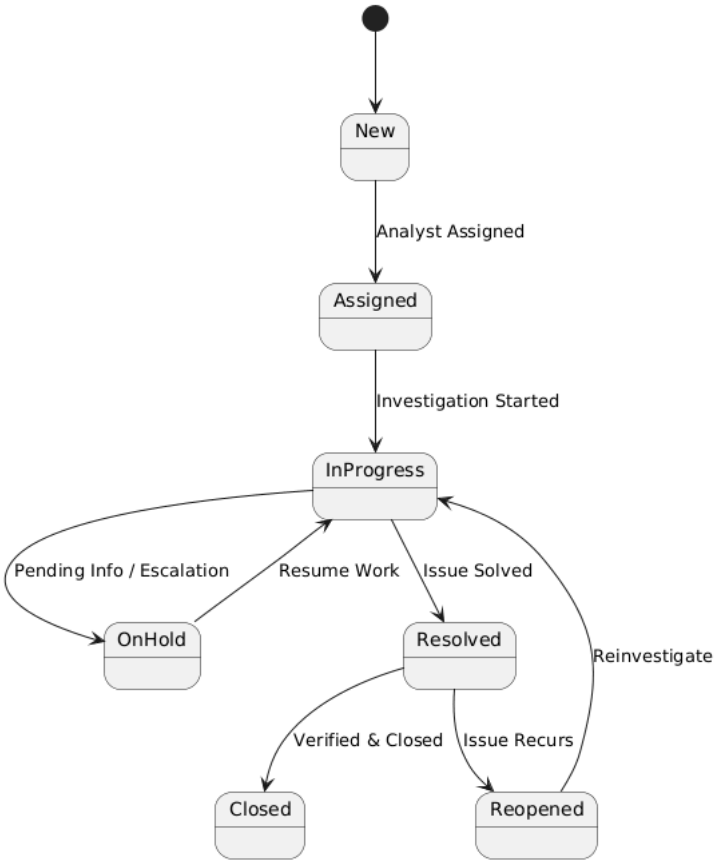

Рисунок 4 – Диаграмма состояний системы лог-менеджмента

1.4.4 Пояснения

- **Диаграмма прецедентов** показывает, кто и какие функции системы использует, включая автоматизированные взаимодействия с внешними источниками и системами.
- **Диаграмма деятельности** иллюстрирует поток обработки события от нормализованного события до создания инцидента и реакции SOC.
- **Диаграмма состояний** описывает жизненный цикл инцидента, включая переходы между состояниями: `New` → `Assigned` → `InProgress` → `Resolved` → `Closed`, с возможностью повторного открытия.

1.5 Верификация модели предметной области

Верификация модели предметной области необходима для обеспечения её корректности, полноты и соответствия требованиям заинтересованных сторон. Процесс верификации позволяет выявить противоречия, ошибки терминологии и несоответствия стандартам проектирования. В контексте системы мониторинга журналов безопасности проверка проводится по следующим критериям.

1.5.1 Критерии верификации

а) **Непротиворечивость**

Все элементы модели должны быть логически согласованы:

- отсутствуют противоречивые определения сущностей и атрибутов;
- кардинальности и связи между классами не конфликтуют;
- бизнес-правила не содержат взаимно исключающих условий.

б) **Полнота**

Модель должна охватывать всю предметную область:

- учтены все ключевые сущности, процессы и взаимодействия;
- все прецеденты использования и активности включены;
- жизненные циклы основных объектов описаны.

в) **Соответствие требованиям заинтересованных сторон**

Проверка выполняется на основе анализа таблицы 1:

- все потребности и цели акторов реализованы в модели;
- каждый прецедент и процесс обеспечивает удовлетворение требований;
- предусмотрены интерфейсы для внешних систем и регуляторов.

г) **Корректность терминологического аппарата (ISO 704)**

Проверяется единообразие и точность терминов:

- определения сущностей соответствуют правилам ISO 704 (родовое понятие + отличительные признаки);
- нет дублирования терминов или неоднозначных формулировок;
- атрибуты и свойства корректно связаны с сущностями.

д) **Соответствие процессам ISO/IEC/IEEE 42010**

Модель проверяется на архитектурное соответствие:

- архитектурные виды (viewpoints) и представления (views) соответствуют требованиям стандартов;
- диаграммы классов, прецедентов, деятельности и состояний отражают архитектурные решения;
- соблюдены принципы трассируемости между требованиями, бизнес-правилами и элементами модели.

1.5.2 Выводы по верификации

Проведённая проверка модели предметной области показала:

- Модель лог-менеджмента является непротиворечивой и логически согласованной.
- Все ключевые сущности, процессы и сценарии использования включены, обеспечивая полноту модели.
- Потребности заинтересованных сторон реализованы через прецеденты использования и функциональные компоненты системы.
- Терминология согласована с ISO 704, определения и атрибуты корректны.
- Диаграммы и архитектурные представления соответствуют стандартам ISO/IEC/IEEE 42010.

Таким образом, модель предметной области прошла верификацию и может быть использована для дальнейшего проектирования системы лог-менеджмента и её архитектуры.

1.6 Заключение

Настоящий отчёт содержит результаты проектирования предметной области системы мониторинга журналов безопасности (лог-менеджмент). Проектирование выполнено с учётом требований ГОСТ Р 7.0.97–2016, ГОСТ Р 59793–2021, ISO/IEC/IEEE 15288, ISO/IEC 12207 и ISO/IEC/IEEE 42010.

Архитектурный документ по ISO/IEC/IEEE 42010 включает следующие компоненты:

- **Stakeholders:** руководство организации, ИТ-администраторы, служба информационной безопасности, SOC, разработчики, аудиторы и внешние системы.
- **Concerns:** корректность данных, своевременное выявление инцидентов, безопасность и целостность информации, соответствие требованиям регуляторов, доступность и надёжность системы.

- **Viewpoints:** структурная (UML class), поведенческая (activity, use case), информационная, эксплуатационная.
- **Views:** UML-диаграммы классов, диаграммы прецедентов, диаграммы деятельности, диаграммы состояний, словарь сущностей, описание бизнес-процессов.
- **Correspondence rules:** соответствие бизнес-правил моделям, непротиворечивость связей, корректность терминологии.
- **Rationale:** архитектура обеспечивает прозрачность процессов, своевременное реагирование на инциденты, целостность и сохранность логов, автоматизацию обработки событий и формирование отчётов.

В ходе проектирования были выполнены:

а) Анализ заинтересованных сторон и их потребностей, определение границ и контекста системы.

б) Терминологический и объектный анализ предметной области, выделение сущностей, атрибутов и бизнес-правил.

в) Построение структурной модели системы с использованием UML-диаграммы классов, отражающей сущности, их связи, кардинальности и ограничения.

г) Моделирование поведения системы: диаграммы прецедентов, деятельности и состояний.

д) Верификация модели на непротиворечивость, полноту, соответствие требованиям заинтересованных сторон, корректность терминологии и соответствие стандартам ISO/IEC/IEEE 42010.

Проверка модели показала её непротиворечивость, полноту и соответствие архитектурным требованиям. Терминологический аппарат соответствует ISO 704, все бизнес-правила корректно отражены в модели, а диаграммы обеспечивают трассируемость между требованиями, процессами и объектами системы.

Таким образом, созданная модель предметной области является надежной основой для дальнейшего проектирования системы лог-менеджмента, её архитектуры, бизнес-процессов и реализации компонентов. Архитектура обеспечивает транспарентность процессов, эффективность реагирования на инциденты и соблюдение нормативных требований организации.

2 Система контроля исполнения политик безопасности рабочих станций

Система контроля исполнения политик безопасности рабочих станций (Endpoint Security Policy Enforcement) — это инструмент, обеспечивающий мониторинг и контроль соблюдения установленных политик безопасности на рабочих станциях пользователей, включая обновления ПО, антивирусную защиту, настройки ОС и конфигурации безопасности.

Данная система является ключевой частью инфраструктуры информационной безопасности организации, поскольку предотвращает нарушения политик, снижает риски компрометации конечных устройств и обеспечивает соответствие внутренним и внешним нормативным требованиям.

2.1 Границы и контекст предметной области

Границы системы определяют функции и компоненты, которые контролируются непосредственно системой, а контекст показывает её взаимодействие с пользователями, ИТ-персоналом и внешними сервисами.

2.1.1 Заинтересованные стороны

Таблица 3 – Заинтересованные стороны системы контроля политик безопасности

Заинтересованная сторона	Потребности	Цели
Руководство	Отчёты о соблюдении политик, метрики рисков	Снижение угроз, принятие управленческих решений
ИТ / Системные администраторы	Удалённое управление политиками, отчёты об отклонениях	Поддержка рабочих станций в безопасном состоянии
Служба информационной безопасности	Контроль выполнения политик, выявление нарушений	Снижение рисков утечек и инцидентов
SOC / Операционные аналитики	События безопасности с рабочих станций	Реагирование на инциденты
Пользователи	Сообщения о нарушениях политик, рекомендации	Обеспечение безопасной работы рабочих станций
Аудит / Комплаенс	История выполнения политик, отчёты	Соответствие стандартам и внутренним регламентам

2.1.2 Границы системы

Внутренние компоненты системы:

- Агент контроля на рабочей станции.
- Централизованная консоль управления политиками.
- Модуль анализа соответствия политик.
- Подсистема уведомлений и отчётности.
- Интерфейсы для администраторов и аудиторов.
- Подсистема хранения логов и конфигураций.

Внешние элементы:

– Рабочие станции и пользователи.
– Сервера обновлений и сторонние сервисы безопасности.
– Регуляторы и внутренние аудиторы.

2.1.3 Внешние интерфейсы

Таблица 4 – Основные внешние интерфейсы системы контроля политик

Внешний объект / система	Тип интерфейса	Назначение
Рабочие станции	Агент, API, SNMP	Контроль и отчёт о соблюдении политик
Сервера обновлений	API, пакетные менеджеры	Проверка и установка обновлений
Антивирус и защитное ПО	API, интеграционные коннекторы	Проверка статуса защиты
ИТ-администраторы	Веб-консоль, отчёты	Управление политиками и инцидентами
Аудиторы / регуляторы	Экспорт отчётов, CSV/PDF	Проверка соблюдения требований

2.2 Терминологический и объектный анализ предметной области

Рабочая станция Компьютер пользователя, на котором должны соблюдаться политики безопасности.

Политика безопасности Правила и требования к конфигурации системы, ПО и поведению пользователя.

Агент контроля Программный компонент, обеспечивающий мониторинг и исполнение политик на рабочей станции.

Нарушение политики Событие, когда рабочая станция не соответствует установленной политике.

Отчёт Документ о состоянии соответствия политик, генерируемый системой.

Администратор Пользователь, управляющий политиками и контролирующий соблюдение.

2.2.1 Атрибуты сущностей

- **Рабочая станция:** идентификатор, статус соответствия, установленное ПО, конфигурация безопасности.
- **Политика безопасности:** уникальный идентификатор, тип политики, требования, приоритет.
- **Агент контроля:** версия, состояние, частота проверок, результаты сканирования.
- **Нарушение политики:** тип, уровень критичности, дата и время, идентификатор рабочей станции.
- **Отчёт:** период, включаемые нарушения, ответственный администратор, формат.

2.2.2 Бизнес-правила

а) Каждая рабочая станция должна иметь установленного агента контроля.
б) Агент периодически проверяет соблюдение всех политик.
в) Нарушения фиксируются и передаются в централизованную консоль.
г) Администратор обязан реагировать на критические нарушения.
д) Отчёты генерируются автоматически в установленные сроки.
е) История нарушений хранится в течение периода, соответствующего требованиям.

2.3 Структурная модель предметной области

2.3.1 UML-диаграмма классов

Ниже приведена UML-диаграмма классов, отражающая основные сущности предметной области системы контроля исполнения политик безопасности рабочих станций, связи между ними, кардинальности и ключевые ограничения. Диаграмма предназначена для согласования модели данных и границ ответственности компонентов системы.

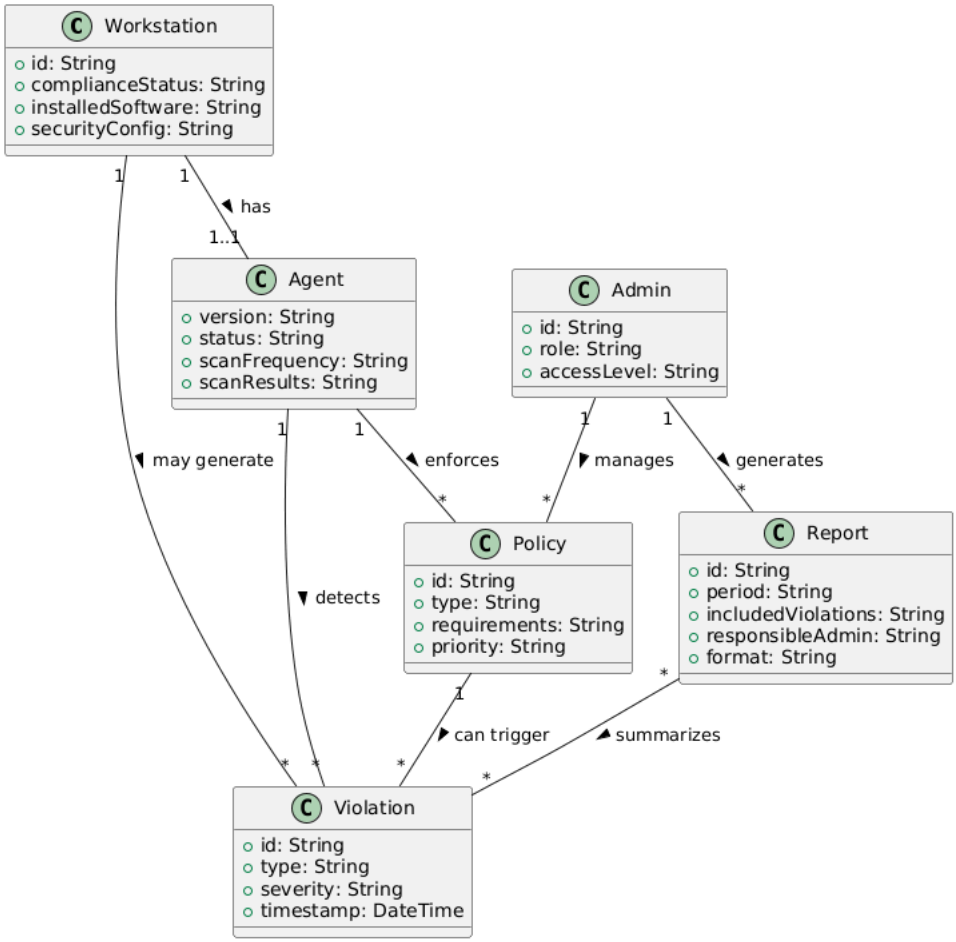

Рисунок 5 – UML-диаграмма классов системы контроля исполнения политик безопасности рабочих станций

2.3.2 Пояснение к элементам диаграммы

- **Workstation (Рабочая станция)** — конечное устройство, на котором должны соблюдаться политики безопасности; каждая рабочая станция имеет одного агента контроля.
- **Agent (Агент контроля)** — устанавливается на рабочей станции, выполняет проверки соответствия политикам и фиксирует нарушения.
- **Policy (Политика безопасности)** — правила и требования к конфигурации, ПО и поведению пользователя; нарушения фиксируются как `Violation`.
- **Violation (Нарушение политики)** — событие, фиксирующее несоответствие политики; может быть обнаружено агентом и включено в отчёт.
- **Admin (Администратор)** — управляет политиками, реагирует на критические нарушения и формирует отчёты.

– **Report (Отчёт)** — документ о состоянии соблюдения политик; содержит информацию о выявленных нарушениях и состоянии рабочих станций.

2.3.3 Основные ограничения и бизнес-правила

а) **Обязательная установка агента:** каждая рабочая станция должна иметь установленного агента контроля.
б) **Регулярная проверка политик:** агент периодически сканирует рабочую станцию на соответствие всем политикам.
в) **Фиксация нарушений:** все несоответствия фиксируются как `Violation` и передаются в централизованную консоль.
г) **Реакция на критические нарушения:** администратор обязан реагировать на нарушения с высоким уровнем критичности.
д) **Генерация отчётов:** отчёты создаются автоматически или по запросу администратора, включают сведения о выявленных нарушениях.
е) **Хранение истории:** данные о нарушениях сохраняются в течение установленного периода для аудита и комплаенса.

2.4 Модель поведения системы

Модель поведения системы отражает динамическое взаимодействие сущностей и пользователей системы контроля исполнения политик безопасности рабочих станций. Она позволяет формализовать сценарии взаимодействия, понять бизнес-процессы и определить жизненные циклы ключевых объектов.

2.4.1 Диаграмма прецедентов (Use Cases)

Диаграмма прецедентов показывает всех акторов системы и основные сценарии взаимодействия с системой контроля политик. Акторы включают: пользователей (администраторов, аудиторов), рабочие станции, агенты контроля, внешние сервисы обновлений и регуляторов.

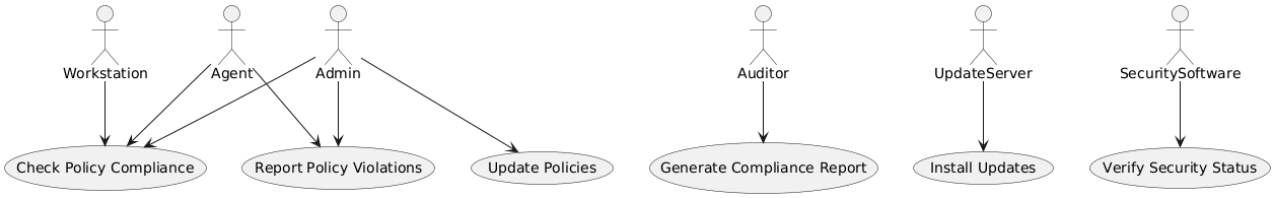

Рисунок 6 – Диаграмма прецедентов системы контроля исполнения политик безопасности рабочих станций

2.4.2 Диаграмма деятельности (Activity Diagram)

Пример ключевого бизнес-процесса: проверка соблюдения политики и обработка нарушения на рабочей станции.

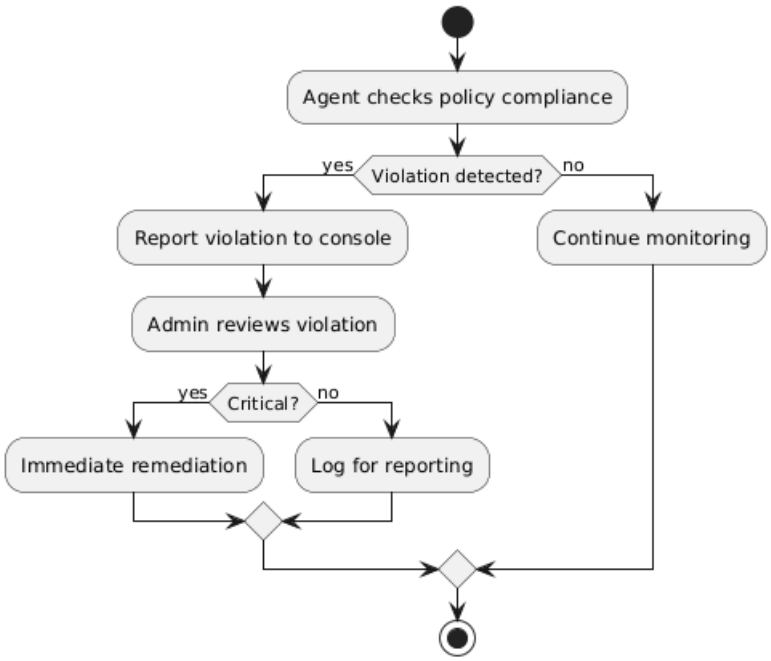

Рисунок 7 – Диаграмма деятельности системы контроля исполнения политик безопасности рабочих станций

2.4.3 Диаграмма состояний (State Diagram)

Пример состояния для сущности `Violation`, которая имеет жизненный цикл: обнаружение, реакция, подтверждение и закрытие.

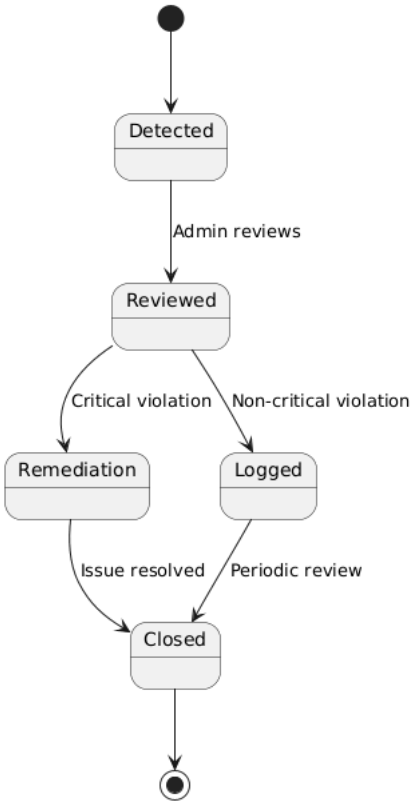

Рисунок 8 – Диаграмма состояний сущности `Violation` в системе контроля исполнения политик безопасности

2.4.4 Пояснения

- **Диаграмма прецедентов** показывает акторов системы и функции, которые они используют, включая автоматизированные проверки и уведомления.
- **Диаграмма деятельности** иллюстрирует процесс проверки политики, фиксации нарушений и уведомления администратора.
- **Диаграмма состояний** описывает жизненный цикл нарушения (`Violation`) с переходами: `Detected` → `Notified` → `InProgress` → `Resolved` → `Closed`.

2.5 Верификация модели предметной области

Верификация модели предметной области необходима для обеспечения её корректности, полноты и соответствия требованиям заинтересованных сторон. Процесс верификации позволяет выявить противоречия, ошибки терминологии и несоответствия стандартам проектирования. В контексте системы контроля исполнения политик безопасности рабочих станций проверка проводится по следующим критериям.

2.5.1 Критерии верификации

а) **Непротиворечивость**

Все элементы модели должны быть логически согласованы:

- отсутствуют противоречивые определения сущностей и атрибутов;
- кардинальности и связи между классами не конфликтуют;
- бизнес-правила не содержат взаимно исключающих условий.

б) **Полнота**

Модель должна охватывать всю предметную область:

- учтены все ключевые сущности, процессы и взаимодействия;
- все прецеденты использования и активности включены;
- жизненные циклы основных объектов описаны.

в) **Соответствие требованиям заинтересованных сторон**

Проверка выполняется на основе анализа таблицы 3:

- все потребности и цели акторов реализованы в модели;
- каждый прецедент и процесс обеспечивает удовлетворение требований;
- предусмотрены интерфейсы для внешних систем, регуляторов и аудиторов.

г) **Корректность терминологического аппарата (ISO 704)**

Проверяется единообразие и точность терминов:

- определения сущностей соответствуют правилам ISO 704 (родовое понятие + отличительные признаки);
- нет дублирования терминов или неоднозначных формулировок;
- атрибуты и свойства корректно связаны с сущностями.

д) **Соответствие процессам ISO/IEC/IEEE 42010**

Модель проверяется на архитектурное соответствие:

- архитектурные виды (viewpoints) и представления (views) соответствуют требованиям стандартов;
- диаграммы классов, прецедентов, деятельности и состояний отражают архитектурные решения;
- соблюдены принципы трассируемости между требованиями, бизнес-правилами и элементами модели.

2.5.2 Выводы по верификации

Проведённая проверка модели предметной области показала:

- Модель системы контроля исполнения политик безопасности рабочих станций является непротиворечивой и логически согласованной.
- Все ключевые сущности, процессы и сценарии использования включены, обеспечивая полноту модели.
- Потребности заинтересованных сторон реализованы через прецеденты использования и функциональные компоненты системы.
- Терминология согласована с ISO 704, определения и атрибуты корректны.
- Диаграммы и архитектурные представления соответствуют стандартам ISO/IEC/IEEE 42010.

Таким образом, модель предметной области прошла верификацию и может быть использована для дальнейшего проектирования системы контроля исполнения политик безопасности рабочих станций и её архитектуры.

2.6 Заключение

Настоящий отчёт содержит результаты проектирования предметной области системы контроля исполнения политик безопасности рабочих станций. Проектирование выполнено с учётом требований ГОСТ Р 7.0.97–2016, ГОСТ Р 59793–2021, ISO/IEC/IEEE 15288, ISO/IEC 12207 и ISO/IEC/IEEE 42010.

Архитектурный документ по ISO/IEC/IEEE 42010 включает следующие компоненты:

- **Stakeholders:** руководство организации, ИТ-администраторы, служба информационной безопасности, SOC, пользователи, аудиторы и внешние сервисы.

- **Concerns:** соблюдение политик безопасности на рабочих станциях, своевременное выявление нарушений, защита конечных устройств, соответствие требованиям регуляторов, доступность и надёжность системы.
- **Viewpoints:** структурная (UML class), поведенческая (activity, use case), информационная, эксплуатационная.
- **Views:** UML-диаграммы классов, диаграммы прецедентов, диаграммы деятельности, диаграммы состояний, словарь сущностей, описание бизнес-процессов.
- **Correspondence rules:** соответствие бизнес-правил моделям, непротиворечивость связей, корректность терминологии.
- **Rationale:** архитектура обеспечивает контроль исполнения политик, прозрачность процессов, своевременное реагирование на нарушения, накопление доказательной базы и автоматизацию формирования отчётов.

В ходе проектирования были выполнены:

а) Анализ заинтересованных сторон и их потребностей, определение границ и контекста системы.

б) Терминологический и объектный анализ предметной области, выделение сущностей, атрибутов и бизнес-правил.

в) Построение структурной модели системы с использованием UML-диаграммы классов, отражающей сущности, их связи, кардинальности и ограничения.

г) Моделирование поведения системы: диаграммы прецедентов, деятельности и состояний.

д) Верификация модели на непротиворечивость, полноту, соответствие требованиям заинтересованных сторон, корректность терминологии и соответствие стандартам ISO/IEC/IEEE 42010.

Проверка модели показала её непротиворечивость, полноту и соответствие архитектурным требованиям. Терминологический аппарат соответствует ISO 704, все бизнес-правила корректно отражены в модели, а диаграммы обеспечивают трассируемость между требованиями, процессами и объектами системы.

Таким образом, созданная модель предметной области является надежной основой для дальнейшего проектирования системы контроля исполнения политик безопасности рабочих станций, её архитектуры, бизнес-процессов и реализации компонентов. Архитектура обеспечивает прозрачность процессов, эффективность реагирования на нарушения и соблюдение нормативных требований организации.

ЗАКЛЮЧЕНИЕ

В рамках рубежного контроля были спроектированы и верифицированы модели предметной области двух ключевых систем информационной безопасности организации:

а) **Система мониторинга журналов безопасности (лог-менеджмент)**, обеспечивающая централизованный сбор, хранение, нормализацию, корреляцию и анализ событий, формирование алертов и инцидентов, а также подготовку отчётов для заинтересованных сторон.

б) **Система контроля исполнения политик безопасности рабочих станций**, обеспечивающая мониторинг и контроль соблюдения установленных политик безопасности на конечных устройствах, фиксацию нарушений, уведомления и отчётность.

Проектирование выполнялось с учётом требований ГОСТ Р 7.0.97–2016, ГОСТ Р 59793–2021, ISO/IEC/IEEE 15288, ISO/IEC/IEEE 12207 и ISO/IEC/IEEE 42010.

В ходе работы были выполнены следующие ключевые этапы:

- Определение заинтересованных сторон и их требований, формирование контекста и границ систем.
- Терминологический и объектный анализ, выделение сущностей, их атрибутов и бизнес-правил.
- Построение структурных моделей с использованием UML-диаграмм классов, отражающих связи, кардинальности и ограничения.
- Моделирование поведения систем через диаграммы прецедентов, деятельности и состояний.
- Верификация моделей на непротиворечивость, полноту, соответствие требованиям заинтересованных сторон, корректность терминологии и соответствие стандартам ISO/IEC/IEEE 42010.

Проверка моделей показала их непротиворечивость, полноту и соответствие архитектурным требованиям. Терминологический аппарат соответствует ISO 704, бизнес-правила корректно отражены в моделях, а диаграммы обеспечивают трассируемость между требованиями, процессами и объектами систем.

Таким образом, созданные модели предметной области обеспечивают надёжную основу для дальнейшего проектирования, внедрения и эксплуатации систем рубежного контроля. Архитектуры систем гарантируют прозрачность процессов, своевременное выявление инцидентов и нарушений, а также соблюдение нормативных требований организации.

ПРИЛОЖЕНИЕ А

PlantUML код для диаграмм системы лог-менеджмента

Листинг А.1 – Диаграмма классов

```
@startuml LogManagementDomain

' --- Классысущности () ---
class Source {
  + sourceId : UUID
  + type : String
  + name : String
  + protocol : String
  + generationRate : String
}

class Agent {
  + agentId : UUID
  + version : String
  + supportedFormats : String
  + transportMethod : String
  + status : String
}

class LogEntry {
  + entryId : UUID
  + timestamp : DateTime
  + sourceId : UUID
  + eventType : String
  + level : String
  + message : String
  + structuredFields : Map
}

class NormalizedEvent {
  + normId : UUID
  + normTimestamp : DateTime
  + hostname : String
  + eventType : String
  + fields : Map
  + completenessFlag : Boolean
}

class CorrelatedEvent {
  + corrId : UUID
  + ruleId : String
  + scenario : String
  + severity : String
  + createdAt : DateTime
}

class Alert {
  + alertId : UUID
  + corrId : UUID
  + severity : String
  + status : String
  + createdAt : DateTime
}

class Incident {
  + incidentId : UUID
  + title : String
  + impact : String
  + status : String
  + openedAt : DateTime
}
```

```
class LogStorage {
  + storageId : UUID
  + capacity : String
  + retentionPeriod : Duration
  + encryptionEnabled : Boolean
}

class User {
  + userId : UUID
  + username : String
  + role : String
  + accessLevel : String
}

class Report {
  + reportId : UUID
  + type : String
  + periodFrom : Date
  + periodTo : Date
  + authorId : UUID
}

' --- Связиикардинальности   ---
Source "1" -- "0..*" Agent : deployed_on >
Agent "0..*" -- "0..*" LogEntry : collects >
Source "1" -- "0..*" LogEntry : generates >
LogEntry "1" -- "1" NormalizedEvent : normalized_to >
NormalizedEvent "0..*" -- "0..1" LogStorage : stored_in >
CorrelatedEvent "1" -- "1..*" NormalizedEvent : aggregates >
CorrelatedEvent "0..*" -- "0..*" LogStorage : indexed_in >
CorrelatedEvent "0..*" -- "0..*" Alert : produces >
Alert "0..*" -- "0..1" Incident : may_create >
User "1" -- "0..*" Alert : acknowledges / handles >
User "1" -- "0..*" Report : creates >
Report "1" -- "0..*" LogStorage : queries_from >

' --- Ограничения / заметки ---
note top of LogEntryБизнесправило
  -:всевходящиезаписидолжныиметькорректный
        timestamp,синхронизированныйпо
    NTP.
end note

note top of NormalizedEventБизнесправило
  -:нормализацияобязательна-каждоевходящеесобытиедолжноиметьсоответствующее

     NormalizedEvent.
end note

note Хранилище" обеспечиваетнеизменность \данныхn втечениесрокаретенции   " as N1
LogStorage .. N1

note right of CorrelatedEventОграничение
  :
  CorrelatedEvent содержитссылкинавсеисходные
   NormalizedEvent (traceability).
end note

note right of AgentОграничение
  :Передачаотагентадолжнабытьзащищенашифрование

  ( + аутентификация).
end note

@enduml
```

Листинг А.2 – Диаграмма прецедентов

```
@startuml UseCases_LogManagement

actor "SOC / Analyst" as SOC
actor "Administrator" as Admin
actor "Auditor / Compliance" as Auditor
actor "Source System" as Source
actor "Security System" as SecSys
actor "ITSM / SOAR" as ITSM
actor "Regulator" as Reg

usecase "Collect Logs" as UC1
usecase "Normalize Events" as UC2
usecase "Correlate Events" as UC3
usecase "Generate Alerts" as UC4
usecase "Manage Incidents" as UC5
usecase "Visualize & Search Logs" as UC6
usecase "Generate Reports" as UC7
usecase "Acknowledge Alerts" as UC8
usecase "Provide Compliance Data" as UC9

' --- связиакторовипрецедентов    ---
Source --> UC1
SecSys --> UC1
UC1 --> UC2
UC2 --> UC3
UC3 --> UC4
UC4 --> UC5
SOC --> UC6
SOC --> UC8
Admin --> UC1
Admin --> UC2
Admin --> UC3
Admin --> UC4
Admin --> UC5
Admin --> UC6
Admin --> UC7
Auditor --> UC7
Reg --> UC9
ITSM --> UC5
UC4 --> ITSM : triggers
UC5 --> ITSM : updates

@enduml
```

Листинг А.3 – Диаграмма деятельности

```
@startuml Activity_LogIncident

start
:Receive Normalized Event;
:Check Correlation Rules;
if (Matches Rule?) then (yes)
  :Create Correlated Event;
  :Generate Alert;
  :Notify SOC;
  :SOC Acknowledges Alert;
  :Evaluate Incident Severity;
  if (Incident Required?) then (yes)
    :Create Incident in ITSM;
    :Assign Incident to Analyst;
    :Investigate and Update Status;
  else (no)
    :Log as Informational;
  endif
else (no)
  :Store Event for Future Analysis;
endif
stop
```

```
@enduml
```

Листинг А.4 – Диаграмма состояний

```
@startuml State_Incident

[*] --> New

New --> Assigned : Analyst Assigned
Assigned --> InProgress : Investigation Started
InProgress --> OnHold : Pending Info / Escalation
InProgress --> Resolved : Issue Solved
OnHold --> InProgress : Resume Work
Resolved --> Closed : Verified & Closed
Resolved --> Reopened : Issue Recurs
Reopened --> InProgress : Reinvestigate

@enduml
```

ПРИЛОЖЕНИЕ Б

PlantUML код для диаграмм системы контроля исполнения политик безопасности рабочих станций

Листинг Б.5 – Диаграмма классов

```
@startuml
class Workstation {
    +id: String
    +complianceStatus: String
    +installedSoftware: String
    +securityConfig: String
}

class Policy {
    +id: String
    +type: String
    +requirements: String
    +priority: String
}

class Agent {
    +version: String
    +status: String
    +scanFrequency: String
    +scanResults: String
}

class Violation {
    +id: String
    +type: String
    +severity: String
    +timestamp: DateTime
}

class Admin {
    +id: String
    +role: String
    +accessLevel: String
}

class Report {
    +id: String
    +period: String
    +includedViolations: String
    +responsibleAdmin: String
    +format: String
}

Workstation "1" -- "1..1" Agent : has >
Agent "1" -- "*" Policy : enforces >
Workstation "1" -- "*" Violation : may generate >
Agent "1" -- "*" Violation : detects >
Admin "1" -- "*" Policy : manages >
Admin "1" -- "*" Report : generates >
Report "*" -- "*" Violation : summarizes >
Policy "1" -- "*" Violation : can trigger >
@enduml
```

Листинг Б.6 – Диаграмма прецедентов

```
@startuml
actor Admin
actor Auditor
```

```
actor Workstation
actor Agent
actor UpdateServer
actor SecuritySoftware

usecase "Check Policy Compliance" as UC1
usecase "Report Policy Violations" as UC2
usecase "Update Policies" as UC3
usecase "Install Updates" as UC4
usecase "Verify Security Status" as UC5
usecase "Generate Compliance Report" as UC6

Admin --> UC1
Admin --> UC2
Admin --> UC3
Auditor --> UC6
Workstation --> UC1
Agent --> UC1
Agent --> UC2
UpdateServer --> UC4
SecuritySoftware --> UC5
@enduml
```

Листинг Б.7 – Диаграмма деятельности

```
@startuml
start
:Agent checks policy compliance;
if (Violation detected?) then (yes)
  :Report violation to console;
  :Admin reviews violation;
  if (Critical?) then (yes)
    :Immediate remediation;
  else (no)
    :Log for reporting;
  endif
else (no)
  :Continue monitoring;
endif
stop
@enduml
```

Листинг Б.8 – Диаграмма состояний

```
@startuml
[*] --> Detected
Detected --> Reviewed : Admin reviews
Reviewed --> Remediation : Critical violation
Reviewed --> Logged : Non-critical violation
Remediation --> Closed : Issue resolved
Logged --> Closed : Periodic review
Closed --> [*]
@enduml
```