ПРИМЕНЕНИЕ РЕЛЯЦИОННОЙ ИНТЕРАКТИВНОЙ ЛОГИКИ ПРИ ПОСТРОЕНИИ СИСТЕМЫ УВЕДОМЛЕНИЙ В РАМКАХ СОБЫТИЙНОЙ МОДЕЛИ ДАННЫХ

Захаров Д. В.
zakharov-danyl@mail.ru

В работе рассматривается подход к построению системы уведомлений в событийно-ориентированных информационных системах. Условия формирования и рассылки уведомлений задаются в формализованном виде с помощью систем логических уравнений и неравенств средствами реляционной интерактивной логики. Запись логических уравнений и неравенств производится с использованием атрибутов соответствующих информационных доменов, в рамках которых происходит функционирование системы.

Введение

В настоящее время в сфере разработки информационных систем и программных комплексов все большую популярность приобретает событийно-ориентированный подход [1]. Отличительная особенность таких систем состоит в том, что они позволяют не только накапливать данные и автоматизировать их обработку, но и обеспечивать своевременную реакцию при наступлении некоторых позитивных или негативных событий. Прежде всего это достигается за счет обработки входящего потока событий в режиме реального времени, при наступлении отдельных значимых событий система способна формировать корректирующее управляющее воздействие, либо уведомлять заинтересованных лиц.

Подобные системы находят свое применение в самых разнообразных сферах. В электронной коммерции они могут использоваться для формирования персонализированных предложений и аналитики действий посетителей онлайн-магазинов. В сфере информационной безопасности обработка входящего потока событий может быть использована для предотвращения атак и своевременного реагирования на инциденты. В промышленности такие системы применяются для контроля за состоянием технологических процессов.

Ключевой сущностью в описанных выше примерах является событие. Событие представляет собой некоторый именованный объект, который также может содержать ряд дополнительных параметров. Отметим, что событие может представлять собой как естественную транзакцию (например, факт оплаты или заказа в онлайн магазине), так и искусственную сущность, которая вводится в функционал системы исключительно в целях аналитики и мониторинга. В качестве примеров событий можно привести переход покупателя в карточку товара на сайте, изменение температуры рабочего оборудования или обращение к сетевому порту роутера.

Примерами дополнительных параметров здесь могут являться название товара, количество градусов, IP-адрес, с которого происходит обращение. Как правило, одним из ключевых параметров события также является временная метка, которая позволяет упорядочивать события и соотносить их очередность. На рисунке 1 представлена обобщенная структура события.

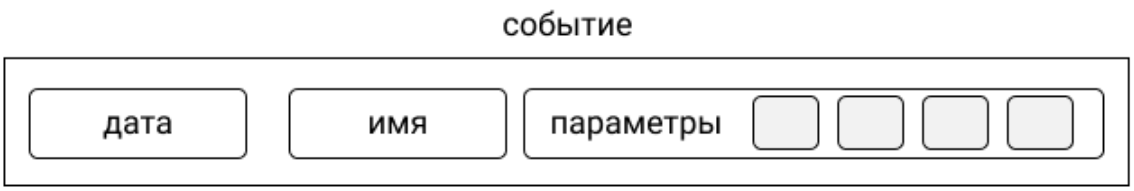

Рис. 1 Общая структура события

Реагирование на входящие события подразумевает настройку обработчиков, задача которых состоит в рассылке информационных сообщений, либо в формировании управляющих воздействий. Процесс конфигурации обработчика в данном случае заключается в записи условий, задающих логику работы обработчика: на какие события и каким образом он должен реагировать. Логика работы обработчика может закладываться непосредственно в коде программной системы, либо фиксироваться в соответствующих конфигурационных файлах.

Очевидно, что в ходе жизненного цикла системы могут появляться новые события, меняться их параметры. Под влиянием различных факторов изменения может претерпевать и общая бизнес-логика рассматриваемой системы. В этом случае для корректной работы системы реагирования может потребоваться перенастройка событийных обработчиков, что в свою очередь влечет необходимость дополнительной работы технических специалистов, а также изменение исходного кода системы. Все это приводит к дополнительным экономическим и ресурсным издержкам.

Вышеперечисленные факты обуславливают необходимость разработки методики, упрощающей настройку и конфигурирование системы реагирования. В идеальном случае перенастройка системы должна производиться без изменения исходного кода и быть доступна для специалистов, не имеющих отдельных компетенций в области разработки программного обеспечения.

В рамках данной работы предлагается формализованный способ записи условий реагирования с помощью систем логических уравнений и неравенств. Форма и способ записи систем логических уравнений определяются предметом и методом реляционной интерактивной логики [2].

В следующем пункте рассматривается общий подход к формированию условий рассылки уведомлений по входящим событиям на примере анализа поведения пользователей некоторой программной системы.

В последнем пункте приводится пример архитектуры информационной системы, а также описание общих принципов ее работы.

1. Формирование условий рассылки уведомлений с помощью РИЛ

Рассмотрим пример записи логического выражения, задающего условие на рассылку удерживающего маркетингового сообщения для нового пользователя программного продукта.

Предположим, что после регистрации пользовательский сценарий предполагает загрузку фотографий. В случае успешной реализации сценария в хранилище событий должно поступить событие «user_upload_photo», данное событие должно поступить через некоторое время после события «registration», которое говорит о регистрации нового пользователя в продукте. Негативный сценарий представляет собой обратную ситуацию, когда после регистрации в приложении пользователю не удалось загрузить фотографии. В этом случае необходимо направить такому пользователю маркетинговое push-уведомление, а также сообщить о наличии такого пользователя в службу технической поддержки. Запишем логическое уравнение, задающее описание негативного сценария.

([user_events].[event] = «registration»).COUNT > 0 AND ([user_events].[event] = « user_upload_photo »).COUNT = 0 EQV TRUE

Данное логическое условие описывает набор пользователей, которые совершили регистрацию, но не загрузили ни одной фотографии. Получив идентификаторы пользователей, соответствующих данному условию, мы можем выполнить рассылку сообщений. Рассмотрим подробнее данное логическое уравнение.

Для записи данного уравнения был использован реляционный терм [user_events].[event], первая часть которого указывает на информационный домен пользовательских событий, а вторая – на поле с названием события. Для отбора конкретных событий были использованы строковые константы, соответствующие названиям событий. Как было сказано в предыдущем пункте, при записи логических уравнений в нотации РИЛ мы можем также применять агрегирующие функции, поддерживаемые диалектом SQL. В данном случае в качестве маркера наличия события была использована агрегирующая функция COUNT, которая возвращает количество событий. Для записи ограничений на количество значений были использованы целочисленные константы, а также алгебраические операторы сравнения. Правая часть уравнения представляет собой логическую константу TRUE.

Использование логических операторов конъюнкции и дизъюнкции позволяет описывать комплексные сценарии, которые состоят из нескольких событий, а также учитывать дополнительные условия.

В данном конкретном случае негативный сценарий представляет собой факт наличия события регистрации и отсутствия события загрузки фотографии. РИЛ обеспечивает средства для гибкой записи как простых условий, состоящих из одного события, так и комплексных сценариев.

Очевидно, что рассмотренный подход подразумевает наличие некоторого программного приложения, позволяющего автоматизировать конфигурацию логических условий, их проверку, а также интеграцию со сторонними системами, обеспечивающими рассылку уведомлений по различным каналам.

Пример архитектуры и описание принципов работы такого приложения будет описан ниже.

2. Проект программной реализации

Рис. 2. Укрупненная схема программного приложения конфигурации СЛУН и настройки рассылок уведомлений

На рисунке 2 представлена общая укрупненная схема приложения.

Приложение состоит из следующих частей:

- Конфигуратор СЛУН – модуль обеспечивает формирование СЛУН при помощи графического интерфейса с использованием набора атрибутов информационного домена, после записи сформированная СЛУН сохраняется во внутреннем вспомогательном хранилище.
- Исполнитель запросов – модуль загружает сохраненную СЛУН, формирует на ее основе запрос к нужному источнику данных и возвращает результат запроса.
- Источники данных – внешние источники данных, содержащие информацию о событиях.
- Диспетчер рассылок – отвечает за рассылку сообщений по данным от исполнителя запросов. Модуль инкапсулирует в себе хранилище шаблонов сообщений, а также набор интеграций со сторонними сервисами (почтовые клиенты, мессенджеры, менеджеры push-уведомлений).
- Диспетчер задач – модуль аналогичен диспетчеру рассылок, но отвечает за выполнение каких-либо дополнительных обработчиков и задач, которые должны быть запущены после срабатывания триггера на событие. Также модуль отвечает за интеграцию со сторонними программными системами и сервисами.

Данная архитектура имеет модульный характер, что дает ряд преимуществ в ее поддержке и расширении. Каждый модуль отвечает за отдельный набор функциональности сервиса и связан с другими модулями за счет внешних абстрактных интерфейсов. При появлении новых источников данных или сервисов рассылок, модульная архитектура позволяет легко включить их в состав системы без глобальных изменений в логике работы.

Заключение

В рамках работы рассмотрен подход к записи условий формирования информационных уведомлений с помощью реляционной интерактивной логики. Отправка уведомления происходит при реализации позитивного либо негативного сценария, каждый из которых представляет собой набор из одного или нескольких событий. Реляционная интерактивная логика позволяет производить формализованное описание сценария в виде системы логических уравнений и неравенств.

Подобный подход позволяет конфигурировать необходимые сценарии для рассылки уведомлений без изменений в коде и логике работы системы. Конфигурация сценария производится с использованием полей и атрибутов из определенного информационного домена.

Применимость предложенной методики продемонстрирована на примере записи условий для отправки push-уведомлений пользователям программного приложения. Гибкость и высокая степень абстракции позволяет применить описанную методику и в рамках других предметных областей [4, 5].

Для автоматизации записи и выполнения сценариев рассылок в рамках данной работы предложена модульная расширяемая архитектура программного приложения, обеспечивающего запись СЛУН, ее выполнение и проверку, а также интеграцию со сторонними сервисами рассылок.

Использование современных интерфейсных средств позволяет привлекать к настройке сценариев не только технических специалистов, но также и экспертов отдельных информационных доменов. Однако несмотря на это, существенным ограничением для использования данной методики является требование понимания основ классической логики и базовых логических операторов, а также принципов агрегации данных, что несомненно может послужить препятствием для отдельных специалистов.

На текущий момент идет подготовка к разработке действующего прототипа программной системы на основе предложенной архитектуры.

Благодарности

Работа выполнена при поддержке Иркутского государственного университета в рамках выполнения работ по гранту для поддержки аспирантов и молодых сотрудников ИГУ № 091-20-301.

Список использованных источников

1. Mironov V., Gusarenko A., Yusupova N. Situation-Oriented databases: document management on the base of embedded dynamic model // Современные информационные технологии и ИТ-образование. 2016. №3-1.
2. Курганский В. И. Реляционная интерактивная логика. От логических уравнений и неравенств к ответам на мудрёные вопросы / В. И. Курганский // Lambert Academic Publishing, 2014. – 124 с.
3. Codd E. F. Relational completeness of data base sublanguages. IBM Research Laboratory, San Jose, California. KO 987 (#170041), March 6, 1972, Computer Sciences.
4. Захаров Д. В. Формализация некоторых эмпирических методов решения логических задач // Материалы Всероссийской научно-технической конференции студентов, аспирантов и молодых ученых. – Томск: В-Спектр, 2015: в 5 частях. – Ч. 4. – С. 49-51.
5. Захаров Д. В., Кузьмин О. В., Курганский В. И., Хоменко А. П. Системный анализ и контроль корректности бухгалтерских данных на основе реляционной интерактивной логики. // Журнал

«Современные технологии. Системный анализ. Моделирование». 2017. №2 (54).