

УДК 004.043: 004.91

DOI:10.38028/ESI.2022.28.4.014

Модельно-ориентированный подход к формированию оперативных печатных отчетов

Коробко Анна Владимировна¹, Шефтель Кирилл Сергеевич²

¹Сибирский государственный университет науки и технологий имени академика М.Ф. Решетнева,

²Институт вычислительного моделирования СО РАН,
Россия, Красноярск, *gglhroom@gmail.com*

Аннотация. Настоящее исследование посвящено разработке методических и алгоритмических средств генерации оперативных печатных отчетных форм на базе модельно-ориентированного подхода. В работе показана актуальность современного подхода к разработке программного обеспечения без программирования и обосновано использование модельно-ориентированной архитектуры в качестве технологической основы «по code» платформ. Приведено описание мета-метамоделей «по code» платформы сбора данных, обеспечивающей динамическую интерпретацию метамоделей и генерацию форм сбора данных. Предложено развитие существующей мета-метамоделей за счет добавления сущностей «шаблон» и «процедура», позволяющее оперативно формировать печатные отчетные формы с динамическим содержанием. Обоснован выбор шаблонизатора Jinja2 для программной реализации модуля генерации отчетов и представлена диаграмма последовательности модуля, показывающая взаимодействие интерфейсной части платформы с шаблонизатором и метаданными.

Ключевые слова: модельно-ориентированная разработка, zero code, платформа сбора данных, генерация печатных отчетов, шаблонизатор, Jinja2

Цитирование: Коробко А.В. Модельно-ориентированный подход к формированию оперативных печатных отчетов / А.В. Коробко, К.С. Шефтель // Информационные и математические технологии в науке и управлении. – 2022. – № 4(28). – С. 181-189. – DOI:10.38028/ESI.2022.28.4.014.

Ведение. В условиях обилия информационных потоков, всевозрастающих вычислительных мощностей и повышения требований к обоснованности принимаемых решений сохраняют свою актуальность печатные отчетные формы, особенно в критически важных областях деятельности человека. Несмотря на повсеместное внедрение оперативных информационных панелей и дашбордов, обновляемых автоматически в момент поступления новых данных, важной частью тактического и стратегического управления являются регламентированные срочные отчетные формы. Печатные отчеты позволяют «фиксировать» состояние отслеживаемой ситуации на момент формирования отчета, предоставлять всем заинтересованным лицам одну и ту же информацию, не беспокоясь о возможном изменении информации и манипуляции с данными.

Несмотря на кажущуюся статичность отчетов, чтобы оставаться актуальными, они должны изменяться. На примере оперативной сводки, формируемой Центром обеспечения реализации полномочий в областях гражданской обороны, чрезвычайных ситуаций Красноярского края (Центра ГО и ЧС) видно, что отчет претерпевает постоянные изменения. В зависимости от сезона меняется перечень контролируемых обстановок: подтопления и лесные пожары характерны для весенне-летнего периода, а зимой необходим мониторинг ледовых переправ. С окончанием отчетного года в некоторые разделы отчета добавляются новые столбцы для сопоставления показателей с аналогичными данными в предшествующие отчетные периоды. Ввиду развития средств инструментального контроля добавляются новые разделы, например, «космический мониторинг термоточек». С точки зрения программных средств формирования оперативного отчета все эти изменения требуют непосредственного участия человека в адаптации и настройке новой версии отчета. В таких случаях всегда было только два решения: пользоваться недорогим типовым программным обеспечением и мириться с невозможностью внести изменения, или приобретать многофункциональную дорогую систему и содержать в штате программиста, готового вносить правки, как только потре-

буется. Второй путь приведет нас к желанной цели, но с дополнительными финансовыми и временными издержками. Современное решение создавшейся проблемы – это low code/no code системы (LC/NC) [1], которые позволяют вносить изменения в клиентское приложение (системы) без программирования совсем или с незначительным объемом программного кода. Суть современной концепции «no code» («zero code») заключается в тезисе: «если участие человека необходимо, то пусть это будет аналитик, а не программист».

Технологической основой разработки «без программирования» является модельно-ориентированный подход, который позволяет создавать «zero code» – системы за счет вынесения бизнес-логики из программного кода в метаданные и «вшивания» алгоритмов построения прикладных систем под управлением этих метаданных [2]. Структура метаданных задает потенциальную функциональность системы, а данные, хранящиеся в этой структуре, определяют тематику системы, ее объем и информационное наполнение.

Настоящее исследование посвящено разработке методических и алгоритмических средств поддержки создания и редактирования печатных отчетных форм на базе модельно-ориентированного подхода. В работе показана актуальность современного подхода к разработке программного обеспечения без программирования и обосновано использование модельно-ориентированной архитектуры в качестве технологической основы «no code» платформ. Приведено описание мета-метамодели «no code» платформы сбора данных, обеспечивающей динамическую интерпретацию метамодели и генерацию форм сбора данных. Предложено развитие существующей мета-метамодели за счет добавления сущностей «шаблон» и «процедура», позволяющее оперативно формировать печатные отчетные формы с динамическим содержимым. Обоснован выбор шаблонизатора Jinja2 для программной реализации модуля генерации отчетов и представлена диаграмма последовательности модуля, показывающая взаимодействие интерфейсной части платформы с шаблонизатором и метаданными. Предложенные методические и алгоритмические средства апробированы на данных Агентства по гражданской обороне, чрезвычайным ситуациям и пожарной безопасности Красноярского края в рамках задачи формирования печатной формы оперативной сводки.

1. Модельно-ориентированный подход. В современном мире программисты постоянно работают над повышением уровня абстракции своих программ. Многие поняли, что повышение уровня абстракции предметной области на один порядок дает расширение сферы применения продукта в десятки раз и возможность занять многие смежные ниши, порой увеличивая количество клиентов в сотни и тысячи раз. Модельно-ориентированный подход является результатом развития методик и подходов абстрагирования от предметных областей в сторону концепции функционирования системы. Вместо требования от разработчиков использовать язык программирования для спецификации того, как система должна работать, подход позволяет использовать модели для декларативного описания того, какая функциональность необходима и какой должна быть архитектура системы. Движение в сторону высокоуровневой спецификации содержит потенциал для значительного упрощения разработки программного обеспечения. Такие проблемы, как распределение объектов, поиск методов, перехватывание исключений и пр., более не являются актуальными задачами для большинства разработчиков и решаются автоматически на этапе компиляции/интерпретации программы. Цель модельно-ориентированного подхода в достижении такого уровня абстрагирования, который позволит автоматизировать решение проблем устойчивости, совместимости и развертывания, в данный момент решаемые вручную с привлечением высококвалифицированных специалистов, оставляя лишь задачу наполнения системы. Классическая разработка на основе модельно-ориентированного подхода подразумевала построение метамodelей отдельно от кода и служила скорее ориентиром, чем функциональной частью системы. С развитием новой парадигмы гибкой разработки и концепции снижения требований к квалифи-

кации пользователей возникла идея динамической интерпретации метамodelей и разработки «no-code» платформ, которые позволяют без программирования «собирать» тематические приложения с заданной функциональностью, но с произвольной специализацией и информационным наполнением. Теперь метамodelи и модели формируются с помощью визуальных конструкторов на базе платформ и определяют бизнес-логику, функциональность и внешний вид будущего приложения. Гибкость и эффективность «no-code» платформ заключается в алгоритмах генерации схемы базы данных, пользовательского и программного интерфейса, аналитических и печатных отчетных форм. Охрана коммерческой тайны проприетарного программного обеспечения в части разработки механизмов конкретной реализации подхода сохраняет актуальность научных исследований, посвященных решению той же задачи, но с исследовательским подходом.

Технология модельно-ориентированной разработки (Model Driven Development – MDD) основывается на построении абстрактной мета-метамodelи, управляющих и прикладных моделей и задании способов их трансформации в поддерживаемые технологии программирования [3, 4, 5]. «Разработка» модельно-ориентированных систем в рамках подхода MDD заключается в построении моделей абстрактного уровня и автоматической генерации моделей прикладного уровня. Проектирование моделей абстрактного уровня включает процесс построения мета-метамodelи – модели языка моделирования – и процесс построения метамodelи различных специфичных областей приложения. Проектирование прикладных моделей состоит из процесса построения моделей прикладного уровня и этапа формирования экземпляров концептов, определенных на абстрактном уровне. Мета-метамodelь в информатике – модель, описывающая другую модель; транзитивное отношение между двумя моделям. Например, если модель M1 описывает язык L0, в котором формулируется модель M0, то M1 является метамodelью M0; если же модель M2 описывает язык L1, в котором была сформулирована модель M1, то M2 – это метамodelь M1, а M2 является тогда для M0 мета-метамodelью.

С целью создания «no-code» платформы для создания систем сбора данных на основе модельно-ориентированного подхода была предложена авторская реализация классического процесса MDD [6]. Во-первых, объединены процессы формирования управляющей и прикладной моделей. Объединенный процесс предполагает, что управляющая модель системы формируется пользователем с помощью интерфейса самой системы. При этом автоматически изменяется структура базы данных и сохраняется формальное описание моделей. Во-вторых, добавлен процесс автоматической трансформации прикладных моделей в пользовательский интерфейс модельно-ориентированной системы – динамическая интерпретация метамodelи. Благодаря этому интерфейс системы в части ввода данных перестраивается автоматически, в соответствии с изменениями, внесенными пользователем в управляющую модель. В основе авторской реализации, как и в основе классического подхода, лежит концептуальная мета-метамodelь, определяющая основные классы сущностей и связи между ними для построения управляющих моделей.

Для решения задачи разработки типовых модельно-ориентированных систем консолидации данных предложена мета-метамodelь сбора данных [7], которая является спецификацией для создания управляющих и прикладных моделей проектируемых систем, обеспечивает динамическую интерпретацию метамodelи и генерацию форм сбора данных. Мета-метамodelь содержит три класса объектов: «Объект», «Атрибут» и «Группа». Объекты класса «Атрибут» описываются свойствами: наименование атрибута, тип атрибута, флаг темпоральности атрибута. В рамках мета-метамodelи задан ряд отношений, определяющих связи между экземплярами классов модели.

Применяемый в платформе сбора данных модельно-ориентированный подход можно применить и к генерации отчетов. Если представить информацию о структуре и наполнению

отчетов как метаданные, то программный код, отвечающий за формирование отчетов на основе метамодели, должен включать унифицированный модуль генерации отчета.

2. Проектирование модуля генерации отчётов. Применение модельно-ориентированного подхода при создании модуля подразумевает модификацию мета-метамодели для полноценной интеграции новой функциональности в систему. Модификация заключается во внесении в структуру метаданных сущностей, без которых процесс генерации отчетов будет невозможен. Отчет принято разделять на статическую и динамическую составляющие. Статическая часть – шаблон отчета – определяет индивидуальную структуру отчета в соответствии с назначением документа: заголовок и параметры генерации, блоки, составляющие отчет, и их наполнение. Динамическая часть формируется в момент генерации отчета по заданному шаблону. Предлагается ввести в мета-метамодель такие сущности, как «шаблон» и «процедура», отвечающие за статическую и динамическую составляющие отчета соответственно. Процедуры позволяют перенести сбор и группировку данных из программного кода в базу данных. При необходимости изменить набор данных, добавить новые данные в отчет или удалить их, благодаря процедурам, больше не нужно редактировать программный код, а достаточно указать новую процедуру в шаблоне или её оттуда удалить. Модифицированная мета-метамодель представлена на рисунке 1.

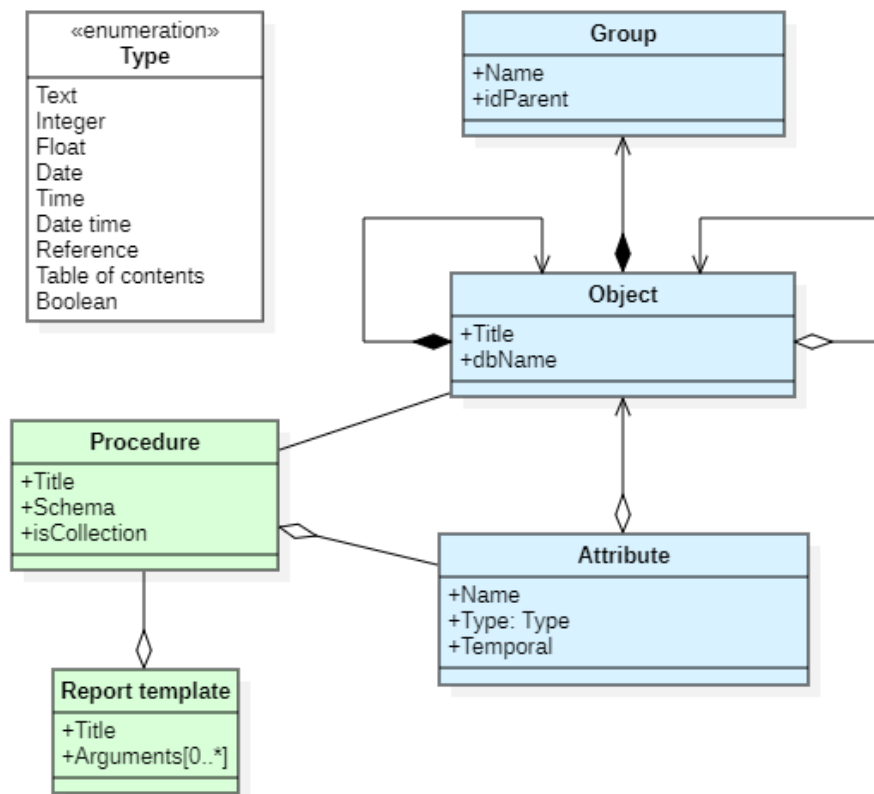


Рис. 1. Модифицированная мета-метамодель модельно-ориентированной платформы

С целью реализации механизма динамической генерации печатных отчетных форм модель дополнена двумя классами: «Procedure» и «Report template». Новые классы описывают организацию доступа к консолидированным данным в виде хранимых процедур и их связь с шаблоном печатной формы, использующим эти данные. Класс «Report template» в настоящей модификации содержит название шаблона, по которому программный модуль

найдет выбранный шаблон на сервере. Класс «Procedure» содержит имя процедуры для получения оперативных данных, схему в базе данных, разграничивающую процедуры из разных разделов системы, и логическое поле «Коллекция». Поле «Коллекция» принимает

значения: «Истина» или «Ложь». Значения определяют, какого рода данные будут получены и как их следует обрабатывать. Значение «Истина» означает, что результатом выполнения хранимой процедуры будет множество записей, которые могут быть представлены в отчете как таблица или могут требовать дополнительной обработки. Значение «Ложь» соответствует ситуации, когда процедура возвращает одну запись или скалярное выражение. Такие данные могут быть представлены в отчете «как есть» или могут использоваться в обработке данных, полученных от процедур со значением параметра «Истина».

3. Программная реализация модуля генерации отчетов. Самый распространённый подход к генерации отчетных документов до недавнего времени заключался в сохранении шаблонов в форматах doc(x) и xls(x) с разметкой, интерпретируемой специализированными программными средствами [8, 9]. Однако, с утратой доверия к форматам корпорации Microsoft актуальной стала задача поиска более демократичного формата для представления шаблонов оперативных печатных форм. В рамках web-приложения таким форматом является HTML (HyperText Markup Language). Мы можем использовать богатые возможности языка гипертекстовой разметки по конструированию визуальных элементов и применять каскадные таблицы стилей (Cascading Style Sheets, CSS) для управления оформлением и даже анимацией страницы. Все эти преимущества могут быть использованы при создании шаблонов и генерации печатных отчетов. С использованием HTML как шаблона и формата представления конечного отчета мы можем как отображать сгенерированный отчет на странице web-приложения, так и экспортировать его в другие форматы, например, PDF или XML.

В разработке пользовательского интерфейса web-приложений задача наполнения шаблона фактическими данными решается с помощью шаблонизатора (Template Engine). Шаблонизатор – программное обеспечение, позволяющее использовать шаблоны для генерации конечных документов с помощью декларативного языка разметки [10]. В реализации шаблонизатора воплощается идея разделения бизнес-логики и представления, формы и содержания при создании документов [11]. Шаблонизаторы призваны обеспечить наполнение повторяющихся элементов web-приложения динамическим контентом на HTML странице, например, для генерации карточек товаров в Интернет-магазине.

Шаблонные движки принимают маркированные строки и выдают на выходе визуализированные строки со значениями вместо маркеров. Они позволяют разработчикам генерировать желаемые типы контента, используя при этом некоторые конструкции программирования, такие как условия и циклы, для управления результатом вывода.

Для программной реализации модуля генерации оперативных печатных отчетов выбран один из наиболее популярных шаблонизаторов на языке Python – Jinja2 [12], который позволяет использовать тэги, фильтры, тесты и глобальные переменные, а также обеспечивает исполнение шаблонов в тестовом режиме в «песочнице», тем самым гарантируя безопасность конечной системы. Jinja2 широко используется вместе с фреймворком Flask и считается хорошим шаблонизатором в случаях, где предполагается постоянная работа над шаблонами, их модификация.

Разработанное дополнение к мета-метамодели может служить концептуальной моделью для описания отчетов системы, которые хранятся в метаданных системы (metadata). Шаблоны (template), хранимые в метаданных, представляют модель отчета, а разрабатываемый модуль генерации осуществляет функции интерпретации шаблона и встроенного динамического описания. На рисунке 2 представлена диаграмма последовательности генерации печатного отчета по шаблону.

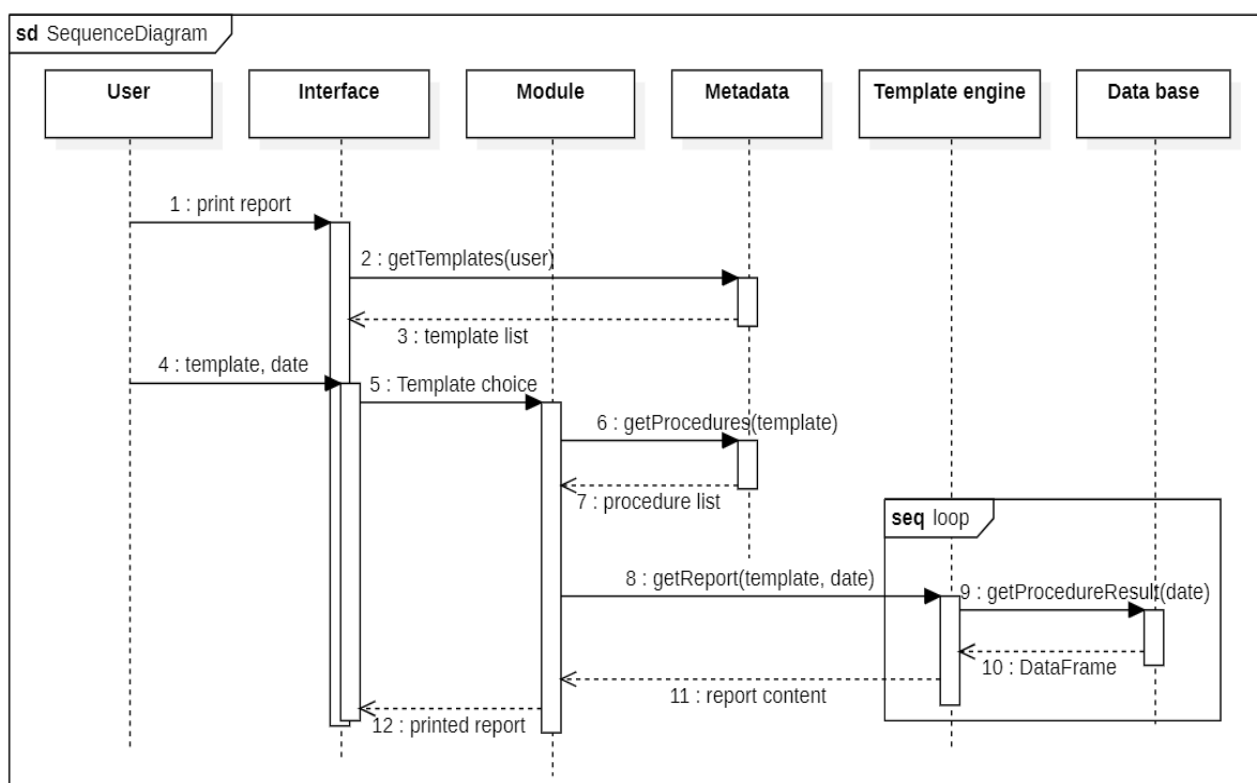


Рис. 2. Диаграмма последовательности генерации оперативного печатного отчета

Диаграмма последовательности фокусируется на временном аспекте взаимодействия и имеет два измерения. Первое измерение, горизонталь, указывает на порядок вовлечения экземпляров сущностей (прямоугольники) во взаимодействие. Второе измерение, вертикаль, указывает на порядок обмена сообщениями (нумерованные стрелки) [22].

После открытия окна генерации отчета (print report) интерфейсная часть приложения запрашивает через API (application programming interface) серверной части метаданные о доступных пользователю шаблонах отчетов (getTemplates(user)). Пользователь выбирает шаблон из представленного списка и задает параметры генерации отчета (template, date). Модуль генерации начинает процедуру формирования отчета с помощью шаблонизатора. Для этого модуль запрашивает метаданные о процедурах, используемых в выбранном шаблоне (getProcedures(template)), и передает управление шаблонизатору Jinja2, который в цикле выполняет запросы на получения фактических данных из базы данных (getReport(template, date)) и генерирует динамический контент отчета в соответствии с разметкой шаблона. Результат генерации печатного отчета в формате HTML предоставляется пользователю.

Модуль генерации отчетов воплощает модельно-ориентированный подход и демонстрирует возможность использования метаданных для обеспечения большей гибкости в построении оперативных печатных отчетов. Шаблон отчета может быть актуализирован пользователем в конструкторе шаблонов и сразу использоваться как основа для построения печатного отчета новой версии.

Заключение. Современные тренды в разработке программного обеспечения и развитие технологий обработки и представления данных дают возможность по-новому взглянуть на многократно решенную задачу формирования оперативных печатных отчетов. Применение модельно-ориентированного подхода диктует необходимость включения сущностей, необходимых для формирования отчетов в мета-метамодель. Использование нестандартных форматов представления шаблонов требует подбора программной библиотеки, легко интегрируемой в существующую систему. Разработка специализированного модуля генерации отчетов обеспечивает взаимодействие всех компонентов: пользовательского интерфейса, программ-

ного интерфейса, метаданных и шаблонизатора. Настоящая работа представляет результаты разработки методических и алгоритмических средств генерации оперативных печатных отчетных форм на базе модельно-ориентированного подхода.

Благодарности. Работа выполнена в рамках проекта государственного задания ФИЦ КНЦ СО РАН программы фундаментальных исследований РФ (рег. № 0287-2021-0003).

Список источников

1. Cypher A. et al. No code required: giving users tools to transform the web. – Morgan Kaufmann, 2010.
2. Cabot J. Positioning of the low-code movement within the field of model-driven engineering. Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings, 2020, pp. 1-3.
3. Atkinson C., Kuhne T. Model-driven development: a metamodeling foundation. IEEE software, 2003, vol. 20, no. 5, pp. 36-41.
4. Pastor O. et al. Model-driven development. Informatik-Spektrum, 2008, vol. 31, no. 5, pp. 394-407.
5. Грищенко М.А. Применение модельно-управляемого подхода для создания производственных экспертных систем и баз знаний / М.А. Грищенко, Н.О. Дородных, О.А. Николайчук, А.Ю. Юрин // Искусственный интеллект и принятие решений, 2016. – №. 2. – С. 16-29.
6. Свидетельство о государственной регистрации программы для ЭВМ № 2018614159 Российская Федерация. Веб-платформа для построения модельно-ориентированных систем сбора данных: № 2018611794: заявл. 22.02.2018 : опубл. 02.04.2018 / Л. Ф. Ноженкова, А. В. Коробко, А. А. Коробко ; заявитель Федеральное государственное бюджетное научное учреждение «Федеральный исследовательский центр «Красноярский научный центр Сибирского отделения Российской академии наук».
7. Korobko A. A., Korobko A. V. A software platform for constructing model-driven systems for primary data consolidation //Journal of Physics: Conference Series, IOP Publishing, 2020, vol. 1679, no. 3, pp. 032056.
8. Ланин В.В. Управление документами в динамически адаптируемых системах, основанных на метамоделировании / В.В. Ланин, Л.Н. Лядова // Труды Конгресса по интеллектуальным системам и информационным технологиям «AIS-IT, 2010. – Т. 10. – С. 510-518.
9. Пенькова Т.Г. Модели и методы оперативного формирования документов / Т.Г. Пенькова // Вычислительные технологии, 2009. –Т. 14. – №2. – С. 98-109.
10. Panati M. Using Template Engines. Beginning KeystoneJS, Apress, Berkeley, CA, 2016, pp. 123-132.
11. Parr T. J. Enforcing strict model-view separation in template engines. Proceedings of the 13th international conference on World Wide Web, 2004, pp. 224-233.
12. Ronacher A. Jinja2 documentation. Welcome to Jinja2—Jinja2 Documentation (2.8-dev), 2008.

Коробко Анна Владимировна. К.т.н., доцент кафедры Информационных управляющих систем Института информатики и телекоммуникаций СибГУ им. М.Ф. Решетнева. Основные направления исследований включают методы и технологии интеллектуального и оперативного анализа данных, разработку и поддержку программного обеспечения, инициацию и выполнение научных исследований и наукоёмких инновационных проектов (StartUp). AuthorID: 615780, SPIN: 7756-2840, ORCID: 0000-0001-5337-3247. gglhroom@gmail.com, 660036, г. Красноярск, Академгородок, д. 50, стр. 44.

Шефтель Кирилл Сергеевич. Бакалавр по направлению «Программная инженерия». Основные направления исследований включают модельно-ориентированную разработку и поддержку программного обеспечения. sheftel.ks@gmail.com, 660036, г. Красноярск, Академгородок, д. 50, стр. 44.

Model-driven approach to generating operational printed reports

Anna V. Korobko¹, Kirill S. Sheftel²

¹Reshetnev Siberian State University of Science and Technology,

²Institute of Computational Modelling of the Siberian Branch of the Russian Academy of Sciences, Russia, Krasnoyarsk, *gglhroom@gmail.com*

Abstract. This work is focused on the development of methodological and algorithmic means of generating operational printed report forms based on the model-oriented approach. The paper shows the relevance of modern approach to software development without programming and justifies the use of model-driven architecture as the technological basis of "no code" platforms. Meta-metamodel description of "no code" data consolidation platform, that provides dynamic interpretation of the metamodel and data consolidation forms generation, is presented. Suggested the development of existing meta-metamodel by adding the entities "template" and "procedure", allowing for the efficient generation of printed report forms with dynamic content. The choice of Jinja2 templating engine for software implementation of report generation module is justified and a diagram of module sequence showing the interaction of the interface part of the platform with templating engine and metadata is presented.

Keywords: model-driven development, zero code, data consolidation platform, report generation, template engine, Jinja2

Acknowledgements: The work was carried out within the framework of the project of the state order of the Federal Research Center of the Siberian Branch of the Russian Academy of Sciences for the program of fundamental research of the Russian Federation (registration No. 0287-2021-0003).

References

1. Cypher A. et al. No code required: giving users tools to transform the web. – Morgan Kaufmann, 2010.
2. Cabot J. Positioning of the low-code movement within the field of model-driven engineering. Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings, 2020, pp. 1-3.
3. Atkinson C., Kuhne T. Model-driven development: a metamodeling foundation. IEEE software, 2003, vol. 20, no. 5, pp. 36-41.
4. Pastor O. et al. Model-driven development. Informatik-Spektrum, 2008, vol. 31, no. 5, pp. 394-407.
5. Grishchenko M. A., Dorodnyh N. O., Nikolaychuk O. A., Yurin A. U. Primenenie model'no-upravlyaemogo podhoda dlya sozdaniya produkcionnyh ekspertnyh sistem i baz znaniy [Applying a model-driven approach to develop productive expert systems and knowledge bases]. Iskusstvennyj intellekt i prinyatie reshenij [Artificial intelligence and decision-making], 2016, no. 2, pp. 16-29.
6. Svidetel'stvo o gosudarstvennoj registracii programmy dlya EVM no. 2018614159 Rossijskaya Federaciya [State Registration Certificate for Computer Programme no. 2018614159 Russian Federation]. Veb-platforma dlya postroeniya model'no-orientirovannyh sistem sbora dannyh: no. 2018611794 : zayavl. 22.02.2018: opubl. 02.04.2018 / L. F. Nozhenkova, A. V. Korobko, A. A. Korobko; zayavitel' Federal'noe gosudarstvennoe byudzhethoe nauchnoe uchrezhdenie «Federal'nyj issledovatel'skij centr «Krasnoyarskij nauchnyj centr Sibirskogo otdeleniya Rossijskoj akademii nauk» [Web-platform for construction of model-oriented data consolidation systems: No. 2018611794: application. 22.02.2018: publ. 02.04.2018 / L.F. Nozhenkova, A.V. Korobko, A.A. Korobko; applicant Federal State Budgetary Scientific Institution "Federal Research Centre "Krasnoyarsk Scientific Centre of Siberian Branch of Russian Academy of Sciences"].
7. Korobko A. A., Korobko A. V. A software platform for constructing model-driven systems for primary data consolidation. Journal of Physics: Conference Series. IOP Publishing, 2020, vol. 1679, no. 3, pp. 032056.
8. Lanin V. V., Lyadova L. N. Upravlenie dokumentami v dinamicheski adaptiruemyh sistemah, osnovannyh na metamodelirovanii [Document management in dynamically adaptable systems based on metamodeling]. Trudy Kongressa po intellektual'nym sistemam i informacionnym tekhnologiyam «AIS-IT» [Proceedings of the AIS-IT Congress on Intelligent Systems and Information Technology], 2010, part 10, pp. 510-518.
9. Penkova T.G. Modeli i metody operativnogo formirovaniya dokumentov [Models and methods for the rapid generation of documents]. Vychislitel'nye tekhnologii [Computational technologies], 2009, part 14, no. 2, pp. 98-109.
10. Panati M. Using Template Engines. Beginning KeystoneJS. Apress, Berkeley, CA, 2016, pp. 123-132.

11. Parr T. J. Enforcing strict model-view separation in template engines. Proceedings of the 13th international conference on World Wide Web, 2004, pp. 224-233.
12. Ronacher A. Jinja2 documentation. Welcome to Jinja2–Jinja2 Documentation (2.8-dev), 2008.

Anna Vladimirovna Korobko. *Candidate of Technical Sciences, Associate Professor of the Department of Information Control Systems of the Institute of Informatics and Telecommunications of the Siberian State University. M.F. Reshetnev. The main areas of research include methods and technologies for intellectual and operational data analysis, software development and support, initiation and implementation of scientific research and science-intensive innovative projects (StartUp). AuthorID: 615780, SPIN: 7756-2840, ORCID: 0000-0001-5337-3247. gllhroom@gmail.com, 660036, Krasnoyarsk, Akademgorodok, 50, building 44.*

Kirill Sergeevich Sheftel. *Bachelor in Software Engineering. The main areas of research include Model-Based Development and Software Support. sheftel.ks@gmail.com, 660036, Krasnoyarsk, Akadem-gorodok, 50, building 44.*
Статья поступила в редакцию 05.08.2022; одобрена после рецензирования 06.09.2022; принята к публикации 19.09.2022.

The article was submitted 08/05/2022; approved after reviewing 09/06/2022; accepted for publication 09/19/2022.