УДК 004.942
DOI 10.25205/1818-7900-2018-16-2-31-40

И. В. Брак [1], **Ю. И. Сазонова** [2, 3]

[1] *НИИ физиологии и фундаментальной медицины*
ул. Тимакова, *4*, *Новосибирск*, *630090*, *Россия*

[2] *Институт вычислительных технологий*
пр. Академика Лаврентьева, *6*, *Новосибирск*, *630090*, *Россия*

[3] *Новосибирский государственный университет*
ул. Пирогова, *1*, *Новосибирск*, *630090*, *Россия*

brack@physiol.ru, *i.sazonova@g.nsu.ru*

РАЗРАБОТКА СЕРВИСА ЗАДАНИЯ СЦЕНАРИЕВ ПРЕДЪЯВЛЕНИЯ СТИМУЛОВ С ИСПОЛЬЗОВАНИЕМ МОДЕЛЬНО-ОРИЕНТИРОВАННОГО ПОДХОДА

Современная физиология не может обойтись без методов количественного анализа данных. Необходимым условием для использования математической статистики, анализа сигналов и машинного обучения является наличие должным образом собранных, размеченных и подготовленных данных. С возможностью совместной обработки данных, собранных в разных условиях и в рамках разных протоколов экспериментов, появилась потребность в наличии структурированной метаинформации. В настоящее время существует множество программных систем, позволяющих создавать, редактировать и запускать сценарии представления стимулов. Их проблемой является сложность использования реализованного сценария как в рамках других систем, так и для аннотирования данных, полученных экспериментально. Целью работы является разработка сервиса, позволяющего задавать сценарии представления стимулов с помощью графического интерфейса с возможностью сохранять метаинформацию эксперимента в независимом от платформы формате и исполнять в закрытых системах. В предлагаемом решении используется модельно-ориентированный подход. В основе платформенно-независимой модели лежит открытый формат эксперимента PsychoPy. Для исполнения полученного сценария используется платформа Neurobs Presentation. С помощью преобразования общей модели сценария эксперимента в модель платформы и описания синтаксической структуры предметно-ориентированного языка Presentation автоматически формируется программный код. Реализация данного подхода может быть расширена для других систем представления стимулов.

Ключевые слова: модельно-ориентированный подход, кодогенерация, предметно-ориентированный язык, система предъявления стимулов.

Введение

Количественный анализ данных является важным методом исследований в современной инструментальной физиологии. Высокой прогностической и диагностической значимостью обладают данные биоэлектрической активности головного мозга. Анализ данных количественной электроэнцефалографии (кЭЭГ) является перспективным направлением для применения математической обработки. В кЭЭГ используются такие параметры, как амплитуда, мощность, спектр, когерентность внутри- и межполушарных взаимодействий и другие характеристики осцилляторной активности головного мозга [1]. Регистрация показателей может происходить в состоянии спокойного бодрствования с открытыми или закрытыми глазами, при выполнении функциональных проб или когнитивных заданий. При создании эксперимента необходимо учитывать такие аспекты, как содержание и структура сценария, взаимодействие и синхронизация с аппаратными компонентами, формат выходных данных.

Брак И. В., *Сазонова Ю. И.* Разработка сервиса задания сценариев предъявления стимулов с использованием модельно-ориентированного подхода // Вестн. НГУ. Серия: Информационные технологии. 2018. Т. 16, № 2. С. 31–40.

© И. В. Брак, Ю. И. Сазонова, 2018

В настоящее время существует ряд программных систем, позволяющих создавать, редактировать и воспроизводить сценарии предъявления стимулов (Cedrus – SuperLab [1], Millisecond Software – Inquisit [2], Mitsar – Psytask [3], Neurobs – Presentation [4], Nottingham University – PsychoPy [5], OkazoLab – EventIDE [6], Psychology Software Tools – E-Prime [7] и др.). Большинство систем платные и позволяют сохранять сценарий либо в виде программы на собственном языке создания сценариев, либо в закрытом формате. Системы, в которых способом создания сценария является написание программы, обладают широкими возможностями исполнения, но сложны для изучения и применения. Проблемой платных систем также является ограничение возможностей использования реализованного сценария как в рамках других систем, так и для аннотирования данных, полученных экспериментально.

Целью работы является разработка сервиса, позволяющего задавать сценарии предъявления стимулов с помощью графического интерфейса с возможностью сохранять метаинформацию о сценарии эксперимента в независимом от платформы формате и исполнять в системах с собственным языком описания сценариев.

В предлагаемом решении используется модельно-ориентированный подход. В основе платформенно-независимой модели лежит открытый формат эксперимента системы PsychoPy. Для исполнения полученного сценария была выбрана платформа Neurobs Presentation. С помощью преобразования платформенно-независимой модели в модель языка Presentation и описания его синтаксической структуры автоматически формируется программный код. Реализация данного подхода может быть расширена для исполнения разработанного сценария в других системах предъявления стимулов с собственным скриптовым языком.

Анализ систем предъявления стимулов

Рассмотрим особенности создания, редактирования и сохранения данных экспериментов для систем задания сценариев предъявления стимулов (табл. 1).

Создание сценария эксперимента может происходить двумя способами: с помощью собственного «скриптового» языка системы (Presentation [8], Inquisit [9]) или с помощью графического интерфейса, функциональность которого иногда дополняют языком общего назначения или его расширением.

Язык Inquisit похож на язык разметки и является декларативным. Он представляет собой набор именованных элементов (стимулы, тестовые пробы, последовательности стимулов и т. п.) и выражений присваивания для их параметров. Кроме констант, списков и ссылок на другие элементы, значениями параметров могут быть выражения (арифметические, присваивающие и условные). Таким образом, потенциальная сложность языка может заключаться в описаниях параметров.

В языке Presentation выделяются декларативная (Scenario Description Language, SDL) и процедурная (Presentation Control Language, PCL) части. Кроме того, в заголовке скрипта Presentation задаются настройки сценария (Header). Декларативная часть языка позволяет описать набор элементов эксперимента и их параметров. В процедурной части есть возможность задать порядок предъявления элементов SDL, используя общие конструкции: переменные, контейнеры, условия и циклы.

Язык для описания сценариев Psytask [10] включает в себя списки стимулов и проб, список предъявления проб и команд, обработку ответной реакции. Набор стимулов ограничен

[1] https://www.cedrus.com/superlab/
[2] https://www.millisecond.com
[3] http://www.mitsar-medical.com/eeg-software/qeeg-software/
[4] https://www.neurobs.com
[5] http://www.psychopy.org
[6] http://www.okazolab.com
[7] https://pstnet.com/products/e-prime/
[8] https://www.neurobs.com/presentation/docs/index_html
[9] https://millisecond.com/support/docs/
[10] http://www.mitsar-eeg.ru/download/manuals/Psytask_UM_RUS_v.1.50.pdf

Таблица 1

Обзор форматов, используемых основными системами предъявления стимулов

Система	Лицензия	Способ задания сценария	Формат	
			сценария	выходных данных
E-Prime	Комм.	Графический интерфейс E-Studio, дополнительная функциональность реализуется с помощью языка E-Basic (на основе Visual Basic)	***.es** – формат для хранения и редактирования в E-Studio; ***.ebs** – формат для исполнения в E-Run	***.edat**
Inquisit	Комм.	Редактор для языка Inquisit	***.iqx –** текстовый файл скрипта	***.tsv**
EventIDE	Комм.	Графический интерфейс (есть возможность использования XAML для графических элементов), дополнительная функциональность реализуется с помощью расширения языка C#	***.eve** – формат EventIDE	**Delimiter separated values**
Presentation	Комм.	Редактор языка Presentation и графический интерфейс для дополнительных настроек	***.sce** – текстовый файл скрипта	**Delimiter separated values**
PsychoPy	О.	Графический интерфейс PsychoPy Builder / редактор Python PsychoPy Coder	***.psyexp –** файл формата XML, соответствующий XSD эксперимента / ***.py** – файл программы на Python	**Delimiter separated values** / ***.psydat** (сериализованный Python объект)
Psytask	Пр.	Графический интерфейс / загрузка файла сценария	***.PRO** – текстовый файл скрипта	***.dbf** (запись в базу данных)
SuperLab	Комм.	Графический интерфейс	***.sl** – формат SuperLab	***.tsv**

Примечание. О системе SuperLab см.: https://www.cedrus.com/superlab/manual.htm/. Обозначения лицензии: Комм. – коммерческая, Пр. – проприетарная, О. – открытая.

Таблица 2

Особенности систем предъявления стимулов с графическим интерфейсом

Система	Графическое представление последовательности предъявления	«Полнота» графического интерфейса	Универсальность системы
E-Prime	+	–	+
EventIDE	+	–	+
PsychoPy	+	+	+
Psytask	–	+	–
SuperLab	–	+	+

несколькими форматами, порядок показа линейный, без возможности рандомизации, за счет чего язык понятен и хорошо подходит для создания простых сценариев. Более универсальные предметно-ориентированные языки (Inquisit, Presentation) включают в себя понятия разных уровней: от уровня предметной области (функциональная проба) до особенностей реализации (цвет шрифта). Вместе с особенностями порядка предъявления стимулов (рандомизацией) и обработкой реакции исследуемого такие языки будут достаточно сложными для людей, не знакомых с программированием.

При создании сценариев с помощью графического интерфейса существующие системы используют формы для задания параметров. Кроме того, некоторые системы используют визуальное представление последовательности стимулов в виде потока работ (PsychoPy [11]), ориентированного графа (EventIDE [12]) или древовидной структуры (E-Prime [13]). Системы E-Prime и EventIDE расширяют функциональность графического интерфейса с помощью расширений для языков общего назначения Visual Basic и C#. Особенности графических систем предъявления стимулов показаны в табл. 2. Под «полнотой» понимается возможность задания любого реализуемого в системе сценария с помощью графического интерфейса (без использования языка программирования). Универсальными названы системы со встроенной рандомизацией и возможностью проектировать нелинейные сценарии**.**

Сценарий в платных системах сохраняется в закрытом формате и может исполняться только внутри системы. В случае, когда сценарий является скриптом, его можно редактировать как текст. Форматы выходных данных чаще всего представляют собой набор значений (delimiter separated values) и хорошо подходят только для анализа показателей в рамках одного исследования.

Исполнение реализованного в определенной системе сценария невозможно без ручного переноса информации в другую систему. Различные аспекты сценария (оформление стимулов, настройки последовательности и времени предъявления, аппаратные особенности, формат вывода и пр.) в большинстве случаев собраны вместе, затрудняя тем самым изучение и применение системы для создания и изменения сценариев. Кроме того, полезная для дальнейшей обработки данных метаинформация о сценарии эксперимента не может быть напрямую получена из закрытого формата эксперимента и из скрипта сценария.

Для решения задачи платформенно-независимой разработки сценариев с использованием возможностей существующих систем предъявления стимулов (графический интерфейс и функциональность) и обеспечением модульной интеграции между ними наиболее целесообразно применить модельно-ориентированный подход [2; 3].

Применение модельно-ориентированного подхода к задаче разработки сценариев

С «архитектурой, управляемой моделями» (Model Driven Architecture, MDA [14]), связывают стандарт MDA, разрабатываемый консорциумом Object Management Group [15] с 2000 г. Согласно методологии MDA модели являются главными элементами процесса разработки. Для конструирования программного приложения должна быть построена подробная, формально точная модель, из которой потом может быть автоматически получен исполняемый программный код. Под моделью понимается выборочное (ограниченное) представление некоторой системы, форма и содержание которого могут быть выражены с помощью набора понятий (концептов). Для описания модели могут быть использованы различные нотации и форматы. Метамодель определяет абстрактный синтаксис языка моделирования.

По стандарту «Метаобъектного средства» (Meta-Object Facility, MOF [16]) различают четыре уровня моделирования: M0–M3. Языком описания верхнего уровня часто является Unified Modeling Language (UML [17]).

[11] http://www.psychopy.org/builder/builder.html#builder
[12] http://www.okazolab.com
[13] https://support.pstnet.com/hc/en-us/categories/115000291167-E-Prime-3-x
[14] https://www.omg.org/mda/
[15] https://www.omg.org
[16] https://www.omg.org/spec/MOF

В процессе модельно-ориентированной разработки можно выделить следующие шаги [4]:

1) создание модели предметной области, независимой от платформы (Platform Independent Model, PIM);

2) создание модели платформы (Platform Specific Model, PSM), которая определяет специфику конкретной реализации;

3) преобразование PIM → PSM, с помощью которой с каждым формальным понятием модели предметной области сопоставляется его реализация;

4) генерирование необходимых артефактов.

К каждому из шагов 1–3 можно возвращаться, расширяя модель. При этом изменения уже существующих элементов повлекут за собой изменения на следующих стадиях.

В настоящее время существуют инструменты для применения стандарта OMG MDA [18]. В работе использовалась свободно распространяемая система Eclipse Modeling Framework (EMF) [5]. Проект EMF представляет собой платформу для моделирования с возможностью генерирования программного кода для создания инструментов и приложений на основе структурированной модели данных. В качестве модели верхнего уровня в EMF используется язык Ecore [6], который похож на UML, но формально не является расширением.

Рассмотрим задачу разработки сценариев с точки зрения модельно-ориентированного подхода.

На рис. 1 показаны разные уровни моделирования для платформенно-независимых (PIM) и платформенно-зависимых (PSM) моделей сценария эксперимента. В качестве модели верхнего уровня в обоих случаях выступает Ecore. В терминах Ecore может быть описан физиологический эксперимент и эксперимент, реализованный в системе Presentation, которая выступает в качестве платформы. На уровне модели могут находиться сценарии конкретных экспериментов (например, «GoNoGo Task»). В общей модели экземпляром эксперимента может быть его представление в виде потока работ, а в платформенно-зависимой – скрипт языка Presentation.

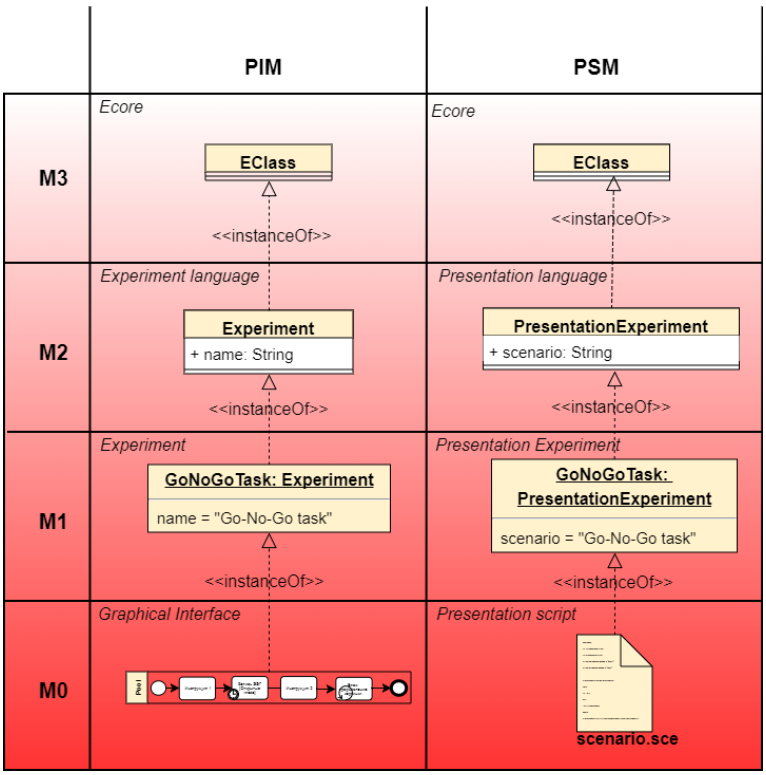

Рис. 1. Уровни моделирования задачи разработки сценария эксперимента

[17] https://www.omg.org/spec/UML/

[18] http://mda-directory.omg.org/vendor/list.htm

В качестве основы для платформенно-независимой модели была использована XML схема эксперимента системы PsychoPy [19]. В соответствии со схемой сохраняются получаемые с помощью PsychoPy Builder [20] эксперименты, при этом она достаточно общая и не содержит информации о деталях реализации. С помощью инструмента EMF была получена Ecore модель, соответствующая схеме на рис. 2.

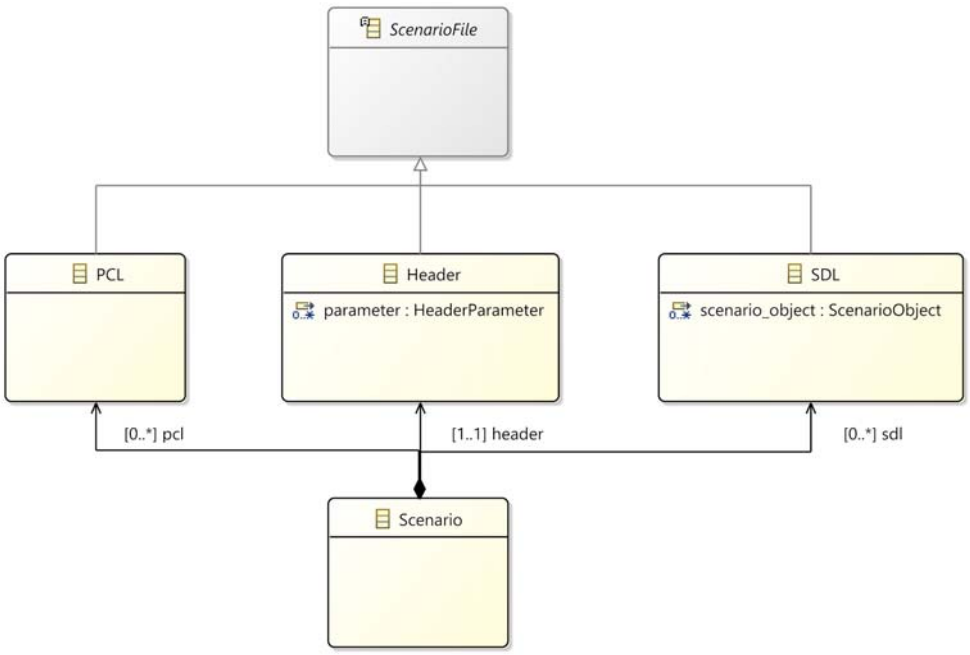

Рис. 2. Описания платформенно-зависимой модели с помощью Ecore (файл сценария)

Платформенно-зависимая модель языка Presentation создавалась с помощью редактора Ecore моделей. Она содержит понятия языка, соответствующие его синтаксическим конструкциям. На рис. 2. показаны концепты моделей верхнего уровня для описания структуры скрипта, выраженных с помощью Ecore.

Работу системы можно представить в виде преобразований моделей (рис. 3):

1) получение экземпляра сценария (1);
2) представление экземпляра сценария в виде экземпляра Ecore модели (1 → 2);
3) преобразование платформенно-независимой модели к модели платформы Presentation;
4) генерирование скрипта для исполнения на Presentation (3 → 4).

Получение экземпляра сценария (1) происходит с помощью графического редактора PsychoPy. Далее, файл сценария в формате XML преобразуется к экземпляру обобщенной модели в формате Ecore с помощью сгенерированного Java кода.

Преобразование 2 → 3 экземпляра обобщенной модели эксперимента в экземпляр модели эксперимента Presentation происходит на уровне M1. Для этого задается трансформация для моделей уровня M2. В предлагаемой реализации используется язык преобразования моделей Epsilon [7]. Он выбран в силу того, что позволяет задавать трансформации Ecore-моделей и может запускаться в качестве независимого программного модуля. Трансформация представляет собой набор правил, описывающих соответствие элементов обобщенной модели элементам платформы Presentation. Запуск преобразования начинается с корня XML-документа и рекурсивно вызывается для вложенных элементов с помощью механизма «ленивого правила» [8].

[19] http://www.psychopy.org/psyexp.html
[20] http://www.psychopy.org/builder/builder.html

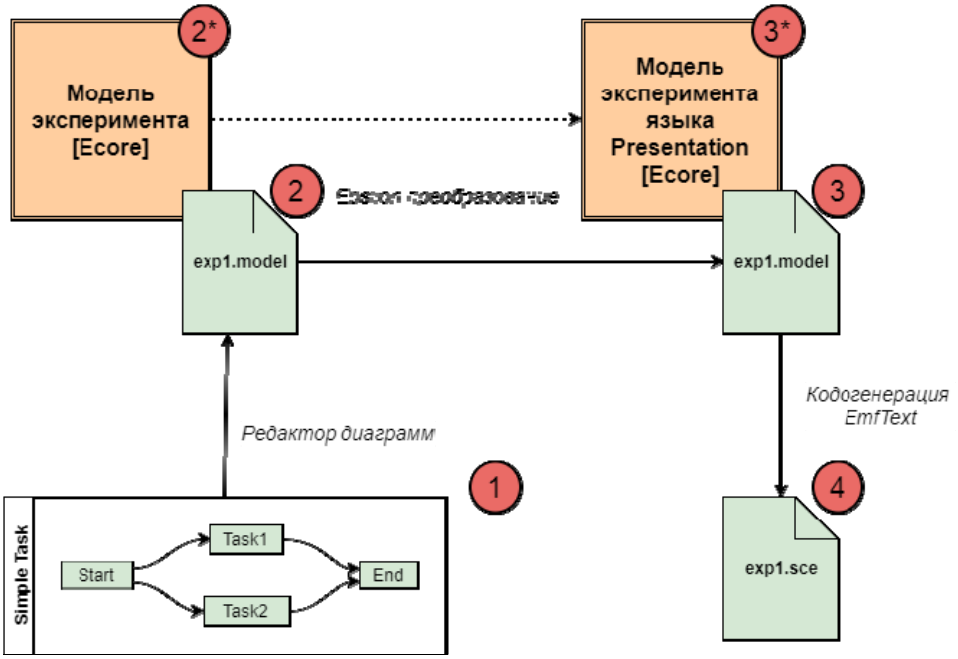

Рис. 3. Преобразования моделей системы

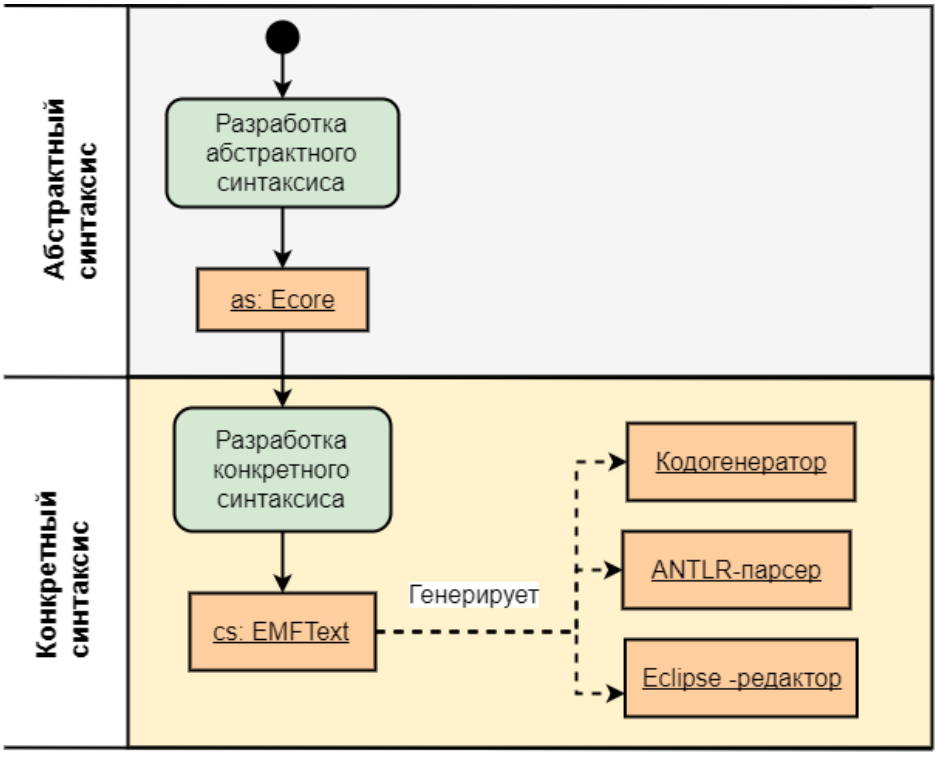

Рис. 4. Процесс разработки в EMFText

Получение исходного кода происходит с помощью инструмента EMFText [9]. На основе модели абстрактного синтаксиса и описания конкретного синтаксиса он позволяет получить ANTLR [21]-парсер, кодогенератор и редактор для предметно-ориентированного языка (рис. 4). Модель абстрактного синтаксиса представляется в формате Ecore. Модель конкретного син-

[21] http://www.antlr.org

таксиса выражается с помощью файла в формате *.cs, содержащего описание токенов, стилей подсветки для редактора и правил грамматики в форме Бэкуса-Науэра, где в качестве служебных символов используются элементы модели абстрактного синтаксиса и их атрибуты.

Сервис представляет собой консольное приложение, написанное на языке Java и позволяющее запустить цепочку преобразований исходного экземпляра сценария эксперимента, полученного с помощью интерфейса PsychoPy, в текст скрипта для исполнения на Presentation.

Заключение и перспективы развития

На основе модельно-ориентированного подхода разработан сервис задания сценариев предъявления стимулов для физиологических экспериментов. Показана работоспособность данного подхода: с помощью преобразования модели эксперимента в модель платформы и описания синтаксической структуры предметно-ориентированного языка Presentation генерируется программный код. Полученный в результате запуска экземпляр модели сценария эксперимента может быть использован в качестве структурированного источника метаданных о сценарии эксперимента. Таким образом, была осуществлена интеграция графического интерфейса PsychoPy и функциональных возможностей Presentation с помощью модельно-ориентированного подхода.

Использованный подход позволил организовать взаимодействие между различными системами за счет информации, выраженной в платформенно-независимой модели. Установлено, что для его применения к системам необходима информация о структуре сценария и стабильность данной структуры для разных версий. Такие условия выполняются для систем с предметно-ориентированным языком создания сценариев. Применение подхода к системам с закрытым форматом не представляется возможным без получения дополнительной информации.

В дальнейшем планируется:

1) расширение подхода для других систем предъявления стимулов, позволяющих задавать сценарии с помощью предметно-ориентированного языка;

2) реализация обратной трансформации: получение экземпляров независимой от платформы модели на основе написанных скриптов Presentation, для чего нужно реализовать обратную трансформацию PSM → PIM;

3) добавление нового уровня моделирования для типовых функциональных проб (закрытые и открытые глаза, классические протоколы).

Список литературы

1. *Кропотов Ю. Д.* Количественная ЭЭГ, когнитивные вызванные потенциалы мозга человека и нейротерапия. Донецк, 2010.

2. *Логвинова К. В.* Современные технологии и средства разработки программного обеспечения // Бизнес-информатика. 2007. № 2.

3. *Paige R. F. et al.* User Experience for Model-Driven Engineering: Challenges and Future Directions // ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems. Institute of Electrical and Electronics Engineers Inc., 2017.

4. *Kleppe A. G., Warmer J. B., Bast W.* MDA explained: the model driven architecture: practice and promise. Addison-Wesley Professional, 2003.

5. *Сорокин А. В., Кознов Д. В.* Обзор Eclipse Modeling Project // Системное программирование. 2010. Т. 5, № 1.

6. *Steinberg D. et al.* EMF: eclipse modeling framework. Pearson Education, 2008.

7. *Kolovos D. S., Paige R. F., Polack F. A. C.* The epsilon transformation language // International Conference on Theory and Practice of Model Transformations. Springer, Berlin, Heidelberg, 2008. P. 46–60.

8. *Kolovos D. et al.* The epsilon book // Structure. 2010. Vol. 178. P. 1–10.

9. *Heidenreich F. et al.* Model-based language engineering with EMFText // Generative and Transformational Techniques in Software Engineering IV. Springer, Berlin, Heidelberg, 2013. P. 322–345.

Материал поступил в редколлегию 20.04.2018

I. V. Brak [1], **Yu. I. Sazonova** [2, 3]

[1] *State Scientific Research Institute of Physiology & Basic Medicine*
4 Timakov Str., Novosibirsk, 630090, Russian Federation

[2] *Institute of Computational Technologies SB RAS*
6 Academician Lavrentiev Ave., Novosibirsk, 630090, Russian Federation

[3] *Novosibirsk State University*
1 Pirogov Str., Novosibirsk, 630090, Russian Federation

brack@physiol.ru, i.sazonova@g.nsu.ru

DEVELOPMENT OF THE SERVICE FOR STIMULI SCENARIO REPRESENTATION BASED ON MODEL DRIVEN ARCHITECTURE

Methods of quantitative data analysis are important in modern physiology. Necessary condition for usage of mathematical statistics, signal analysis and machine learning is the availability of properly collected, marked and prepared data. Thus, preservation of meta-information and structuring results will be useful for their further processing. Physiological experiment consists of a set of trials (samples), in which instructions and certain stimuli are presented to the participant. Reaction on the test sample is recorded as physiological measures. Currently there are many software systems that allow you to create, edit and present scenarios of stimuli representation. Existing systems of presentation stimulus scenario can solve a wide range of tasks but they are not suitable for reusing and there is no universal way to extract metadata of the scenario of the experiment. Purpose of the work is development of the service for stimuli scenario representation with graphical interface, features of saving data in platform independent format and execution in one of the systems. Proposed approach uses model driven architecture principles. The platform-independent model is based on the open format of PsychoPy experiment. Neurobs Presentation system is used to execute scenario. Program code is generated automatically with transformation of the platform-independent model into platform-specific model and describing the syntax of the Presentation domain specific language. Implementation of this approach may be extended to other systems.

Keywords: model driven architecture, code generation, domain specific language, system of stimuli representation

References

1. Kropotov J. D. Quantitative EEG, event-related potentials and neurotherapy. Donetsk, 2010. (in Russ.)

2. Logvinova K. V. Modern technologies and tools for software development. *Business informatics*, 2007, no. 2. (in Russ.)

3. Paige R. F. et al. User Experience for Model-Driven Engineering: Challenges and Future Directions. *ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems*. Institute of Electrical and Electronics Engineers Inc., 2017.

4. Kleppe A. G., Warmer J. B., Bast W. MDA explained: the model driven architecture: practice and promise. Addison-Wesley Professional, 2003.

5. Sorokin A., Koznov D. Review of the Eclipse Modeling Project. *System Programming*, 2010, vol. 5, no. 1. (in Russ.)

6. Steinberg D. et al. EMF: eclipse modeling framework. Pearson Education, 2008.

7. Kolovos D. S., Paige R. F., Polack F. A. C. The epsilon transformation language. *International Conference on Theory and Practice of Model Transformations*. Springer, Berlin, Heidelberg, 2008, p. 46–60.

8. Kolovos D. et al. The epsilon book. *Structure*, 2010, vol. 178, p. 1–10.

9. Heidenreich F. et al. Model-based language engineering with EMFText. *Generative and Transformational Techniques in Software Engineering IV*. Springer, Berlin, Heidelberg, 2013, p. 322–345.

For citation:

Brak I. V., Sazonova Yu. I. Development of the Service for Stimuli Scenario Representation Based on Model Driven Architecture. *Vestnik NSU. Series*: *Information Technologies*, 2018, vol. 16, no. 2, p. 31–40. (in Russ.)

DOI 10.25205/1818-7900-2018-16-2-31-40