# Интеллектуальная система рекомендации вычислительных методов архитектурной акустики

**И.Л. Артемьева[1], А.Е. Чусова[1]**

*[1] Департамент программной инженерии и искусственного интеллекта ДВФУ*

**Аннотация.** В докладе представлен программный комплекс, который позволит специалистам в области архитектурной акустики выбрать наиболее подходящие способы моделирования звука и подбора отделочных материалов в зависимости от поставленных задач и параметров помещения. Отличительной особенностью данной системы является наличие онтологии предметной области, описывающей термины и связи между понятиями, а также модулей для решения различных задач в области архитектурной акустики. Подобный подход позволит рекомендовать пользователю наиболее подходящие для его запроса методы моделирования вследствие учета специфики помещения и функциональных требований клиента. Программная система по запросу позволит оптимизировать и распараллеливать программы, которые написаны с помощью предметно-ориентированного языка программирования. В работе описаны принципы анализа программного кода для выявления участков экономии и применения трансформаций, представленных в банке паттернов. Также в докладе рассматривается подход к построению предметно-ориентированного языка программирования, основанного на онтологии предметной области ODSL (Ontology-Based Domain-Specific Language), позволяющего специалистам описывать алгоритмы, не вникая в используемые методы оптимизации и распараллеливания. Новизна работы заключается в предложенной архитектуре модулей, основанных на прикладной онтологии, что позволяет адаптировать решение под другие предметные области.

**Ключевые слова:** онтология, архитектурная акустика, оптимизация, параллелизм, ODSL

# Intelligent Recommendation System of Computational Methods of Architectural Acoustics

**I.L. Artemieva[1], A.E. Chusova[1]**

[1] *Department of Software Engineering and Artificial Intelligence Far Eastern Federal University*

**Abstract.** The report presents the software package that will allow specialists in the field of architectural acoustics to choose the most appropriate methods for modeling sound and selecting finishing materials depending on the tasks and parameters of a building A distinctive feature of this system is the presence of an ontology of the subject area that describes the terms and relationships between concepts, as well as modules for solving various problems in the field of architectural acoustics. Such an approach will allow the user to recommend the most suitable simulation methods for one's request due to considering the specifics of the premises and the functional requirements of the client. The on-demand software system allows to optimize and parallelize programs written in a domain-specific programming language. The paper describes the principles of source code analysis to identify critical areas and modify them using a bank of patterns. The report also discusses an approach to develop a domain-specific programming language based on domain ontology, ODSL (Ontology-Based Domain-Specific Language), which allows specialists to describe algorithms not accounting for different specific optimization and parallelization methods. The novelty of the work lies in the proposed architecture of modules based on applied ontology, which makes it possible to adapt the solution to other subject areas.

**Keywords:** ontology, architectural acoustics, optimization, parallelism, ODSL

## 1. Введение

Задачи вычислительной акустики обладают рядом особенностей, одна из которых заключается в том, что длина волны зачастую сравнительно больше длины помещения. Если расстояние от источника до преграды меньше половины длины волны, то возникают сложности анализа, обусловленные ближней зоной поля: поскольку звук на этом участке не установлен, соотношение сил и полей имеют сложную для измерения и предсказания зависимость.

Ещё одной особенностью является наличие как объективных показателей акустики помещения, таких как время реверберации, так и субъективных, целиком зависящих от человеческого восприятия. Для моделирования акустических процессов с учётом физиологии слуха используют методы аурализации, где искажения моделируются набором передаточных функций.

Наличие множества моделей, описывающих распространение звука с разной степенью достоверности, также является отличительной чертой вычислительной акустики. На сегодняшний момент не существует единой модели, позволяющей в полной мере учесть все акустические процессы, а факторов, влияющих на звучание, настолько много, что множество исследователей всё ещё работают над созданием других моделей.

Обозревая работы, невозможно однозначно решить, какая из представленных является наилучшей. Таким образом, специалисты должны иметь чёткое представление о применимости тех или иных методов решения задач для достижения акустического комфорта.

Также стоит брать во внимание высокую вычислительную сложность процессов моделирования звука, особенно для случая больших помещений. В связи с этим возникает необходимость в применении различных оптимизаций, в том числе методов распараллеливания. В существующих компиляторах наборы оптимизаций, а также стратегии их выполнения, являются фиксированными. Это не только осложняет поиск оптимальной последовательности преобразований, но и может привести к замедлению программы, поскольку оптимизации могут влиять друг на друга. Таким образом, возникает необходимость в индивидуальной оценке применимости не только алгоритмов решения акустических задач, но и набора оптимизаций для каждого из них для достижения наибольшей оперативности вычислений.

Целью данной работы является описание архитектуры программной системы, позволяющей решать различные задачи в области архитектурной акустики за счёт набора модулей, с возможностью оптимизации программного кода методов, реализованных на предметно-ориентированном языке.

Научная новизна работы заключается в предложенной архитектуре модулей с использованием онтологии предметной области и последующем использовании терминов для реализации алгоритмов.

Статья организована следующим образом: в разделе 2 кратко описаны базовые компоненты системы, а также сценарии использования. В разделе 3 приводится архитектура модулей рекомендации методов решения акустических задач с использованием онтологии. В разделе 4 описывается модуль оптимизации и распараллеливания с использованием предметно-ориентированного языка. Раздел 5 включает описание механизмов анализа зависимостей с использованием графа программы и контекстных формул. В разделе 6 описывается подход оптимизации программ на основе их профиля. В раздел 7 делаются краткие выводы по статье.

## 2. Базовые компоненты системы

На рисунке 1 приведена верхнеуровневая диаграмма программной системы рекомендации методов моделирования. Выделяются следующие базовые компоненты: подсистема постановки акустической задачи, графическая подсистема, онтология предметной области, библиотеки методов решения задач, оптимизаций и параллельных стратегий. Программный комплекс включает в себя набор модулей и имеет возможность добавления новых модулей для расширения области

применимости системы. В данный момент спроектированы подсистемы получения рекомендации об оптимальных методах моделирования звука в различных помещениях, а также оптимизации и распараллеливания их программной реализации.

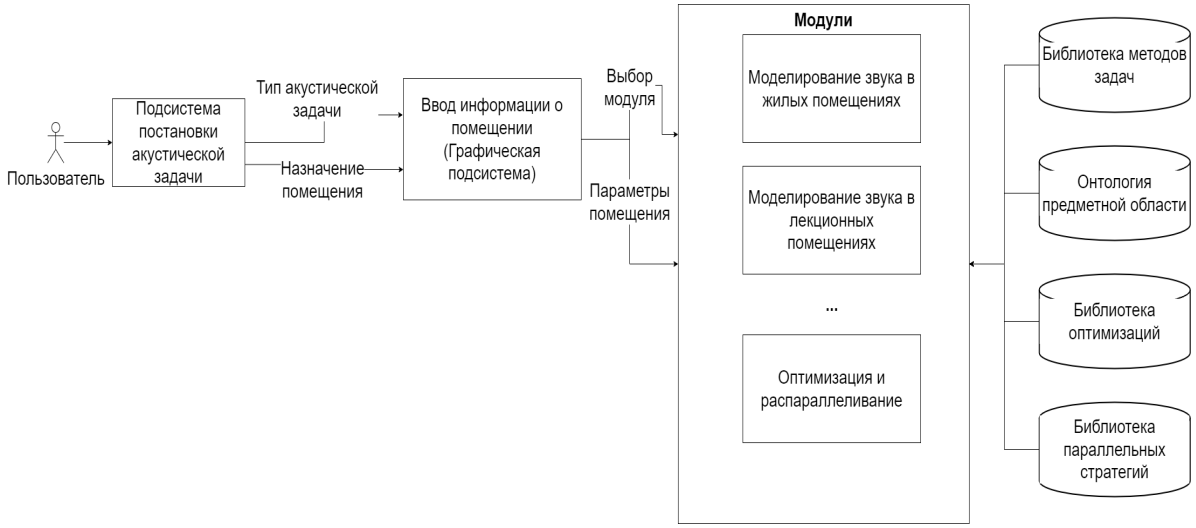


Рис. 1. Верхнеуровневая диаграмма системы

Работа пользователя с комплексом начинается с постановки задачи, поскольку от этого зависит, какие модули необходимо будет задействовать. Процесс выбора модуля показан на рис. 2.

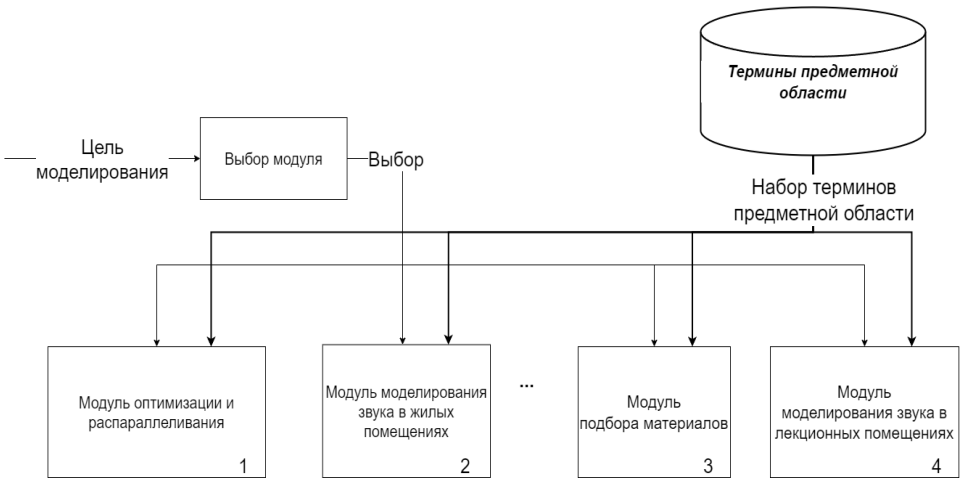


Рис. 2. Структура модуля моделирования звука

Поскольку система включает в себя разнообразные модули, требующие различных входных данных, графическая подсистема предоставляет пользователю инструменты для их ввода в зависимости от задачи и целевого назначения помещения (рисунок 3). Интерфейс данной подсистемы подстраивается под задачу, отображая необходимые параметры, так, например, для моделирования звука будут отображены

поля создания проекта помещения, загрузки готового файла описания строения, различные чертежные инструменты – стены, источники звука и его приёмники. В случае задач подбора материалов для создания определенного акустического оформления помещения больший упор будет сделан на требуемое время реверберации.


Рис. 3. Графическая подсистема

Подобный гибкий интерфейс строится с использованием базы знаний о предметной области, где термины описывают набор свойств, определяющих, для каких модулей может понадобиться ввести значение данного параметра, задаваемого термином [1]. В дальнейшем описанную модель помещения можно будет выгрузить в XML-файл и использовать для получения рекомендаций по методам решения задач или моделирования звука в других системах.

Наличие большого количества методов моделирования обуславливает необходимость в различных классификациях, так как каждый из них может быть применен для нескольких видов акустических задач. Учитывая эту особенность, были выделены модули для решения различного класса задач с учётом их специфики, архитектура которых будет описана ниже. В дальнейшем специалисты предметной области сами смогут дополнять и создавать новые модули за счёт внедрения инструмента ввода новых терминов и условия выбора методов.

После выбора метода моделирования пользователь может модифицировать его программную реализацию на предметно-ориентированном языке, который основан на онтологии предметной области (Ontology-Based Domain-Specific Language, далее ODSL). В этом случае специалист может работать либо со стандартной реализацией

выбранного метода, либо разработать самостоятельно. Затем исходный код метода будет оптимизирован и распараллелен с учётом параметров вычислительных устройств.

Последним этапом является трансляция кода, написанного на языке ODSL, в код на C++ и генерация исполняемой библиотеки, совместимая с системой моделирования физических полей CAMaaS.

## 3. Модули рекомендации методов решения акустических задач

Рассмотрим архитектуру подсистем, позволяющих подобрать оптимальные методы решения акустических задач. В случае задачи по рекомендации способов моделирования, будут использованы модули, структура которых отражена на рисунке 4.

Подсистема на основе введенных параметров помещения и условий выбора, описанных в онтологии, выдаёт рекомендации в пользу теорий описания акустических процессов и их методов с обоснованием принятых решений. Пример подобной онтологии представлен в работе [2].

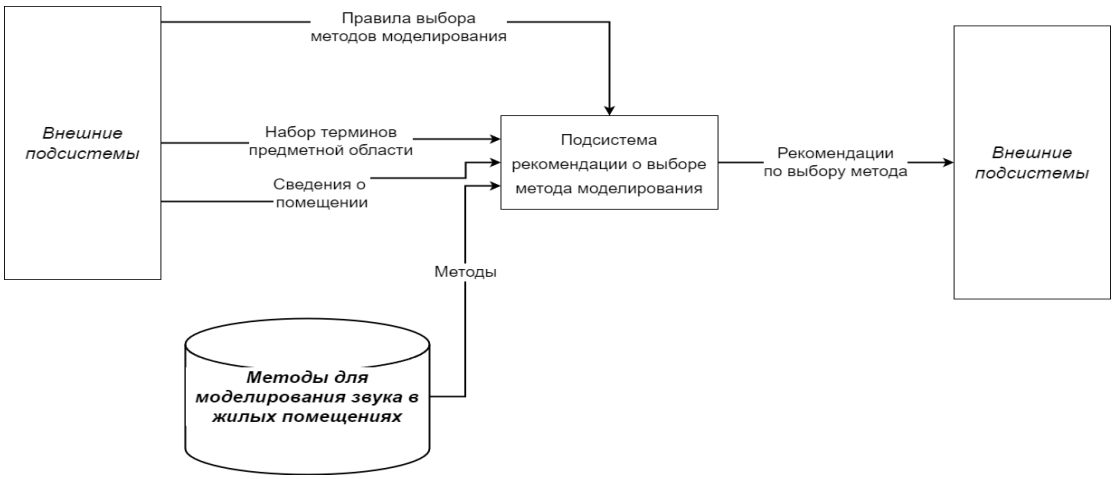


Рис. 4. Структура модуля моделирования звука

Методы моделирования описываются с помощью терминов предметной области, а также содержат набор метаданных, позволяющих определить множество задач, решаемых данным алгоритмом.

## 4. Модули оптимизации и распараллеливания

Помимо модулей, связанных с рекомендацией методов моделирования, у пользователей есть возможность разработать методы решения поставленных задач или использовать стандартную программную реализацию выбранного метода моделирования, выполненную на предметно-ориентированном языке [3] с неявным параллелизмом [4]. После выбора метода моделирования будет предложено модифицировать код с использованием предметно-ориентированного языка. Некоторые

значимые термины предметной области, например, частота, амплитуда, а также арифметические операции и ряд функций используются для описания алгоритма.

Далее созданные методы моделирования по запросу пользователя могут быть оптимизированы, в частности с использованием технологий распараллеливания, с учётом архитектуры вычислительных узлов. Модуль оптимизации и распараллеливания представлен на рисунке 5.

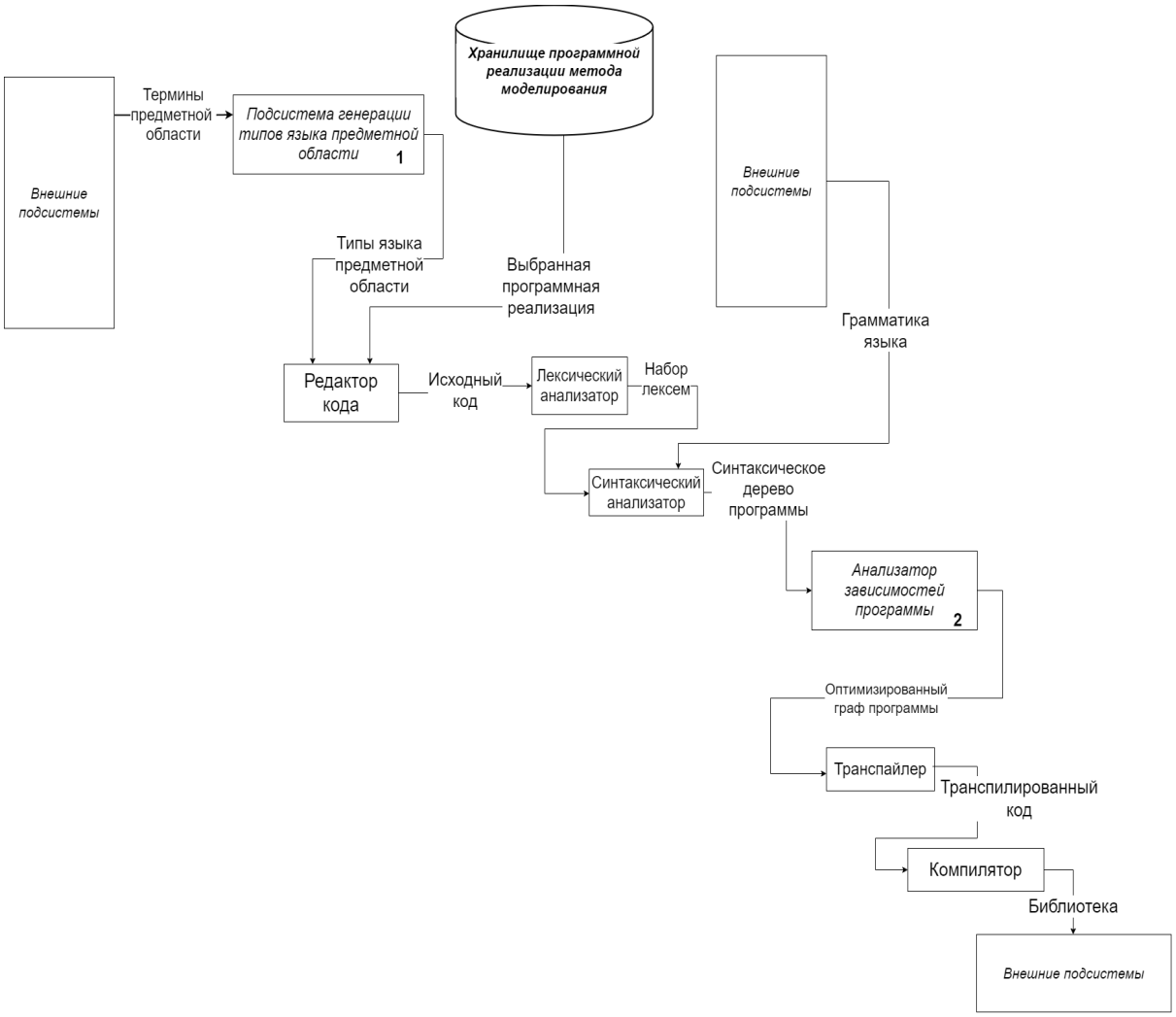


Рис. 5. Структура модуля оптимизации и распараллеливания

Язык включает в себя типы данных, ассоциированных с множеством терминов предметной области, методов, позволяющих взаимодействовать между объектами этого типа, а также множество свойств, характеризующее эти объекты. Каждый термин предметной области, который будет включен в ODSL, может быть связан с несколькими типами. Система генерации типов языка предметной области отражена на рисунке 6.

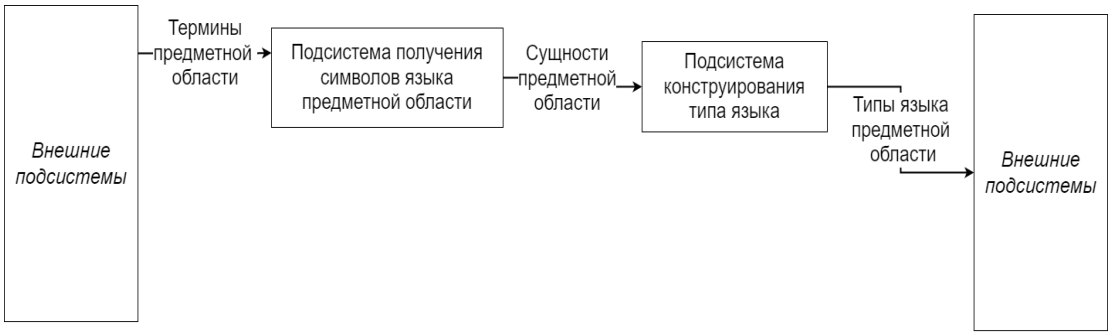


Рис. 6. Структура подсистемы генерации типов языка

Данная подсистема представляет собой сервис, который обращается к онтологии предметной области посредством запросов, получая термины со свойствами в формате RDF/XML, после чего свойства и отношения преобразуются в типы, которые в дальнейшем будут задействованы в реализации программ.

Программный код метода на ODSL языке может быть выбран из базы готовых реализаций или же создан самим пользователем в редакторе кода. Далее текст программы анализируется для выявления участков экономии и определения наиболее подходящих преобразований.

## 5. Анализ зависимостей программы

Прежде чем применять оптимизации и распараллеливание, в частности, необходимо найти участки кода, которые потенциально могут быть преобразованы.

Наиболее традиционным подходом является использование графа зависимостей программы (Program Dependency Graph, далее PDG), отражающем связи по данным и управлению [5]. Несмотря на естественность и понятность подхода, он имеет существенный недостаток: построение графа и его последующий анализ существенно замедляет процесс оптимизации.

Другой подход отражен в работах [6, 7], где представлена интеллектуальная система с использованием онтологий. Анализ программ, допускающий использование логического программирования, наследует преимущества декларативного анализа программ в плане производительности. Что еще более важно, он преодолевает три вышеупомянутых недостатка существующего декларативного анализа программ за счет стандартизации определений терминов предметной области и гибкое представление различных источников знаний. Логические правила имеет преимущество при анализе программы за счёт скорости поиска участков для последующих преобразований, поскольку их применение не занимает столь большого времени, что и построение графа.

Перечисленные выше подходы реализованы в подсистеме анализа зависимостей программы. Детальнее анализатор зависимостей рассмотрен на рисунке 7.

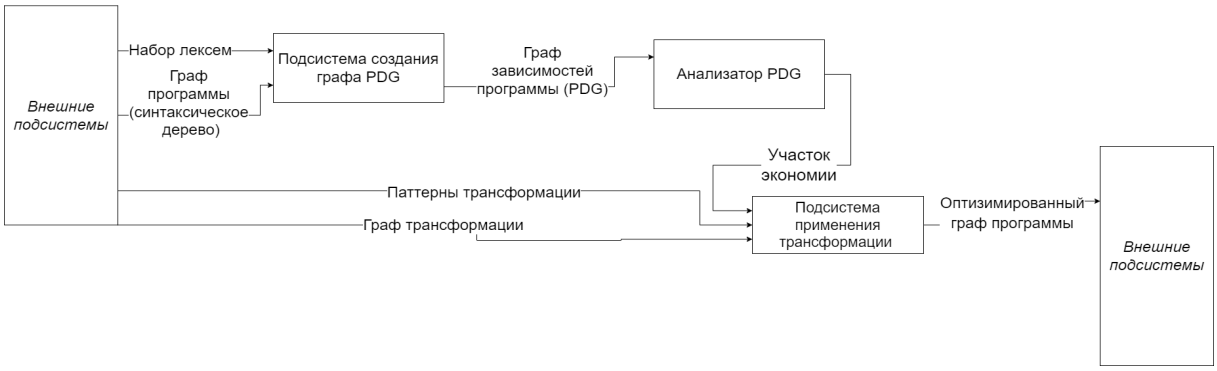


Рис. 7. Структура анализатора зависимостей

На вход системы подается граф программы, знания о моделях структурных программ, а также набор преобразований, каждое из которых содержит алгоритм трансформации и условие, в контексте которого оно применяется. Подсистемы занимаются поиском фрагментов моделей программ, обращаясь к банку паттернов. Хранилище оптимизаций и параллельных стратегий будет хранить графы паттернов, которые необходимо обнаружить, чтобы применить трансформацию, которая также представлена в виде графа; результатом работы модуля является участок экономии – минимальная часть программы, внутри которой будут происходить преобразования. Затем осуществляется замена участка экономии оптимизированным фрагментом кода. Таким образом, задача анализа зависимости сводится к задаче поиска изоморфного подграфа (в данном случае паттерна) графа зависимостей программы.

## 6. Оптимизации на основе профиля программы

Поскольку вычислительные мощности ограничены, необходим индивидуальный подход к определению последовательности преобразований программ. Для подобных оптимизаций, частным случаем которых является распараллеливание, применяется профиль программы. Этот подход позволяет подстроить существующие оптимизации из банка знаний с учётом детальных параметров архитектуры, например, размер кэша и наличие векторных инструкций.

Подход на основе профиля программы продемонстрирован в работе [8], где рассматриваются ряд оптимизаций, применяемые в системе двухэтапной компиляции. Первым этапом преобразований является создание профиля программы, который содержит информацию о подключенных модулях, размерах и именах его функций. В дальнейшем в зависимости от оптимизации происходит оценка блока или функции, где

каждому элементу присваивается вес, который зависит от частоты вызова. На основе выделенной информации принимается решение о применении оптимизаций.

Отдельно стоит обратить внимание на преобразования данных. Несмотря на рост производительности процессоров, до сих пор отмечаются проблемы с ускорением как последовательных, так и параллельных программ. В данный момент основным узким местом является скорость доступа к данным. Учёт структуры данных может оказать существенно влияние на производительность. Подход, учитывающий эти особенности, описан в работе [9] и известен как проектирование на основе данных. Этот подход представляет собой дизайн, ориентированный на данные, который фокусируется на наилучшем размещении данных для центрального процессора и кэш-памяти, что позволяет существенно ускорить выполнение как последовательного, так и параллельного кода.

Подход с созданием профиля программы является перспективным, а его недостатки в виде затрат на создание профиля можно скомпенсировать, задействовав контекстные формулы для сбора характеристик.

## 7. Заключение

В данной работе рассмотрена архитектура системы решения различного рода задач в области архитектурной акустики. Система позволит решать широкий задач, адаптируясь под запросы пользователя. Отличительными особенностями программного комплекса является использование модулей на основе онтологии предметной области для рекомендации методов и разработки библиотек для моделирования, что облегчает взаимодействие акустикам с системой, поскольку для обоснования рекомендаций и разработки кода используются общепринятые понятия из архитектурной акустики. Представленная архитектура программного комплекса может быть масштабирована и распространена на другие предметные области, что, несомненно, облегчит решение задач, с которыми специалисты сталкиваются каждый день.

## Литература

1. Грибова В.В., Федорищев Л.А. Адаптивный генератор-WIMP-интерфейса редакторов базы знаний на основе онтологии // Вестн. Том. гос. ун-та. Управление, вычислительная техника и информатика. — Томск: Национальный исследовательский Томский государственный университет, 2019. — С. 110–119. —

https://cyberleninka.ru/article/n/adaptivnyy-generator-shmr-interfeysa-redaktorov-bazy-znaniy-na-osnove-ontologii
2. Чусова А. Е. Онтология методов моделирования в области архитектурной акустики / А. Е. Чусова // Прикаспийский журнал: управление и высокие технологии. – 2023. – № 1(61). – С. 140-149.
3. Walter T., Parreiras F. S., Staab S. OntoDSL: An Ontology-Based Framework for Domain-Specific Languages // Model Driven Engineering Languages and Systems. MODELS 2009. Lecture Notes in Computer Science, vol 5795. — Berlin : Springer, 2009. — P. 408–422. — https://link.springer.com/chapter/10.1007/978-3-642-04425-0_32
4. Касьянов В.Н., Касьянов Е.В. Методы и технологии конструирования эффективных и надежных программ и программных систем на основе графовых моделей и семантических преобразований // Системная информатика. — Новосибирск : Институт систем информатики СО РАН, 2021. №19. — С. 1-14. — https://doi.org/10.31144/si.2307-6410.2021.n19.p1-14.
5. Yamaguchi F., Golde N., Arp D., Rieck K. Modeling and discovering vulnerabilities with code property graphs // 35th IEEE Symposium on Security and Privacy – San Jose, 2014 – pp. 590–604.
6. Yue Z., Liao G., Shen X. Towards Ontology-Based Program Analysis // The 30th European Conference on Object-Oriented Programming – Rome, 2016 – pp. 26:1–26:25.
7. Selvaraj G. K. Improving Program Analysis using Efficient Semantic and Deductive Techniques. PhD thesis, The University of Auckland, 2022, 203 pp. https://researchspace.auckland.ac.nz/handle/2292/64360
8. Курмангалеев Ш. Ф. Методы оптимизации Си/Си++ приложений, распространяемых в биткоде LLVM с учетом специфики оборудования // Труды ИСП РАН. — М.: ИСП РАН, 2013. Т.24, №1. — С. 127–144.
9. Faryabi W. Data-oriented Design approach for processor intensive games. Master thesis, Norwegian University of Science and Technology, 2018, 179 pp. https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2575669

**References**

1. Gribova V.V., Fedorishchev L.A. Adaptivnyi generator-shmr-interfeisa redaktorov bazy znanii na osnove ontologii // Vestn. Tom. gos. un-ta. Upravlenie, vychislitelnaia tekhnika i informatika. — Tomsk : Natsionalnyi issledovatelskii Tomskii gosudarstvennyi universitet, 2019. — S. 110–119. — https://cyberleninka.ru/article/n/adaptivnyy-generator-shmr-interfeysa-redaktorov-bazy-znaniy-na-osnove-ontologii
2. Chusova A. E. Ontologija metodov modelirovanija v oblasti arhitekturnoj akustiki / A. E. Chusova // Prikaspijskij zhurnal: upravlenie i vysokie tehnologii. – 2023. – № 1(61). – S. 140-149.

3. Walter T., Parreiras F. S., Staab S. OntoDSL: An Ontology-Based Framework for Domain-Specific Languages // Model Driven Engineering Languages and Systems. MODELS 2009. Lecture Notes in Computer Science, vol 5795. — Berlin : Springer, 2009. — P. 408–422. — https://link.springer.com/chapter/10.1007/978-3-642-04425-0_32
4. Kasianov V.N. Integrirovannaia vizualnaia sreda podderzhki konstruirovaniia parallelnykh programm // Problemy informatiki. — Novosibirsk : Institut vychislitelnoi matematiki i matematicheskoi geofiziki SO RAN, 2008. №1. — S. 59-66. — https://cyberleninka.ru/article/n/adaptivnyy-generator-shmr-interfeysa-redaktorov-bazy-znaniy-na-osnove-ontologii
5. Yamaguchi F., Golde N., Arp D., Rieck K. Modeling and discovering vulnerabilities with code property graphs // 35th IEEE Symposium on Security and Privacy – San Jose, 2014 – pp. 590–604.
6. Yue Z., Liao G., Shen X. Towards Ontology-Based Program Analysis // The 30th European Conference on Object-Oriented Programming – Rome, 2016 – pp. 26:1–26:25.
7. Selvaraj G. K. Improving Program Analysis using Efficient Semantic and Deductive Techniques. PhD thesis, The University of Auckland, 2022, 203 pp. https://researchspace.auckland.ac.nz/handle/2292/64360
8. Kurmangaleev Sh. F. Metody optimizatsii Ci/Ci++ prilozhenii rasprostraniaemykh v bitkode LLVM s uchetom spetsifiki oborudovaniia // Trudy ISP RAN. — M.: ISP RAN, 2013. T.24, №1. — S. 127–144.
9. Faryabi W. Data-oriented Design approach for processor intensive games. Master thesis, Norwegian University of Science and Technology, 2018, 179 pp. https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2575669