УДК 004.89

АДАПТАЦИЯ СИСТЕМ ПОДДЕРЖКИ РЕШЕНИЙ НА ОСНОВЕ ВЫСОКОУРОВНЕВЫХ АРХИТЕКТУР

А. В. Карканица[1], В. В. Краснопрошин[2]

[1] *К. т. н., доцент кафедры современных технологий программирования Гродненского государственного университета имени Янки Купалы, Гродно, Беларусь*

[2] *Д. т. н., профессор, заведующий кафедрой информационных систем управления Белорусского государственного университета, Минск, Беларусь*

Реферат

В работе исследуется прикладная задача, связанная с проблемой адаптации систем поддержки принятия решений к изменениям условий среды. Предложен подход, основанный на использовании методологии высокоуровневых архитектур High Level Architecture (HLA).

Во введении выделяются свойства современных задач и среды принятия решений. Обосновывается актуальность разработки технологий, ориентированных на адаптацию компьютерных систем.

В основной части выполнен анализ технологий и средств разработки систем, выделены причины их быстрого устаревания. Для решения проблем масштабируемости и адаптируемости предложен вариант архитектуры программной системы в форме HLA-федерации. Описаны основные этапы предлагаемой технологии и их реализация при проектировании и программировании архитектуры. Эффективность подхода демонстрируется на примере прикладной системы поддержки принятия решений, которая используется в рамках образовательного проекта международной программы Erasmus+.

Ключевые слова: системы поддержки принятия решений, адаптивные системы, высокоуровневая архитектура, High Level Architecture, HLA, федерация, федерат.

ADAPTATION OF DECISION SUPPORT SYSTEMS BASED ON HIGH-LEVEL ARCHITECTURE

A. V. Karkanitsa, V. V. Krasnoproshin

Abstract

The paper investigates an applied problem related to adaptation of decision support systems to environmental changes. An approach based on High Level Architecture (HLA) methodology is proposed. The properties of modern decision-making environment are highlighted. The relevance of the development of technologies focused on the computer systems adaptation is substantiated. To resolve the scalability and adaptability problem a variant of the software architecture in the form of HLA-federation is proposed. The main stages of the proposed technology are presented. The approach effectiveness is demonstrated on the example of an applied decision support system used for implementing international educational project under the Erasmus+ program.

Keywords: decision support systems, adaptive systems, high-level architecture, HLA, federation, federate.

Введение

Стремительная глобализация мира с одновременным формированием информационного общества происходит в условиях значительных изменений природной среды обитания человека, появления новых уже виртуальных сред. Современное общество активно влияет на решение экономических, социальных, политических и других задач государства. В таких условиях возрастает роль и ответственность решений, которые принимаются как отдельными личностями, так и на государственном уровне в целом.

Фундаментальным свойством среды, в которой в настоящее время реализуется процесс принятия решений, стала информационная неопределенность [1]. С высокой интенсивностью появляются новые источники знаний и одновременно наблюдается тенденция их быстрого «устаревания» [2]. Возрастает риск использования недостоверных или устаревших знаний. Предметная область (ПрО) задачи принятия решений (ЗПР) может оказаться неадекватной проблемной ситуации, что может привести к негативным результатам – принятию запоздалых, а порой и недопустимых решений [3, 4].

Существующие технологии построения систем поддержки принятия решений (СППР), призванных максимально автоматизировать процесс решения задач, не успевают или технически не могут адаптироваться к таким изменениям. Нередки случаи внедрения устаревших еще на стадии разработки систем до их введения в эксплуатацию. Поэтому разработка технологий, направленных на решение проблем адаптации систем, является в настоящее время крайне актуальной.

В работе исследуется один из возможных подходов к решению проблемы, когда уже на этапе проектирования системы для создания среды адаптации предлагается использовать технологию моделирования высокоуровневых архитектур (в англоязычной терминологии – High Level Architecture).

doi.org/10.36773/1818-1212-2020-123-5-14-20

Анализ проблемы

Существующие средства разработки СППР классифицируются, как правило, в зависимости от специфики ЗПР, возможностей получения знаний и средств их формализации. Можно выделить следующие классы систем и соответствующих им технологий [5]:

1. Системы (BPM, ERP, CRM, ECM-системы), основанные на обработке корпоративных данных компании и ориентированные на автоматизацию процессов ее управления.

2. Системы оперативно-аналитической обработки данных, так называемые OLAP-системы (On-line Analytical Processing), основанные на идее создания и использования постоянно пополняющегося многомерного хранилища (Data Warehouse).

3. Системы, построенные на основе технологий Semantic Web и ориентированные на извлечение знаний из текстов, представленных в Интернете (глобальный текст).

Анализируя типовую архитектуру существующих СППР, можно выделить несколько объективных причин их быстрого устаревания.

Одной из основных является концептуально жесткая, монолитная, порой избыточная архитектура системы. Модули в таких системах взаимозависимы. Даже несущественное функциональное обновление одного из компонентов затронет все взаимозависимые модули и потребует полного повторного развертывания программного обеспечения.

Вторая – заключается в отсутствии унифицированной структуры компонентов (модулей), единых стандартизированных интерфейсов взаимодействия и обмена данными. Это усложняет интеграцию модулей и возможность их повторного использования.

И, наконец, третья причина – это ориентация типовой архитектуры СППР на использование локальных данных, заранее накопленных в базах и хранилищах, отсутствие возможности приобретения, формализации и быстрого тиражирования инновационных знаний у географически распределенных экспертов (рисунок 1).

Для решения актуальных проблем масштабируемости и адаптируемости предлагается уже на этапах проектирования архитектуры системы и программной реализации ее компонентов использовать технологию моделирования высокоуровневых архитектур, так называемую HLA-технологию. Последняя является международным стандартом и фактически определяет эталонную архитектуру, которая удовлетворяет принципам модульности, совместимости и разделения ответственности.

Архитектура и состав HLA

В основе стандарта лежат пять основных концептов: Runtime Infrastructure (RTI), Federate, Federation, Federation Object Model (FOM), Federation Execution [6, 7]. Концепт RTI является средством коммуникации федератов.

Федераты при необходимости обменивается между собой данными в соответствии с механизмом «Издатель/Подписчик» (англ. Publisher/Subscriber). Для взаимодействия федератов используется сервис Object Management, который отвечает за регистрацию, обновление и удаление объектов федерации, а также за отправку и получение взаимодействий (interactions) [6, 7].

Каждый федерат описывается объектной моделью Simulation Object Model (SOM), которая представлена в формате, основанном на XML. В SOM включаются определения:

- типов объектов (Object Classes) и их атрибутов, совместно используемых в федерации, с указанием возможности их подписки/публикации;
- типов взаимодействий (Interaction Classes) и их параметров, значения которых отправляют или получают федераты;
- простых типов данных атрибутов и параметров взаимодействий, и специфичных для ПрО типов-перечислений.

Объектная модель (FOM) формируется из SOM отдельных федератов. Фактически этот компонент является языком федерации, который можно разработать для любой предметной области и представить в xml-формате.

Процесс разработки систем в соответствии с архитектурой HLA-федерации требует системного структурированного

Рисунок 1 – Компоненты архитектура СППР

подхода. Теоретически в качестве такого подхода можно использовать традиционные технологии разработки программного обеспечения.

Основные этапы технологии включает разработку архитектуры программного обеспечения и ее реализацию средствами выбранной программной платформы. Следует лишь адаптировать эти этапы к получению решения в виде HLA-федерации.

Основные этапы технологии

Следуя методологии HLA, предлагаются рассмотреть следующие основные этапы технологии:

Этап 1. Определение целей и ограничений федерации.

Этап 2. Концептуальный анализ.

Этап 3. Проектирование федерации.

Этап 4. Разработка среды моделирования.

Этап 5. Интеграция федератов и тестирование их совместной работы.

Этап 6. Запуск исполнения федерации.

В соответствии этими этапами необходимо решить следующие основные задачи:

1. Сформулировать соглашение о федерации, определив при этом: типы объектов, федератов и параметры их взаимодействия, а также поведение (функциональность) каждого типа.

2. Построить объектные модели федератов (SOM) и федерации (FOM), а также среду взаимодействия федератов в рамках федерации.

3. Разработать архитектуру федератов и федерации, механизмы адаптации к среде.

4. Выбрать программные средства разработки и на их основе реализовать приложения для федератов, механизмы сборки федерации и развертывания целевой системы.

Рассмотрим решение описанных выше проблем на примере разработки адаптивной системы для принятия решений на основе знаний, получаемых у географически распределенных экспертов.

Объектная модель федерации

Федерация строится для реализации процессов построения и адаптации предметной области: инициализации и декомпозиции целевой задачи, приобретения экспертных знаний, оценки уровня готовности, адаптации предметной области и интеграции знаний. Федерация моделирует состояние и поведение центра (ЛПР) и экспертов, изменение условий задачи, изменение количества и качества знаний, ролей и действий всех участников (акторов). Процессы внутри федерации реализуют сценарий ее исполнения.

Федерация формируется из нескольких типов федератов, которые соответствуют типам участников сцены построения предметной области: федераты Center, Expert, Permission Manager (менеджер разрешений), Validator (валидатор). Обмен данными между федератами определяется их объектной моделью (SOM). Каждый федерат имеет возможность обновлять и отображать (publish) значения атрибутов объектов, которыми он владеет, получать и отправлять данные (interactions), передавать или получать права на владение атрибутами объектов (динамически, во время исполнения федерации).

Объектная модель федерации (FOM) формируется из SOM-моделей федератов. Она включает описание классов объектов (Object Classes) и их атрибутов (Attributes), классов взаимодействий (Interaction Classes) и их параметров (Parameters), простых и перечисляемых типов данных (Simple Datatypes, Enumerated Datatypes). Таким образом, в SOM включаются описания классов Task, Expert, Center, которые наследуются от одного базового класса.

Для каждого класса объектов необходимо описать значения атрибутов, которые доступны федерации. Необходимо также указать тип атрибута, возможности обновления значения (static, conditional) и передачи права владения (transfer, notransfer), а также схему publish/subscribe (рисунок 2).

Также SOM описывает классы взаимодействий федератов: получение и отправку сообщения о выполнении федератом действия, которое может отражаться на состоянии объекта или поведении другого федерата.

Посредством механизмов взаимодействия федерат-Center информирует подписавшихся федератов об изменении состояний объектов Task и Expert. Для адаптации федератов к изменениям в их SOM-модель включено описание взаимодействий двух типов: TaskInteraction и ExpertInteraction. Через механизм наследования уточняется поведение соответствующими классами наследников, которые описывают совершение конкретного действия с объектом федерации. Такими классами – наследниками являются: TaskCreated, TaskStateChanged, ExpertJoined, ExpertAssigned и другие. Все типы взаимодействий и объекты федерации наследуются от базовых классов HLAInteractionRoot и HlaObjectRoot соответственно. Это позволяет выполнить адаптацию SOM-модели федерата простым включением новых классов-наследников.

В результате объединения модели участников федерации SOM в один xml-файл строится объектная модель федерации FOM. Она определяет внешние возможности федерации и

objectClass

1 objectClass

name	Task
sharing	PublishSubscribe
semantics	Concrete task to be solved or decomposed by Expert

attribute (9)

	name	dataType	updateType	updateCondition	sharing	semantics
1	ID	HLAInteger32BE	Static	NA	Publish	Unique task identifier
2	AncestorID	HLAInteger32BE	Static	NA	Publish	Ancestor task ID
3	ExpertID	HLAInteger32BE	Conditional	On change	PublishSubscribe	ID of expert assigned to solve the task
4	Name	HLAunicodeString	Static	NA	Publish	Task name
5	Text	HLAunicodeString	Conditional	On change	Publish	Task statement
6	Spec	HLAunicodeString	Conditional	On change	PublishSubscribe	Solution requirements
7	State	TaskState	Conditional	On change	PublishSubscribe	Current state of the Task
8	Alg	HLAunicodeString	Conditional	On change	PublishSubscribe	Algorithm URL address
9	Tech	HLAunicodeString	Conditional	On change	PublishSubscribe	Technology URL address

2 objectClass

Рисунок 2 – Фрагмент модели SOM

doi.org/10.36773/1818-1212-2020-123-5-14-20

интерфейсы взаимодействия между федератами и сервисами RTI. Поведение федератов определяется в зависимости от их роли в рамках сцены решения задачи.

Федерат Center инициализирует и декомпозирует исходную задачу, формируют БД всех акторов, назначает экспертов, их роли и разрешения. Функциональность данного федерата определяется алгоритмами построения сцены, модели предметной области и ее структурной адаптации [8, 9].

Поведение федерата Expert зависит от его роли (Solver или Editor). При этом федерат Solver формирует паттерн знаний для решения назначенной задачи. Федерат Editor выполняет структурную и информационную адаптацию ПрО. То есть инициирует создание, удаление или обновление объектов федерации. Функциональность федерата определяется соответствующими алгоритмами структурной и информационной адаптации, а также алгоритмом приобретения знаний [9].

Деятельность федерата Permission Manager заключается в проверке разрешений на выполнение действий над объектами федерации (задача, эксперт, знания). Федерат Validator выполняет оценку уровня готовности. Функциональность федерата определяется алгоритмом оценки [10].

Заключительным этапом реализации описанной технологии является разработка архитектуры федератов и федерации в целом.

5. Архитектура федерации и состав адаптивной СППР

Архитектура системы должна обеспечить взаимодействия всех участников сцены решения задачи. Для этого необходимо объединить автономных федератов в одну систему моделирования (федерацию). Предлагается использовать классическую трехуровневую архитектуру проектирования. Она состоит из уровней представления, бизнес-логики и доступа к данным. Добавим к ней еще уровень коммуникаций, который реализует функции взаимодействия федератов с RTI и между собой [11]. На рисунке 3 представлена архитектура федерата Center.

Так как внешняя функциональность федерата определяется его объектной моделью SOM, то реализация уровня бизнес-логики зависит от SOM. Следовательно, SOM также необходимо включить в архитектуру федерата.

Для взаимодействия актора с системой, реализуется графический интерфейс пользователя (GUI) на уровне представления (Presentation Layer). Данный уровень разрабатывается отдельно для каждого типа федерата и зависит от программной платформы, на которой разворачивается федерация. Например, это может быть клиентское веб-приложение, если федерация разворачивается в инфраструктуре Интернет, или десктопное приложение в случае использования локальной сети.

Внутренняя и внешняя логика функциональности федерата реализуется на уровне сервисов (Service Layer). Сервисы могут обращаться к уровням доступа к данным и коммуникаций. Доступ к данным, по запросу с уровня сервисов дает возможность чтения, удаления и обновления данных БД (БД задач, БД экспертов). Уровень коммуникаций реализуется средствами RTI. Для построения архитектуры федератов их объектная модель SOM отображается на описанную многоуровневую архитектуру.

Модуль FederateSOM, на уровне сервисов, реализует объектную модель федерата: классы объектов и взаимодействий, также специфичные для ПрО типы-перечисления (TaskState, ActorRole и другие). Класс CenterFederate является наследником класса GenericFederate и реализует внутреннюю функциональность федерата с использованием библиотеки операций CRUDOperations, а также обработку взаимодействий федератов. Класс FederateManager отвечает за инициализацию федерата и его присоединение к исполнению федерации, отправку взаимодействий подписанных на него федератов. Доступ к данным классы DBContext и FactoryDAO реализуют через одну из технологий доступа к данным БД. Модуль UIController обрабатывает изменения данных и действия пользователя.

Архитектура федерации должна обеспечить взаимодействие входящих в нее федератов. Совместимость федератов обеспечивается модулем FOM, построенным на основе SOM-моделей федератов. В состав архитектуры федератов входят уровни доступа к данным и коммуникаций. Реализация коммуникаций одинакова для всех типов федератов, а доступ к данным зависит от того, к каким сущностям БД имеет право обращаться конкретный федерат. Данные уровни вынесены на уровень архитектуры федерации. Это позволяет избежать дублирования программного кода. Соответствующий вариант архитектуры в форме HLA-федерации представлен на рисунке 4.

Во избежание громоздкости схемы, архитектура федератов представлена в упрощенном виде, а названия уровней обозначены аббревиатурами (SL, PL).

Проведенное выше разделение архитектуры на уровни обеспечивает ее гибкость, взаимозаменяемость и возможность повторного использования компонентов, а также масштабируемость и открытость. Механизм адаптации федерации в данном случае, реализован за счет: изменения внешней модели поведения федерата, расширения функционала самой федерации, а также исключения федерата из состава федерации.

Перечисленные действия реализуется сервисами «Join Federation Execution» и «Resign Federation Execution», которые в процессе исполнения федерации позволяют изменять ее состав.

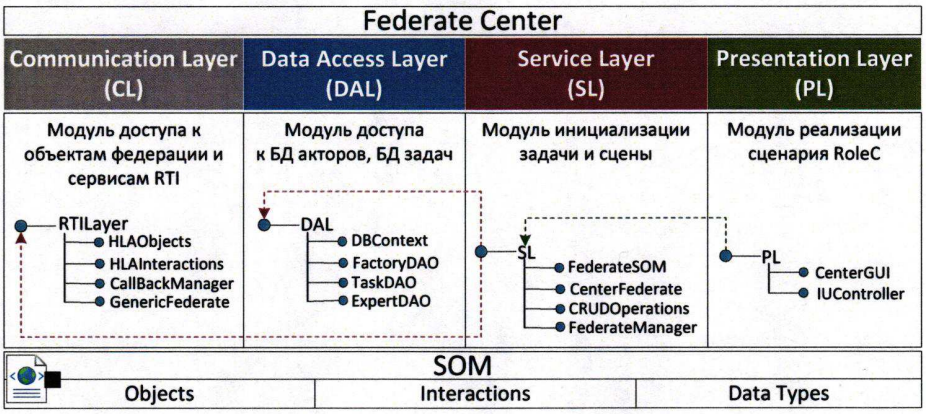

Рисунок 3 – Архитектура федерата Cente

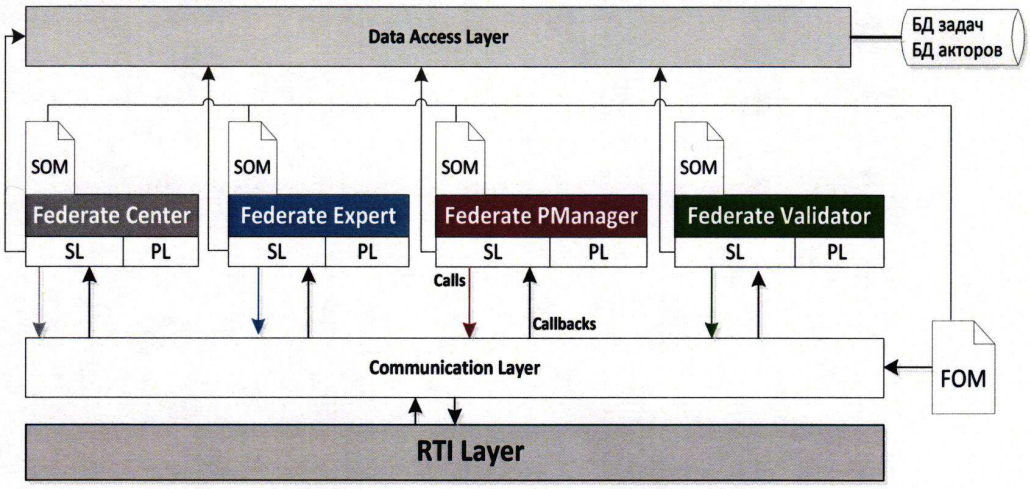

Рисунок 4 – Архитектура федерации

6. Программная реализация

Заключительным этапом исполнения технологии является программная реализация системы и ее развертывание. Последнее можно производить в локальной сети организации (LAN), в глобальной корпоративной сети организации (WAN) или в инфраструктуре Интернета.

Первые два варианта ограничивают использование системы авторизованными пользователями внутренней сети организации (интранет) или ее эксплуатацию с рабочих станций локальной сети. Это усложняет возможность привлечения сторонних территориально удаленных экспертов.

Сложность программной реализации архитектуры федерации определяется затратами на уровне компонента RTI Layer. Стандарт HLA не предоставляет готовой реализации RTI и интерфейсов взаимодействия с ней. Однако, существуют коммерческие и свободно распространяемые реализации стандарта, например, Open HLA, pRTI и др. В большинстве своем они предоставляют низкоуровневые интерфейсы и ограничивают использование языков программирования для разработки федератов (чаще всего C++ и Java).

Для реализации взаимодействия с RTI наиболее популярным является использование так называемых библиотек-оберток (wrapper) над RTI. Классы таких библиотек инкапсулируют сложности низкоуровневой реализации RTI и предоставляют разработчику федератов понятные для реализации стандартных действий интерфейсы.

Для коммуникаций с модулем RTI предлагается использовать свободно распространяемую библиотеку RACoN (RTI Abstraction Component for .Net) [12]. А в качестве программной платформы для архитектуры федерации **–** .Net от Microsoft и входящую в ее состав технологию разработки веб-приложений ASP.NET.

В качестве языка программирования выбран объектно-ориентированный язык C#. Однако для реализации HLA-федерата может быть использован любой язык программирования платформы .Net.

7. Пример

В качестве демонстрации возможностей технологии опишем кратко архитектуру и состав системы поддержки принятия решений, разработанной на основе описанной выше методологии.

Архитектура федератов реализована в форме одноименных приложений-федератов: CenterFederateApp, ExpertFederateApp, ValidatorFederateApp, PManagerFederateApp. По структуре приложения-федераты состоят из нескольких физически разделенных динамических библиотек (.Net-сборок), реализующих соответствующие уровни архитектуры.

Представление всех федератов реализовано в виде веб-приложения на базе фреймворка ASP.NET MVC 5. Коммуникации – с использованием свободно распространяемого пакета с открытым исходным кодом RACoN.

Сервисы уровня доступа к данным (Data Access Layer) используются приложениями-федератами при необходимости осуществления доступа к БД и манипуляций с данными (таблицами задач, экспертов, ролей и разрешений).

На рисунке 5, на примере приложения-федерата ExpertFederateApp, представлена частичная UML диаграмма классов, изображающая, как классы приложения отображаются на соответствующие уровни архитектуры. Уровневая архитектура позволяет использовать принцип разделения ответственностей в отношении трех типов задач, выполняемых каждым федератом: взаимодействие с пользователем (Presentation Layer), реализация поведенческой модели федерата (Service Layer), взаимодействие и обмен данными внутри федерации (Communication Layer).

Уровни слабо связаны между собой (loosely coupled), что позволяет при необходимости изменить реализацию одного из них, не затрагивая остальные. Это обеспечивает расширяемость и гибкость архитектуры, а также значительно упрощает модульное тестирование различных функций приложения.

Для старта исполнения федерации и последующей эксплуатации СППР разработано веб-приложение «Task Sharing». Приложение является точкой входа в систему и представляет собой клиентское приложение, выполняемое в браузере.

На стороне клиентского приложения формируется навигационная панель, соответствующая настройкам системы. С ее помощью пользователь взаимодействует с СППР, получает доступ к объектам федерации (задачи, эксперты, валидатор, менеджер разрешений) и вызывает методы федератов для построения предметной области, ее адаптации и приобретения необходимых экспертных знаний.

Приведенное приложение использовалось в задачах поддержки принятия решений при реализации международных проектов в сфере высшего образования по программе Erasmus+.

doi.org/10.36773/1818-1212-2020-123-5-14-20

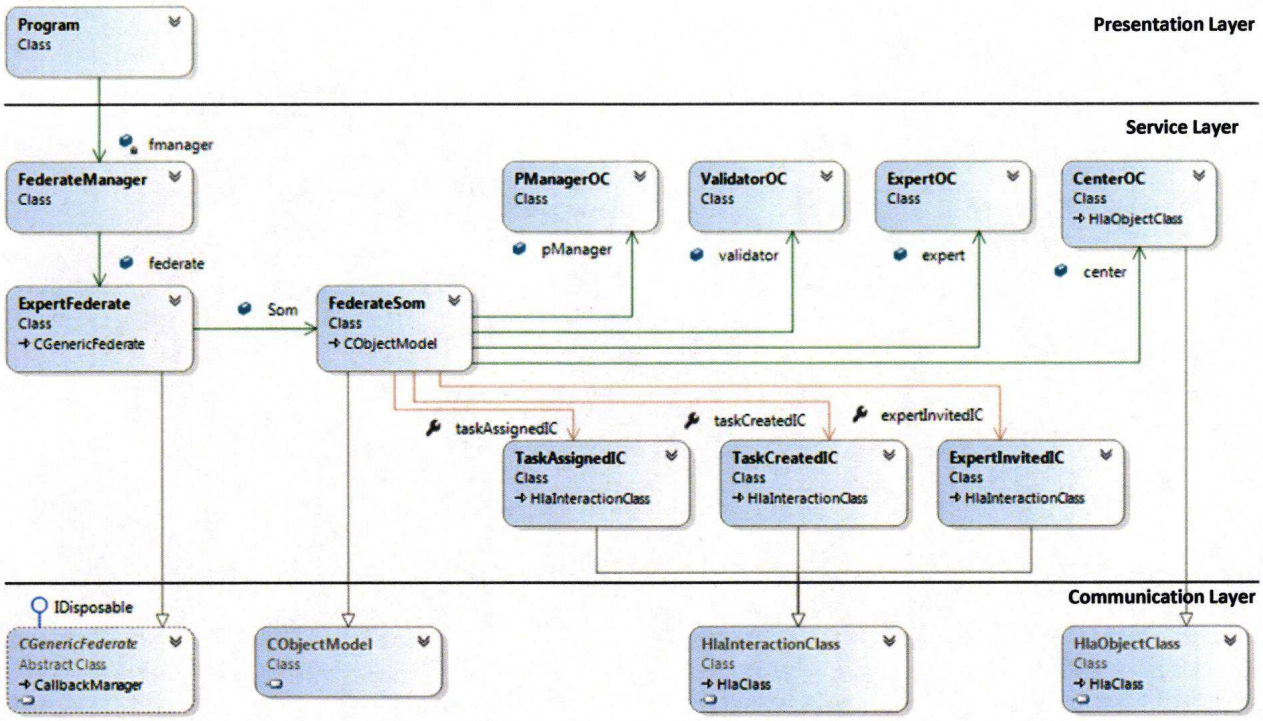

Рисунок 5 – UML диаграмма классов приложения ExpertFederateApp

Заключение

В работе исследуется прикладная задача, связанная с актуальной в настоящее время проблемой адаптации компьютерных систем поддержки решений к изменениям условий среды. Предложен один из возможных подходов к ее решению, основанный на использовании методологии построения высокоуровневых архитектур. Описаны основные этапы предлагаемой технологии и представлена архитектура системы в форме HLA-федерации.

Разработана технология построения системы, основанная на интеграции сервисов инфраструктуры HLA и технологий разработки веб-приложений, приведен вариант ее конкретной реализации. Работоспособность технологии демонстрируется на примере построения прикладной системы в рамках реализации проекта в рамках международной программы Erasmus+ системы образования.

Показано, что использование системы способствовало эффективной организации процессов взаимодействия рабочих групп проекта и интеграции знаний, получаемых у географически распределенных экспертов.

Список цитированных источников

1. Блюмин, С. Л. Модели и методы принятия решений в условиях неопределенности / С. Л. Блюмин, И. А. Шуйкова. – Липецк : ЛЭГИ, 2001. – 138 с.
2. Канеман, Д. Принятие решений в неопределенности: Правила и предубеждения / Д. Канеман, П. Словик, А. Тверски. – Харьков : Издательство Институт прикладной психологии «Гуманитарный Центр», 2005. – 632 с.
3. Карканица, А. В. Моделирование предметных областей для адаптивных систем поддержки принятия решений / А. В. Карканица, В. В. Краснопрошин // Штучний інтелект. – 2018. – № 2(80). – С. 83–93.
4. Карканица, А. В. Онтологический подход к построению моделей динамических предметных областей / А. В. Карканица // Веснік Гродзенскага дзяржаўнага універсітэта імя Янкі Купалы. Серыя 2. Матэматыка. Фізіка. Інфарматыка, вылічальная тэхніка і ўпраўленне. Біялогія. – 2010. – № 1(92). – С. 92–97.
5. Виссиа, Х. Э. Р. М. Принятие решений в информационном обществе: учебное пособие / Х. Э. Р. М. Виссиа, В. В. Краснопрошин, А. Н. Вальвачев. – СПб : Лань, 2019. – 228 с.
6. Topçu, O. Guide to Distributed Simulation with HLA / O. Topçu, H. Oğuztüzün. – Springer, 2017. – 307 p.
7. Kuhl, F. Creating Computer Simulation Systems: An Introduction to the High Level Architecture / F. Kuhl, R. Weatherly, J. Dahmann. – Prentice Hall, 1999. – 224 p.
8. Краснопрошин, В. В. Алгоритмы модификации деревьев для построения динамических предметных областей / В. В. Краснопрошин, А. В. Карканица // Искусственный интеллект. – 2010. – № 4. – С. 388–394.
9. Karkanitsa, A. V. Models, Algorithms, and Architecture for Generating Adaptive Decision Support Systems / A. V. Karkanitsa // Pattern Recognition and Image Analysis. – 2020. – Т. 30. – № 2. – С. 174–183.
10. Карканица, А. В. Оценка неопределенности в адаптивных системах принятия решений / А. В. Карканица // Вестник Брестского государственного технического университета. Серия «Физика, математика, информатика». – 2017. – № 5. – С. 17–20.
11. Karkanitsa, H. Adaptive Decision Support Systems / H. Karkanitsa // Pattern Recognition and Information Processing (PRIP'2019) : Proc. of the 14th Intern. Conf., Minsk, Belarus, 21–23 May 2019. – Minsk : Bestprint, 2019. – P. 342–345.
12. RACoN 0.0.2.4. High Level Architecture (HLA) Runtime Infrastructure (RTI) Abstraction Layer for MS.NET // NuGet Gallery. – Mode of access: https://www.nuget.org/packages/ RACoN/. – Date of access: 25.08.2020.

References

1. Blyumin, S. L. Modeli i metody prinyatiya reshenij v usloviyah neopredelennosti / S. L. Blyumin, I. A. SHujkova. – Lipeck : LEGI, 2001. – 138 s.
2. Kaneman, D. Prinyatie reshenij v neopredelennosti: Pravila i predubezhdeniya / D. Kaneman, P. Slovik, A. Tverski. – Har'kov : Izdatel'stvo Institut prikladnoj psihologii «Gumanitarnyj Centr», 2005. – 632 s.
3. Karkanica, A. V. Modelirovanie predmetnyh oblastej dlya adaptivnyh sistem podderzhki prinyatiya reshenij / A. V. Karkanica, V. V. Krasnoproshin // SHtuchnij intelekt. – 2018. – № 2(80). – S. 83–93.
4. Karkanica, A. V. Ontologicheskij podhod k postroeniyu modelej dinamicheskih predmetnyh oblastej / A. V. Karkanica // Vesnik Grodzenskaga dzyarzhaŭnaga universiteta imya YAnki Kupaly. Seryya 2. Matematyka. Fizika. Infarmatyka, vylichal'naya tekhnika i ŭpraŭlenne. Biyalogiya. – 2010. – № 1(92). – S. 92–97.
5. Vissia, H. E. R. M. Prinyatie reshenij v informacionnom obshchestve: uchebnoe posobie / H. E. R. M. Vissia, V. V. Krasnoproshin, A. N. Val'vachev. – SPb : Lan', 2019. – 228 s.
6. Topçu, O. Guide to Distributed Simulation with HLA / O. Topçu, H. Oğuztüzün. – Springer, 2017. – 307 p.
7. Kuhl, F. Creating Computer Simulation Systems: An Introduction to the High Level Architecture / F. Kuhl, R. Weatherly, J. Dahmann. – Prentice Hall, 1999. – 224 p.
8. Krasnoproshin, V. V. Algoritmy modifikacii derev'ev dlya postroeniya dinamicheskih predmetnyh oblastej / V. V. Krasnoproshin, A. V. Karkanica // Iskusstvennyj intellekt. – 2010. – № 4. – S. 388–394.
9. Karkanitsa, A. V. Models, Algorithms, and Architecture for Generating Adaptive Decision Support Systems / A. V. Karkanitsa // Pattern Recognition and Image Analysis. – 2020. – T. 30. – № 2. – S. 174–183.
10. Karkanica, A. V. Ocenka neopredelennosti v adaptivnyh sistemah prinyatiya reshenij / A. V. Karkanica // Vestnik Brestskogo gosudarstvennogo tekhnicheskogo universiteta. Seriya «Fizika, matematika, informatika». – 2017. – № 5. – S. 17–20.
11. Karkanitsa, H. Adaptive Decision Support Systems / H. Karkanitsa // Pattern Recognition and Information Processing (PRIP'2019) : Proc. of the 14th Intern. Conf., Minsk, Belarus, 21-23 May 2019. – Minsk : Bestprint, 2019. – P. 342–345.
12. RACoN 0.0.2.4. High Level Architecture (HLA) Runtime Infrastructure (RTI) Abstraction Layer for MS.NET // NuGet Gallery. – Mode of access: https://www.nuget.org/packages/RACoN/. – Date of access: 25.08.2020.

Материал поступил в редакцию 17.12.2020