На правах рукописи

РАТКЕВИЧ ИРИНА СЕРГЕЕВНА

РАСШИРЕННЫЙ ЯЗЫКОВОЙ СЕРВИС FRIS ДЛЯ ПРОГРАММИРОВАНИЯ НА ЯЗЫКЕ FORTRAN В MICROSOFT VISUAL STUDIO

Специальность 05.13.11 – «Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей»

Автореферат
диссертации на соискание учёной степени
кандидата физико-математических наук

Саров – 2016

Работа выполнена в государственной корпорации по атомной энергии «РОСАТОМ» Федеральном государственном унитарном предприятии (ФГУП) «Российский федеральный ядерный центр – Всероссийский научно-исследовательский институт экспериментальной физики (РФЯЦ-ВНИИЭФ)», г. Саров.

Научный руководитель	доктор физико-математических наук, главный научный сотрудник **Бартенев Юрий Германович**
Официальные оппоненты:	**Хранилов Валерий Павлович**, доктор технических наук, профессор, заместитель директора Института радиоэлектроники и информационных технологий НГТУ имени Р.Е. Алексеева, г. Нижний Новгород; **Петунин Сергей Александрович**, кандидат физико-математических наук, главный специалист ФГУП «ВНИИА» имени Н.Л. Духова, г. Москва
Ведущая организация	Федеральное государственное унитарное предприятие Федеральный научно-производственный центр «Научно-исследовательский институт измерительных систем им. Ю.Е. Седакова» (ФГУП ФНПЦ НИИИС), г. Нижний Новгород

Защита состоится «___» _____ 2017 г. в ___ часов ___ минут на заседании диссертационного совета Д 002.024.01, созданного на базе Федерального государственного учреждения «Федеральный исследовательский центр Институт прикладной математики им. М.В. Келдыша Российской академии наук», расположенного по адресу: 125047, Москва, Миусская пл., д. 4.

С диссертацией можно ознакомиться в библиотеке и на сайте Федерального государственного учреждения «Федеральный исследовательский центр Институт прикладной математики им. М.В. Келдыша Российской академии наук»: www.keldysh.ru.

Автореферат разослан «____» _____ 2017 г.

Ученый секретарь диссертационного совета,
кандидат физико-математических наук А.Е. Бондарев

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы исследования. Язык программирования Fortran используется при разработке приложений для наукоёмких высокопроизводительных вычислений, учитывающих особенности постоянно развивающихся технологий программирования и аппаратных компонентов.

Современные программы заметно усложнились:

- разработка и усовершенствование ведётся коллективом специалистов;
- используются библиотеки программ, написанные на разных языках;
- применяются технологии распараллеливания вычислений: MPI, OpenMP, SIMD-операции.

Таким образом, возникает **актуальная научная задача** по построению абстрактной (общей) модели анализатора динамически меняющейся во времени программы, выполняющего построение её внутренней структуры и предоставляющего затем эти сведения пользователю в процессе написания текста программы для автоматизации этого процесса.

Ключевой особенностью является априорное знание о некорректности с точки зрения синтаксиса и семантики языка программирования редактируемой пользователем программы. Поэтому инструмент анализа должен делать ряд предположений о том, что имеет в виду пользователь, набирая текущую синтаксическую конструкцию.

Важным моментом является предоставление справки по программным элементам текущего контекста, включая внешние библиотеки, например MPI и OpenMP. Наличие подсказок с описанием спецификации и назначения программных элементов непосредственно в процессе написания текста программы существенно повысит удобство их использования.

Выбор Visual Studio (VS) в качестве целевой среды разработки и стандарта Fortran 2003 обусловлен их широким практическим использованием, в частности, в ФГУП «РФЯЦ-ВНИИЭФ» при написании программ.

Степень разработанности. Будем называть *языковым сервисом*[1] инструмент, отвечающий за предоставление ориентированной на конкретный язык программирования поддержки программиста при редактировании текста программ в редакторе интегрированной среды разработки (IDE). По определению Microsoft (MS), *базовый языковой сервис* должен предоставлять подсветку синтаксических конструкций (далее синтаксиса) программы. *Расширенный языковой сервис* должен, помимо подсветки синтаксиса, предоставлять поддержку технологии IntelliSense и, возможно, другие функции.

Технология IntelliSense, согласно MS, обозначает следующие возможности:

- построение списка элементов сложного объекта (List Members);
- отображение сведений о параметрах процедур (Parameter Info);

[1] Здесь определение языкового сервиса, данное Microsoft только для среды разработки Visual Studio, распространено на все интегрированные среды разработки.

- отображение кратких сведений об элементе языка программирования (Quick Info);
- завершение слова, или автодополнение (Complete Word).

Все современные IDE поддерживают указанные возможности, хотя их названия могут варьироваться. Будем использовать терминологию MS, понимая под ней функциональное назначение той или иной возможности.

Языковые сервисы от используемых в РФЯЦ-ВНИИЭФ компиляторов Fortran фирм Intel и PGI реализуют не все возможности IntelliSense, а также не поддерживают работу с внешними библиотеками и не предоставляют смысловое описание для элементов языка программирования.

Основной частью любого языкового сервиса является блок анализа текста программ. Данная область довольно хорошо развита, например, в работах Ахо А., Сети Р., Ульмана Д., Пэрра Т. Однако дополнительные сложности при разработке анализаторов могут вызывать особенности языка программирования. В Fortran примером может служить отсутствие зарезервированных ключевых слов. Подавляющее большинство существующих анализаторов разработаны для стандартов Fortran 77/90/95, которые уже устарели. Новые стандарты, начиная с Fortran 2003, вводят дополнительные языковые конструкции, например для поддержки объектно-ориентированного программирования.

Среди отечественных программ выделим две разработки ИПМ им. Келдыша: интерактивную программу – анализатор Fortran программ на основе пакета Sage, и систему автоматизации распараллеливания САПФОР. Обе программы имеют графический пользовательский интерфейс и используют для анализа пакет Sage. При этом первая работает с языком Fortran 77, а вторая – с Fortran 95. Отметим, что пакет Sage имеет ограничения на длину обрабатываемого файла – 5000 строк. Если файл имеет большую длину, его необходимо разбивать на блоки, удовлетворяющие требованиям пакета, что не всегда возможно. Второй важный момент – необходимость явного запуска инструментов анализа, т.е. анализ выполняется не во время непосредственного набора текста, поэтому не является в строгом смысле интерактивным.

Среди зарубежных анализаторов для Fortran отметим проект Open Fortran Parser (OFP) Лос-Аламосской национальной лаборатории США. OFP позиционируется как синтаксический анализатор, встраиваемый в другие продукты для анализа программ на языке Fortran 2003/08. Он работает только с синтаксическими определениями элементов, не предоставляя механизмов по извлечению из программы комментариев, а значит, и возможности сопровождать определения элементов их смысловым описанием. Кроме того, OFP реализован на Java (нельзя интегрировать в VS) и существенно изменяет исходный текст, что может исказить часть информации, например касающейся местоположения в нём программных элементов.

Целью диссертационной работы является автоматизация процесса написания текста программ на языке Fortran 2003 в MS VS, направленная на повышение удобства программирования и учитывающая:

- специфику написания сложных программ;
- использование средств параллельного программирования.

Для достижения поставленной цели нужно решить **следующие задачи** по разработке и реализации для языка Fortran 2003:

1) языкового сервиса с полной поддержкой технологии IntelliSense;
2) подсистемы анализа текстов, учитывающей его особенности и позволяющей, помимо основного определения элемента, извлекать из программного кода его смысловое описание;
3) системы хранения информации о значимых элементах языка для обеспечения оперативной работы языкового сервиса;
4) механизма документирования текста программ, позволяющего использовать комментарии в виде подсказок в языковом сервисе;
5) механизма поддержки[2] языковым сервисом внешних[3] программных средств:
   - технологий параллельного программирования MPI, OpenMP, SIMD-операций;
   - внешних проблемно-ориентированных библиотек программ. В РФЯЦ-ВНИИЭФ это УРС-ОФ и ЕФР.

Научной новизной данной работы обладают:

1. Абстрактная модель языкового сервиса, обеспечивающая, в отличие от аналогов, расширяемую интеллектуальную поддержку использования в программе внешних программ не обязательно на том же языке.
2. Модель значимых для языкового сервиса элементов языка Fortran 2003, позволяющая, в отличие от аналогов, строить в оперативной памяти эквивалентное представление программы в виде дерева значимых элементов, снабжённых смысловым описанием своего предназначения.
3. Концепция расширенной поддержки внешних программных библиотек Fortran программы, основанная на использовании языка XML, позволяющая, в отличие от аналогов, выполнять анализ программы с использованием любых средств обработки структурированных XML-данных, включающая:
   - модель прикладных программных интерфейсов (API) Fortran, служащую как для описания Fortran API внешней библиотеки программ, так и для описания Fortran API программного проекта;
   - модель комментариев документирования, позволяющую предоставлять смысловые описания для элементов модели Fortran API.
4. Алгоритм подсветки синтаксических конструкций Fortran-программ в текстовом редакторе интегрированной среды разработки, который, в отличие от аналогов, поддерживает выделение элементов внешних библиотек программ и средств параллельного программирования MPI, OpenMP.

Теоретическая и практическая значимость работы. Результаты диссертационной работы носят как теоретический, так и практический характер. Предложена модель абстрактного языкового сервиса, обеспечивающая расширенную поддержку языка программирования и учитывающая работу с внешними про-

[2] Выделение цветом, включение в подсказки IntelliSense.

[3] Под внешней библиотекой программ понимается библиотека программ, не являющаяся частью текущего решения VS и, возможно, реализованная на языке, отличном от Fortran.

граммными библиотеками, реализованными на языках, отличных от целевого. Практическим результатом является программная реализация языкового сервиса FRIS [1-11] для языка программирования Fortran 2003 и MS VS, использующегося для поддержки написания текстов программ в РФЯЦ-ВНИИЭФ [9], ОАО «НПК КБМ» [10] и АО «КБП им. академика А.Г.Шипунова» [11]. Данный сервис:

- учитывает использование внешних библиотек программ и средств параллельного программирования: MPI, OpenMP, SIMD-операции;
- предоставляет контекстную помощь, включающую смысловое описание элементов программы.

FRIS повышает удобство программирования, позволяет ускорить написание текстов программ, скоординировать «командную» работу и снизить количество ошибок.

Модель прикладных программных интерфейсов Fortran совместно с моделью комментариев документирования может использоваться для автоматического создания документации программиста и/или пользователя.

Разработанные и реализованные алгоритмы анализа Fortran-программ могут использоваться для создания статических анализаторов и препроцессоров.

Методология и методы исследования. При выполнении работы использовались: деятельностный и системный подход, методы теории построения компиляторов, трансляторов и языковых приложений, методы анализа текстов на искусственных языках, методы отображения конкретного синтаксиса в абстрактный, методы построения таблиц символов. При реализации языкового сервиса использовался объектно-ориентированный подход к программированию.

Положения, выносимые на защиту

1. Абстрактная модель языкового сервиса, предназначенная для построения языковых сервисов для языков программирования и интегрированных сред разработки, обеспечивающая поддержку использования внешних библиотек программ не обязательно на том же языке.
2. Модель значимых элементов языка программирования Fortran 2003, позволяющая строить в оперативной памяти эквивалентное представление программы в виде дерева элементов, снабжённых смысловым описанием своего предназначения, и связанные с ней:
   - модель описания прикладных программных интерфейсов Fortran 2003 для внешних программных библиотек;
   - модель XML комментариев документирования для Fortran 2003, позволяющая предоставлять смысловое описание для значимых элементов.
3. Алгоритмы анализа программ на языке Fortran 2003, обеспечивающие при редактировании программы:
   - обработку некорректных конструкций программы;
   - построение дерева значимых элементов с учётом наличия в анализируемом тексте их смыслового описания;
   - подсветку синтаксиса в режиме построчного инкрементального разбора;
   - поддержку выделения элементов внешних библиотек.

4. Программная реализация моделей, алгоритмов и структур данных в языковом сервисе FRIS, предназначенном для ускоренного написания программ на языке Fortran 2003.

Степень достоверности и апробации результатов. Достоверность результатов подтверждается реализацией и тестированием разработанных моделей, алгоритмов, программных модулей и языкового сервиса в целом.

Изложенные в диссертации результаты обсуждались на международных и российских научно-технических конференциях:

- математическая конференция РФЯЦ-ВНИИТФ (2014 г., Снежинск);
- IX Отраслевая научно-техническая конференция молодых специалистов Росатома «Высокие технологии атомной отрасли. Молодёжь в инновационном процессе» в рамках Третьего Международного бизнес-саммита (2014 г., Нижний-Новгород);
- XV Международная конференция «Супервычисления и математическое моделирование» (2014 г., Саров);
- XIII научно-техническая конференция молодых работников и специалистов «Молодёжь в науке» (2014 г., Саров);
- 9-й Весенне-летний коллоквиум молодых учёных по программному обеспечению (ИСП РАН) SYRCoSE 2015 (2015 г., Самара);
- VI научно-техническая конференция молодых учёных и специалистов по тематике «Актуальные вопросы развития систем и средств ВКО» (2015 г., Москва);
- VII Всероссийский конкурс молодых учёных, посвящённый 70-летию Победы (2015 г., Миасс).

Личный вклад. Все описанные модели, алгоритмы разработаны теоретически и реализованы программно лично автором в языковом сервисе FRIS.

Публикации. Результаты диссертации опубликованы в 8 работах, 4 из них статьи в журналах перечня ВАК [1-4], 2 публикации в сборниках трудов и тезисов всероссийских и международных конференций [5-6], получено 2 свидетельства на регистрацию программы для ЭВМ [7-8]. В совместной работе [5] личный вклад автора состоит в разработке описанного в диссертации языкового сервиса, его моделей и алгоритмов.

Объём и структура диссертации. Диссертация состоит из введения, 3 глав, заключения, списка сокращений и условных обозначений, списка литературы и 8 приложений. Полный объём диссертации 195 страниц, в том числе: 48 рисунков, 38 таблиц, список литературы состоит из 110 источников, из них 8 авторских.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

В первой главе рассматриваются основные разработанные модели. Для учёта специфики написания современных сложноструктурированных программ языковой сервис должен:

- предоставлять контекстно-зависимую помощь, содержащую определение элемента языка программирования и его смысловое описание;
- иметь встроенную поддержку:

  - внешних библиотек, для РФЯЦ-ВНИИЭФ это УРС-ОФ и ЕФР;
  - средств параллельного программирования: MPI, OpenMP и SIMD-операций;
- визуально выделять элементы указанных библиотек и средств распараллеливания;
- работать в процессе написания текстов программ.

Указанные требования реализуются в *абстрактной (общей) модели языкового сервиса* [2,4], обеспечивающей расширенную поддержку языка программирования. Языковой сервис может быть представлен в виде пяти основных блоков (рис. 1). Стрелками обозначены обмены данными между блоками. Укажем назначение блоков:

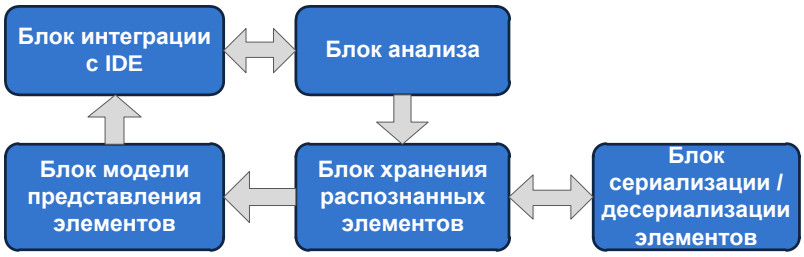

Рис. 1. Абстрактная модель языкового сервиса

- блок интеграции со средой разработки (IDE) – абстрагирует остальные части языкового сервиса от особенностей IDE;
- блок анализа – выполняет анализ текстов на целевом языке программирования и сбор информации для построения эквивалентной структуры программы и подсветки синтаксиса;
- блок хранения – выполняет хранение элементов программы, полученных при её анализе и обработке описаний API и документации внешних библиотек;
- блок сериализации и десериализации – сохраняет элементы блока хранения в виде XML-представления, а также восстанавливает их из XML-представления и помещает в блок хранения распознанных элементов;
- блок модели представления – приводит элементы блока хранения в вид, подходящий для работы компонентов технологии IntelliSense.

Дадим формализованное, в терминах теории множеств, описание абстрактной модели языкового сервиса. При обсуждении грамматики языка используются устоявшиеся определения из работ Ахо А., Сети Р., Ульмана Д.

Модель состоит из следующих элементов (рис. 2), множеств:

- *P*, программных проектов – совокупность файлов метаданных, исходного кода, а также правил для их сборки и отладки;
- *F*, файлов исходного кода – файл с текстом программы на целевом языке;
- *E*, событий – некое свершившееся действие: нажатие клавиши, выполнение команды и т.д.;
- *A*, действий – последовательность исполняемых операций, предназначенных для достижения какой-либо определённой цели;
- *I*, значимых элементов – именованный элемент языка программирования, который можно использовать в программе по его имени и/или псевдониму;
- *V*, элементов представления – шаблон отображения элементов пользователю;
- *C*, цветов для визуального выделения текста – код цвета текста, фона и начертания в доступной палитре цветов ЭВМ;
- *U*, унифицированных элементов описания API – элемент, задающий синтаксическое определение значимого элемента в терминах специального метаязыка;

- *D*, смысловых описаний элементов – элемент, задающий для значимого элемента его описание в терминах специального метаязыка;
- *B*, шаблонов кода – текст с выделенными позициями *H* для заполнения;
- *G=(T,N,R,St)* – грамматики, где: *T* – множество *терминальных символов*, или *токенов*[4]; *N* – множество *нетерминальных символов*[5]; *R* – множество *продукций*, или *правил вывода*; *St* – стартовый нетерминальный символ.

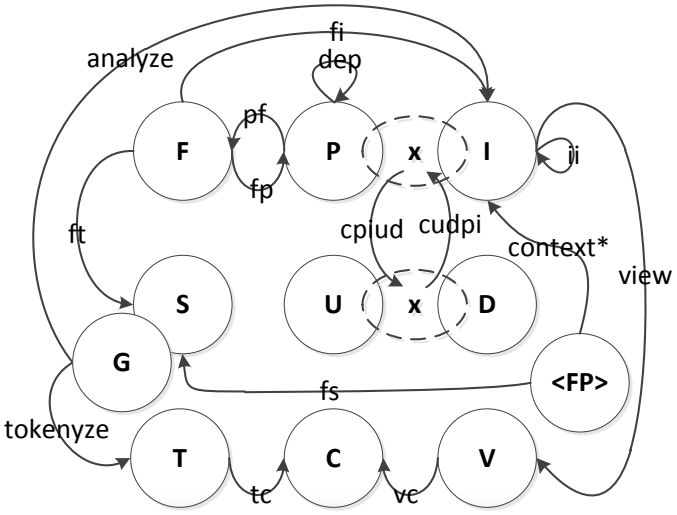

Рис. 2. Основные отношения модели

На рисунке 2: S – множество строк; <FP> – файловая позиция (номер строки в файле, символа в строке); dep – зависимости проектов; pf и fp – принадлежность файлов проектам; fi – зависимость между файлом и значимыми элементами; ft – соответствие файла и его текста; analyze – распознаёт элементы в тексте при помощи грамматики; tokenize – получает токены для заданного текста; tc и vc – получают цвета для токенов и элементов представления соответственно; fs – получает текст для заданной файловой позиции; ii – характеризует вложенность элементов; context* – получает для указанной файловой позиции различные виды контекстов (все видимые элементы, члены типа данных, аргументы процедуры, определения для элемента); view – задаёт соответствие между значимыми элементам и элементами представления; cpiud и cudpi – задают соответствие между проектом и его значимыми элементами и их унифицированным описанием в виде описания API и смыслового описания элементов.

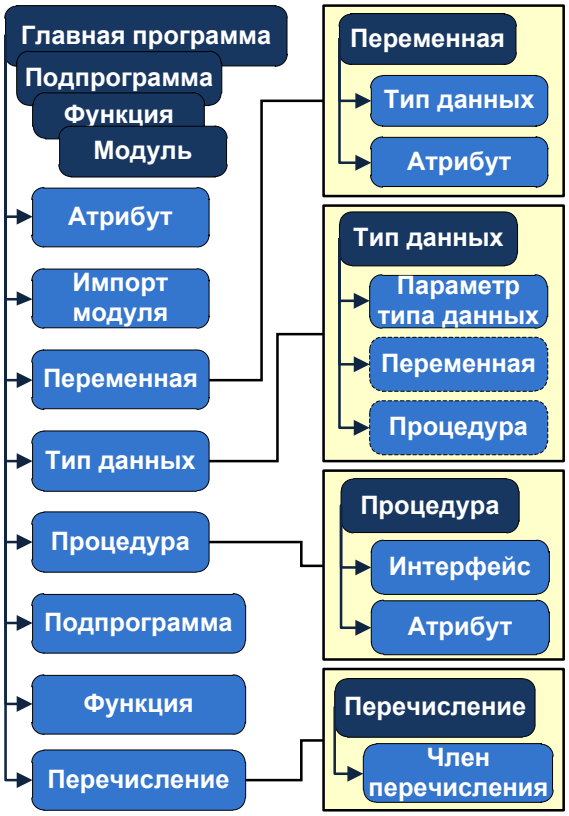

Рис. 3. Модель значимых элементов языка Fortran

Модель *оперирует абстрактными понятиями*, которые не привязаны к конкретному языку программирования и среде разработки, и отражает общую схему построения языкового сервиса и его возможности. Модель можно использовать для построения языковых сервисов для различных языков

[4] Терминальный символ, токен или лексема – элементарные символы языка программирования.
[5] Нетерминал или нетерминальный символ – множество последовательностей терминалов.

программирования (Fortran, C/C++, Java, Python и др.) и IDE (MS VS, Eclipse, NetBeans, CodeBlocks и др.).

В диссертации рассмотрена конкретизация данной модели применительно к языку программирования Fortran 2003 и среде разработки MS VS.

Модель значимых элементов Fortran 2003 (рис. 3), которые необходимо распознавать, хранить и предоставлять возможностям технологии IntelliSense, построена исходя из анализа стандарта Fortran 2003. В качестве значимых рассматривались именованные элементы Fortran, которые могут использоваться в программе по имени или псевдониму.

Данная модель учитывает вложенность областей видимости Fortran и, в отличие от аналогов, строит эквивалентное представление программы в виде дерева значимых элементов, снабжённых смысловым описанием своего назначения. Она определяет содержание связанных с ней моделей: комментариев документирования и описания прикладных программных интерфейсов.

Модель комментариев документирования (рис. 4), позволяет задавать смысловое описание элементам программы как в её теле, так и во внешнем XML-файле специальной структуры, и автоматически отображать данную информацию программисту в дополнение к синтаксическому описанию элементов.

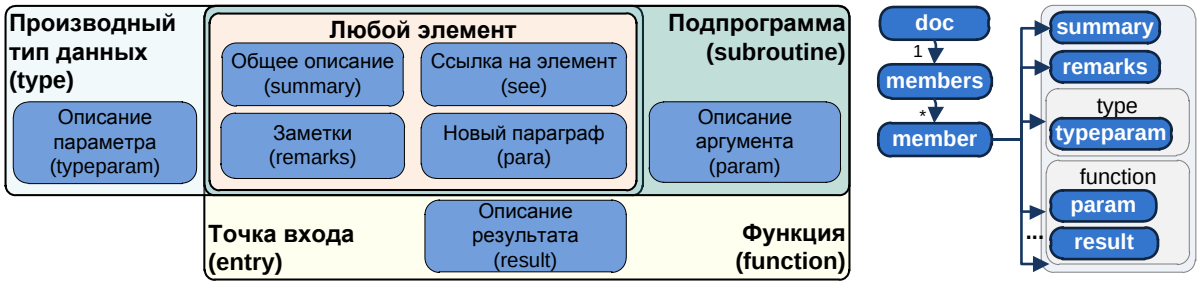

Рис. 4. Модель комментариев документирования

Модель описания прикладных программных интерфейсов (API) Fortran (рис. 5) обеспечивает языковой сервис необходимыми данными, если тексты подключаемой к основному программному проекту библиотеки недоступны или если она реализована на языке программирования, отличном от Fortran. Она позволяет описывать Fortran API во внешнем XML-файле специальной структуры.

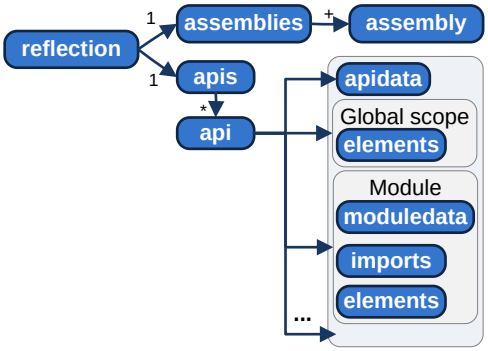

Рис. 5. Модель Fortran API

Модель комментариев документирования и прикладных программных интерфейсов составляют основу *концепции расширенной поддержки внешних библиотек*, позволяющей выполнять анализ программы с использованием любых средств обработки структурированных XML-данных.

Во второй главе изложены основные вопросы построения алгоритмов анализа текста на языке Fortran 2003 и программной реализации языкового сервиса FRIS. FRIS является специализированной реализацией абстрактной модели языкового сервиса (рис. 6).

Основная часть блока интеграции с IDE состоит из реализации классов - наследников от базовых классов Managed Package Framework (MPF) в части языкового сервиса, которые обеспечивают взаимодействие с IDE и отклики на различные события: редактирования текста, выбора пунктов меню и т.д. Эти классы являются своеобразными «пустышками», и их информационное наполнение возложено на плечи автора языкового сервиса.

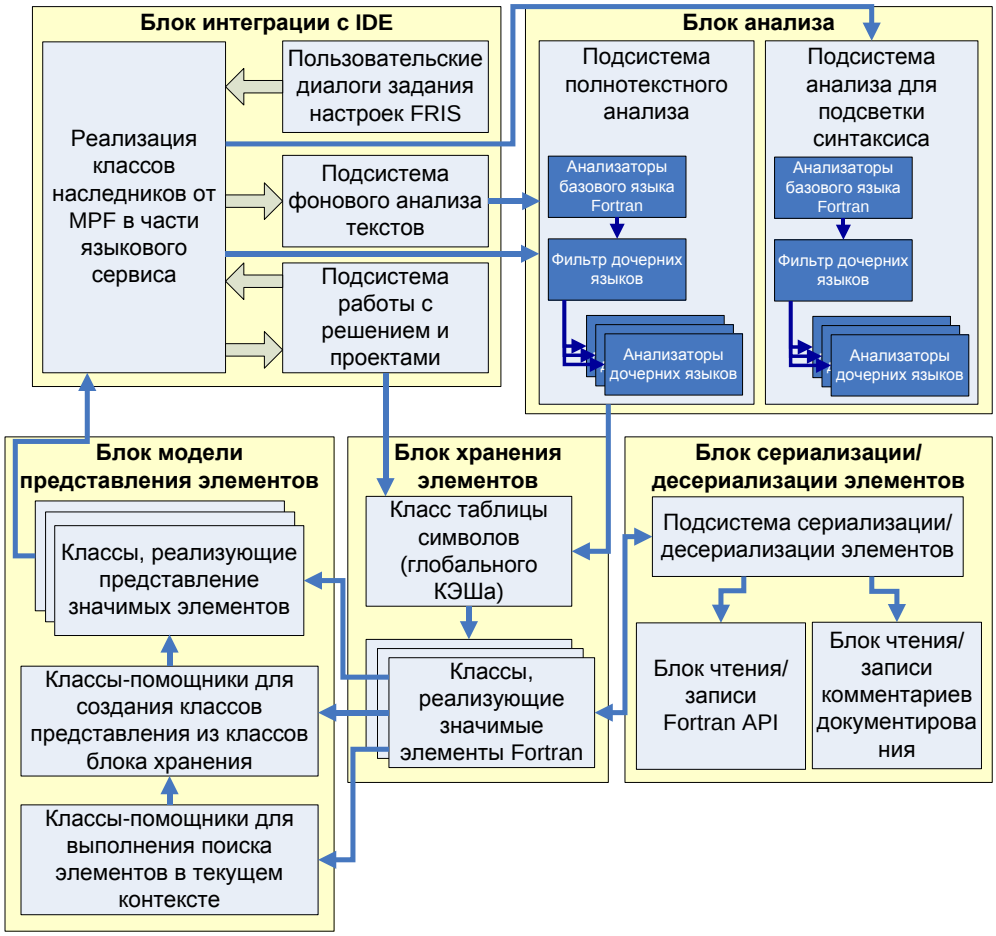

Рис. 6. Организация языкового сервиса FRIS

Подсистема фонового анализа служит для предварительного сбора информации о значимых элементах, осуществляемого при загрузке проекта средой разработки. Она запускает анализаторы подсистемы полнотекстного анализа для каждого файла проекта.

Подсистема работы с проектами и решениями отвечает за работу с программными проектами и составляющими их файлами. Она отслеживает изменения, производимые с проектами: добавление/удаление/переименование файлов, настройка зависимостей проектов друг от друга. Она также отслеживает связи файлов и их представлений в блоке хранения (таблице символов). Данная подсистема хранит виртуальные программные проекты, соответствующие внешним библиотекам программ, полученным из блока сериализации/десериализации. Таким образом обеспечивается встроенная поддержка внешних библиотек программ и средств параллельного программирования.

Блок анализа состоит из двух частей: подсистемы полнотекстного анализа и подсистемы анализа для подсветки синтаксиса. Подсветка синтаксиса выполняется постоянно, после каждого напечатанного символа, а полнотекстный анализ выполняется с большей периодичностью, определяемой VS. Для подсветки синтаксиса используются только лексические анализаторы (для обеспечения приемлемой скорости работы) с набором управляющих правил, а полнотекстный анализ использует полную схему: лексический анализатор, препроцессор, синтаксиче-

ский и семантический анализаторы. Результат анализа для подсветки синтаксиса сразу применяется к тексту программы средствами VS, а результат полнотекстного анализа в виде внутреннего представления дерева значимых элементов заносится в таблицу символов и подсистему работы с проектами и решениями. Таким образом обеспечивается визуальное выделение элементов внешних библиотек в процессе написания текстов программ.

Блок хранения элементов предназначен для хранения и предоставления информации о значимых элементах. Он состоит из таблицы символов и конкретных элементов. Блок наполняется из двух источников, подсистем – полнотекстного анализа и сериализации/десериализации элементов. Информация в нём непрерывно обновляется в результате редактирования пользователем файлов с текстами программ.

Блок сериализации/десериализации элементов предназначен для поддержки работы с внешними библиотеками, чьи исходные тексты недоступны. В своей работе использует два блока чтения и записи: Fortran API и комментариев документирования. Он позволяет из внешних XML-файлов специального формата создавать элементы модели значимых элементов и помещать их в блок хранения и в подсистему работы с решениями и проектами.

Блок модели представления значимых элементов отвечает за предоставление данных для отображения пользователю с учётом текущего контекста (места в файле программы, откуда была запрошена помощь). Таким образом предоставляется контекстно-зависимая помощь, в которой, помимо определения элемента языка программирования, присутствует его смысловое описание.

Рассмотрим обладающую научной новизной *реализацию алгоритмов анализа программ на языке Fortran 2003, обеспечивающих при редактировании программы*:

- обработку некорректных конструкций программы;
- построение дерева значимых элементов с учётом наличия в анализируемом тексте их смыслового описания;
- подсветку синтаксиса в режиме построчного инкрементального разбора;
- поддержку выделения элементов внешних библиотек.

Анализаторы, работающие при редактировании программы пользователем, должны обрабатывать лексические, синтаксические и семантические ошибки. Для этого используется *ослабленная версия грамматики* G^* языка программирования. Рассмотрим общий метод её построения [4].

Пусть есть исходная грамматика языка программирования $G=(T,N,R,S)$, предназначенная для распознавания корректных программ.

1) Дополним множество терминалов T их неполными версиями T' и специальным терминалом для совпадения с любым символом $TANY \to .$, где «.» – любой символ.

2) Во множестве нетерминалов N и правил вывода R выделим те из них, что соответствуют отдельным инструкциям языка программирования N^S и R^S.

3) Дополним их правилом вывода и нетерминалом *any* для инструкции языка в самом общем виде: $any \to .^* \; end\_of\_stmt$, где: «.» – любой терминал; «*» – повторение 0 и более раз; *end_of_stmt* - конец инструкции.

Таким образом получена новая грамматика $G^* = \left(T^*, R^*, N^*, S^*\right)$, где $T^* = T \cup T' \cup TANY$, $R^* = R^S \cup start \cup statment \cup any$, $N^* = N^S \cup start \cup statment \cup any$, $S^* = start$:

$start \to statement^* \; end\_of\_file$
$statment \to s_1 \mid s_2 \mid ... \mid s_n \mid any$, где *end_of_file* – конец анализируемого файла.
$s_i \in R^S$

По построению, для G^* корректными будут все предложения, как корректные, так и недопустимые в G. Таким образом, *анализаторы, использующие грамматику* G^*, *будут обрабатывать некорректные конструкции программы в обычном режиме*.

В G^* отсутствуют продукции для распознавания блочных конструкций. Для их обработки необходимо классифицировать конструкции по убыванию приоритета и ввести набор управляющих правил по работе с ними:

- при обнаружении конструкции с таким же приоритетом текущая конструкция завершается;
- при обнаружении конструкции с более высоким приоритетом завершаются все конструкции с низшим приоритетом, начиная с текущей, и конструкция с таким же приоритетом, что и вновь обнаруженная.

Заметим, что для обработки специальных случаев, зависящих от спецификации конкретного языка программирования, применяются дополнительные правила, учитывающие текущий контекст.

Рассмотрим общую схему работы с дочерними языками (рис. 7). Здесь *базовый язык* – основной язык программирования (Fortran), а *дочерний* – специальный язык, не входящий в определение базового и использующийся внутри него.

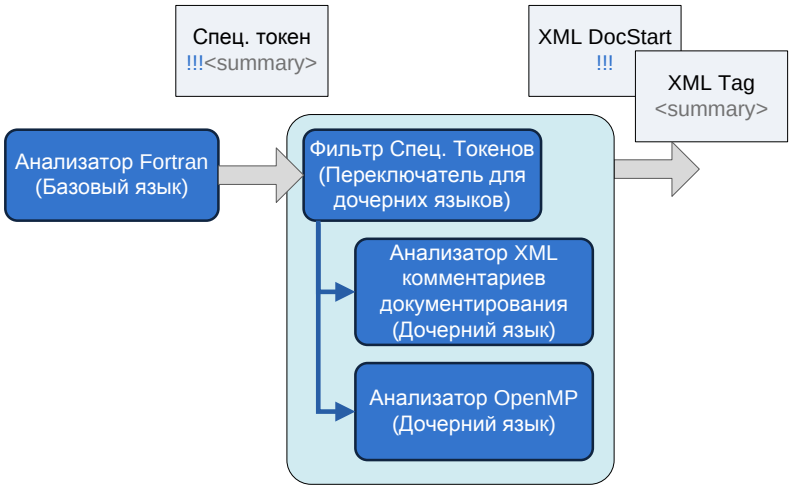

Рис. 7. Общая схема работы анализаторов

Для однозначной идентификации дочернего языка необходим специальный признак – терминальный символ. Всё, что следует за ним, относится к дочернему языку.

Анализатор базового языка генерирует токены, проходящие затем через специальный фильтр. Если токен совпадает с одним из зарегистрированных дочерних языков, вызывается соответствующий анализатор. На выходе получается множество полностью распознанных токенов для всех поддерживаемых языков.

Синтаксический анализатор FRIS использует «оптимистическую» стратегию разбора. Анализ файла ведётся с использованием ослабленной версии грамматики Fortran 2003 в режиме «по одной инструкции». *Для каждой инструкции строится абстрактное дерево разбора (AST), в которое включается смысловое описание, получаемое из комментариев документирования*. Если инструкция не может быть распознана, для неё генерируется особое дерево, включающее в себя все её токены до признака конца инструкции включительно.

В паре с синтаксическим анализатором работает построитель полного дерева разбора (рис. 8), строящий из AST для индивидуальных инструкций *полное дерево разбора*. В задачу построителя входит отслеживание операций открытия и закрытия синтаксических контекстов, в частности их оптимистическое завершение согласно изложенной ранее схеме обработки блочных конструкций с учётом их приоритета.

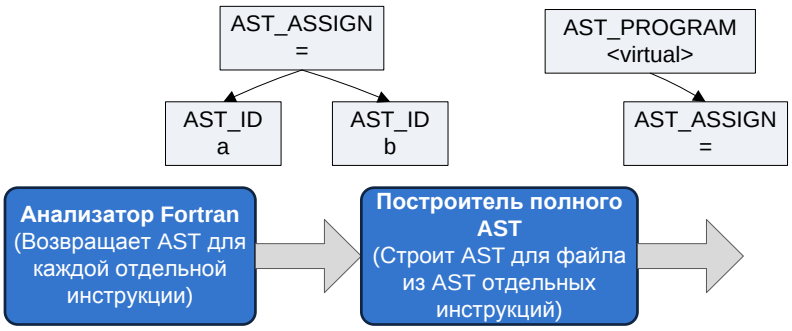

Рис. 8. Схема работы синтаксического анализатора

Таким образом, *синтаксический анализатор на выходе всегда предоставляет корректное дерево разбора текущего файла, при этом нераспознанные инструкции помещаются в специализированные узлы и каждый узел снабжается смысловым описанием, если оно доступно*. Это позволяет упростить алгоритм семантического анализа. Семантический анализатор обходит дерево разбора и собирает информацию обо всех значимых элементах Fortran, которые помещает в блок хранения распознанных элементов.

Рассмотрим особенности реализации *алгоритма анализа программ для подсветки синтаксиса* [1]. Алгоритм предоставляет VS информацию о назначении цветов символам, отображаемым в текстовом редакторе. Ключевой особенностью VS является осуществление *построчного инкрементального анализа* для подсветки синтаксиса и требование, налагаемое на анализаторы, заключающееся в *умении сохранять и загружать своё состояние* для продолжения разбора.

Во FRIS состояние анализатора кодируется признаком переноса строки, кодами текущего токена и лексического правила. При вызове очередного правила разбора запоминается текущее состояние. Отметим, что коды токена и правила позволяют однозначно определить, как продолжить анализ.

Работа с дочерними языками ведётся по общей схеме (рис. 7). Каждый токен дочернего языка выделяется соответствующим ему цветом.

Рассмотрим алгоритм *выделения элементов внешних библиотек*. Имена процедур и типов данных с лексической точки зрения являются идентификаторами. При их обработке производится детальный анализ с привлечением информации из блока хранения распознанных элементов.

Если идентификатор является именем процедуры, то проверяется, является ли он процедурой поддерживаемых внешних библиотек, в частности: встроенной

в язык (intrinsic) процедурой; процедурой OpenMP, MPI, УРС-ОФ или ЕФР. Аналогичные проверки производятся для имени производного типа данных.

Всем специальным токенам назначаются цвета, выделяющие их из остального текста.

Третья глава посвящена вопросам подтверждения работоспособности языкового сервиса FRIS [3].

Демонстрируется поддержка во FRIS технологий параллельного программирования — MPI 3.1, OpenMP 4.0, SIMD-операций, библиотек РФЯЦ-ВНИИЭФ УРС-ОФ и ЕФР, позволяющая упростить работу с ними за счёт визуального выделения и предоставления контекстной помощи для их элементов (рис. 9).

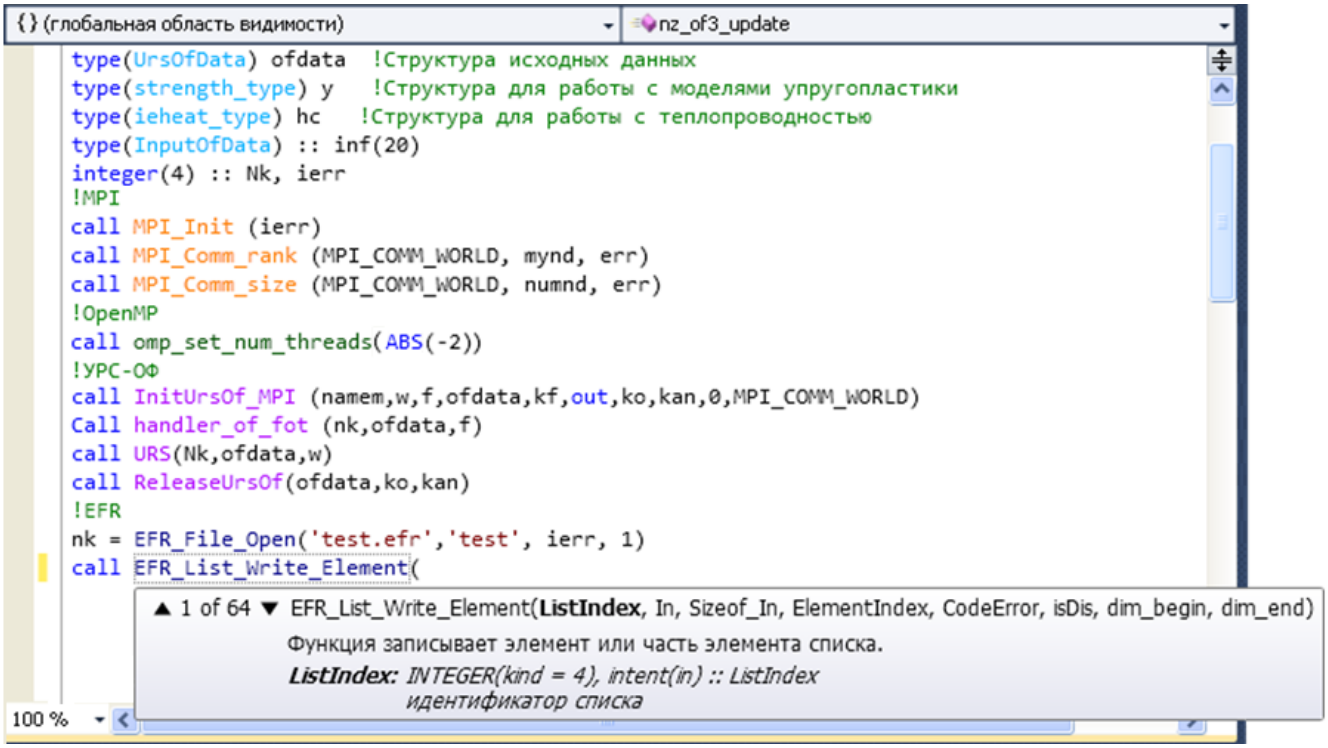

Рис. 9. Комплексный пример реализованных возможностей FRIS

В РФЯЦ-ВНИИЭФ языковой сервис FRIS используется при написании ряда производственных программ (табл. 1).

Таблица 1. Основные сведения о программах

Название	Объём, строк	Использование внешних библиотек	
		Количество	Языки
ЛЭГАК[1]	>800 000	9	Fortran 90, C/C++
КПД-1D[1]	>500 000	9	Fortran 90/2003, C++, Delphi
АРКТУР[1]	>200 000	7	Fortran 90/2003, C++, Delphi
MAGIK[1]	76 000	3	C++
CONCORD[2]	>150 000	6	Fortran 90, C++, Delphi
Алькор[3]	>500 000	15	Fortran 90/2003, C/C++
Newt[4]	>100 000	3	C++

1 – моделирование работы спец. изделий; 2 – реакторных установок; 3 – военных действий; 4 – решение СЛАУ.

Апробация FRIS началась в библиотеке MAGIK, её новая версия реализована с использованием FRIS. Оценка ускорения написания текста с использованием FRIS выполнена на ряде программных алгоритмов, в том числе из библиотеки MAGIK, и составляет 1.4 раза. Для анализа исходных кодов библиотеки MAGIK 76000 строк в 79 файлах FRIS потребовалась 3.21 секунды и 165 Мбайт оперативной памяти для хранения значимых элементов. Оценки получены на ЭВМ: ЦП Intel Core i5 2.8 ГГц, 8 Гбайт ОЗУ.

Отметим, что использование FRIS повышает удобство программирования за счёт использования различных видов контекстно-зависимой помощи. Это позволяет программисту, особенно не автору программы, получить сведения для корректной модификации кода, не отвлекаясь на поиск определений элементов.

Приведём сравнение FRIS с аналогами (табл. 2). Он является единственным продуктом, работающим с внешними библиотеками.

Phortran в Eclipse – единственный сервис, поддерживающий возможности рефакторинга программ, и в этой части имеет преимущество над FRIS. Однако он разрабатывался специально для этой цели, как указано в его описании.

Fortran в CodeBlocks наиболее близок к FRIS по реализованным возможностям. Однако он не учитывает важных особенностей стандарта Fortran 2003, например переноса лексемы на одну и более строк. Если в тексте программы есть такая лексема, данный языковой сервис полностью теряет свою функциональность. CodeBlocks работает с программой в представлении абстрактного синтаксического дерева, т.е. хранит деревья разбора. FRIS работает с программой в представлении модели значимых элементов, отражающей весь программный проект и не связанной с абстрактным синтаксическим деревом.

Отсутствие публикаций с описаниями моделей, реализованных в рассмотренных языковых сервисах, затрудняет сравнение в этой части. Свидетельствами их наличия могут являться зависящие от них функциональные возможности сервисов (табл. 2). Наличие во FRIS расширенной поддержки внешних библиотек, не обязательно на языке Fortran, *может свидетельствовать о новизне и оригинальности* разработанных автором:

- абстрактной модели языкового сервиса;
- модели значимых элементов языка программирования Fortran 2003;
- концепции расширенной поддержки внешних библиотек;
- алгоритма подсветки синтаксических конструкций Fortran программ, поддерживающего выделение элементов внешних библиотек.

Таблица 2. Сравнение языкового сервиса FRIS с аналогами

Возможность	Visual Studio					Кроссплатформенные		
	FRIS	**Intel**	**Lahey**	**PGI**	**Fortran CodeNav**	**Eclipse Phortran**	**CodeBlocks Fortran**	**NetBeans Fortran**
Основные возможности								
Подсветка синтаксиса	есть + внешние библиотеки	есть	есть	есть	нет	есть	есть	есть
Построение списка элементов сложного объекта	есть	нет	нет	нет	нет	нет	есть	нет
Отображение сведений о параметрах процедур	есть	частично	частично	частично	частично	нет	есть	нет
Отображение кратких сведений об элементе языка программирования	есть	частично	частично	частично	частично	частично	есть	нет
Автодополнение	есть	частично	есть	частично	есть	частично	есть	нет
Дополнительные возможности								
Поддержка комментариев документирования	есть, XML-комментарии	нет	нет	нет	нет	нет	есть, Doxygen	нет
Поддержка сниппетов кода	есть + автодополнение	есть	есть	нет	нет	есть	нет	нет
Поддержка внешних библиотек	**есть**	нет	нет	нет	нет	нет	нет	нет
Рефакторинг кода	нет	нет	нет	нет	нет	есть	нет	нет

ЗАКЛЮЧЕНИЕ

В диссертации исследованы проблемы автоматизации процесса написания текста программ на языке Fortran в интегрированной среде разработки Microsoft Visual Studio.

Основные результаты работы:

1. Предложена абстрактная модель языкового сервиса, обеспечивающая расширенную поддержку языка программирования и учитывающая работу с внешними программными библиотеками, реализованными на языках, отличных от целевого.
2. Разработана модель значимых элементов языка программирования Fortran 2003 для внутреннего представления программы и связанные с ней модели, составляющие основу концепции поддержки внешних библиотек:
   - модель описания прикладных программных интерфейсов Fortran, позволяющая языковому сервису отражать значимые элементы внешних библиотек без анализа их исходного кода;
   - модель XML комментариев документирования для Fortran, позволяющая снабжать значимые элементы их смысловым описанием.
3. Разработаны алгоритмы анализа Fortran-программ, работающие при непосредственном редактировании текста программы и обеспечивающие подсветку синтаксиса с выделением элементов внешних библиотек в режиме построчного инкрементального разбора с возможностью сохранения и восстановления состояния разбора в произвольные моменты времени.
4. Реализован языковой сервис FRIS, учитывающий специфику написания современных сложноструктурированных Fortran-программ в Microsoft Visual Studio, использующийся в РФЯЦ-ВНИИЭФ, ОАО КБП и КБМ, повышающий удобство программирования и обеспечивающий поддержку MPI, OpenMP, SIMD-операций и двух проблемно-ориентированных библиотек РФЯЦ-ВНИИЭФ.

СПИСОК РАБОТ, ОПУБЛИКОВАННЫХ АВТОРОМ ПО ТЕМЕ ДИССЕРТАЦИИ

1. Раткевич И.С. Анализатор Fortran программы для реализации технологии IntelliSense в текстовом редакторе Microsoft Visual Studio // Системы управления и информационные технологии, Воронеж: Научная книга, 2015, № 1.1(59), с. 168-172

2. I.S. Ratkevich. FRIS language service for extended Fortran support in Microsoft Visual Studio // Proceedings of the Institute for System Programming Volume 27 (Issue 3). 2015 y. pp. 9-28; ISSN 2220-6426 (Online), ISSN 2079-8156 (Print). DOI: 10.15514/ISPRAS-2015-27(3)-1

3. Раткевич И.С. Языковой сервис FRIS для эффективной разработки Fortran-приложений. Обзор возможностей // Вопросы атомной науки и техники, сер. Математическое моделирование физических процессов. 2015 г. Выпуск 4, с. 49-58

4. Раткевич И.С. Общая модель для расширенной поддержки языков программирования в интегрированных средах разработки. // Успехи современной радиоэлектроники. 2016 г. Выпуск 2, с. 187-194

5. Раткевич И.С., Бартенев Ю.Г., Касаткин С.С.; Языковой сервис FRIS для эффективной разработки Fortran-приложений // Труды XV международной конференции «Супервычисления и математическое моделирование» /Под ред. Р.М. Шагалиева, г. Саров: ФГУП «РФЯЦ-ВНИИЭФ», 2015 г., ISBN 978-5-9515-0300-8, с.385-395

6. Раткевич И.С. Общая модель для расширенной поддержки языков программирования в интегрированных средах разработки // Итоги диссертационных исследований – Материалы VII Всероссийского конкурса молодых учёных, посвящённого 70-летию Победы - М.: РАН, 2015; Т. 3. с. 27-38

7. Свидетельство о государственной регистрации программы для ЭВМ № 2015615397. Языковой сервис для эффективной разработки Fortran приложений в Microsoft Visual Studio («FRIS») / Раткевич И.С.; зарегистрировано в Реестре программ для ЭВМ 18.05.2015г.

8. Certificate of Registration computer program. Registration number TXu 1-936-258. FRIS (FoRtran Intelligent Solutions) / Irina Sergeevna Ratkevich; register of copyrights, United States of America, 31.07.2014

9. Выписка из акта внедрения программного обеспечения FRIS; Акт / ВНИИЭФ; Шагалиев Р.М.; Инв. № 195-2025-25/169122; Саров, 2016.

10. Акт реализации результатов диссертационного исследования младшего сотрудника ФГУП «Российский федеральный ядерный центр – Всероссийский научно-исследовательский институт экспериментальной физики» Раткевич Ирины Сергеевны; Акт / Открытое акционерное общество «Научно производственная корпорация «Конструкторское бюро машиностроения»»; Грачиков Д.В., Шабалкин А.П., Асцатрян А.С., Родионов К.А.; Инв. № 007-122/14431; Коломна, 2015.

11. Акт реализации результатов диссертационного исследования младшего научного сотрудника ФГУП «Российский федеральный ядерный центр – Всероссийский научно-исследовательский институт экспериментальной физики» Раткевич Ирины Сергеевны; Акт / Акционерное общество «Конструкторское бюро приборостроения им. академика А.Г. Шипунова»; Семашкин В.Е., Симаков С.Ю., Болосов Д.А., Гусев А.В.; Инв. № 14803/ДИТ; Тула, 2015.

Подписано в печать 10.01.2017
Формат 60×84/16 Усл. печ. л. 0,93
Тираж 100 экз. Заказ 2268-2016
ФГУП «РФЯЦ-ВНИИЭФ», г. Саров Нижегородской обл., пр. Мира, 37
Отпечатано в ИПЦ ФГУП «РФЯЦ-ВНИИЭФ»,
г. Саров Нижегородской обл., ул. Силкина, 23