УДК 004.4'2

ЯЗЫКОВОЙ СЕРВИС FRIS ДЛЯ ЭФФЕКТИВНОЙ РАЗРАБОТКИ FORTRAN-ПРИЛОЖЕНИЙ. ОБЗОР ВОЗМОЖНОСТЕЙ

И. С. Раткевич
(ФГУП "РФЯЦ-ВНИИЭФ", г. Саров Нижегородской области)

Рассматриваются возможности языкового сервиса FRIS по расширенной поддержке эффективной разработки приложений на языке программирования Fortran в Microsoft Visual Studio. Проводится сравнение возможностей FRIS с имеющимися в наиболее распространенных аналогах от Intel и PGI.

Ключевые слова: FRIS, Fortran Intelligent Solutions, расширение Visual Studio, языковой сервис, Fortran-2003.

Введение

В настоящее время для большинства языков программирования существует множество интегрированных сред разработки приложений, которые позволяют решать самые разнообразные задачи, начиная от написания текста и заканчивая отладкой и автоматическим тестированием полученного приложения.

Для программирования в операционной системе Windows наиболее распространенной средой разработки является Microsoft Visual Studio (VS). Она поставляется с интеграциями для языков программирования, поддерживаемых Microsoft, например C++, C#, Visual Basic, которые реализуют широкие возможности по поддержке программиста при написании текста программы, предоставляя множество списков автоматического дополнения элементов программы и разнообразные виды контекстной помощи. Необходимо отметить, что VS является расширяемой и позволяет дополнять свои базовые возможности практически любым функционалом, например вводить поддержку новых языков программирования.

Однако далеко не все интеграции для поддержки языков программирования реализуют все доступные возможности. Сказанное в полной мере относится к наиболее распространенным интеграциям для языка Fortran [1] от Intel [2] и PGI [3]: разработка приложений с их помощью менее эффективна, чем могла бы быть. Именно такая ощутимая разница между Fortran и базовыми языками от Microsoft явилась основанием для разработки FRIS.

FRIS — это языковой сервис, являющийся расширением для Microsoft VS, который поддерживает все возможности технологии IntelliSense [4] для языка программирования Fortran с использованием стандарта Fortran-2003 [1], повышая эффективность разработки Fortran-приложений.

IntelliSense является ключевой технологией, которая используется в VS для быстрой разработки приложений и обеспечивает возможности поддержки:

- списка членов типов данных (List Members);
- сведений о параметрах подпрограмм и функций (Parameter Info);
- кратких сведений об элементе языка программирования (Quick Info);
- завершения слова или автодополнения (Complete Word).

Эти возможности ускоряют процесс программирования, позволяя улучшить качество разрабатываемых программ и уменьшить время разработки за счет расширения видов контекстной помощи. Возможности IntelliSense отчасти реализованы в интеграциях Fortran от Intel и PGI.

Основные возможности и особенности FRIS

В настоящее время FRIS может функционировать в Microsoft VS 2005/2008/2010. К основ-

ным возможностям, реализованным во FRIS, относятся:

- подсветка синтаксиса;
- выделение структурных элементов кода;
- поддержка панели навигации в текущем документе;
- поддержка XML-комментариев документирования;
- все функции технологии IntelliSense:
  - поддержка списка элементов производных типов данных;
  - поддержка списков автодополнения;
  - отображение сведений о параметрах подпрограмм и функций;
  - отображение кратких сведений об элементе языка программирования;
- переход к определению элемента языка программирования;
- работа со сниппетами (фрагментами) исходного кода;
- работа с панелью *Список ошибок (Error List)*;
- общий механизм расширенной поддержки произвольных (пользовательских) библиотек программ:
  - выделение цветом элементов библиотек в файле исходного кода;
  - предоставление контекстной помощи об элементах библиотек;
  - поддержка встроенных функций и подпрограмм Fortran-2003; библиотек УРС-ОФ [5], используемой при расчете уравнений состояния, и ЕФР [6] для параллельного ввода-вывода сеточных данных.

Отличительной особенностью FRIS является обработка программы, которую редактирует программист в VS, в реальном времени.

В подавляющем большинстве случаев в момент редактирования текст программы является недопустимым с точки зрения лексических и синтаксических правил языка. Это представляет значительную сложность для классической реализации соответствующих анализаторов, поскольку они, как правило, ориентированы на работу с полными грамматиками в предположении, что тексты анализируемых программ не содержат ошибок. В случае обнаружения ошибки такие анализаторы прекращают дальнейший разбор файла исходного кода.

Во FRIS эта сложность преодолевается оригинальной стратегией лексического, синтаксического и семантического разбора. Анализаторы, разработанные и используемые во FRIS, построены с применением генератора анализаторов ANTLR [7] и имеют свои особенности. В частности, разбор осуществляется при помощи стратегии LL(*) [7], включающей разбор с возвратами, сохранением обнаруженных альтернатив разбора и использованием предикатов.

При лексическом анализе неизвестные символы исключаются из входного потока и токены для них не формируются. При синтаксическом анализе, когда фактическая последовательность токенов не соответствует ни одному из возможных правил, возникает ошибка разбора, которую перехватывает FRIS. Он использует специально разработанное правило, которое позволяет обработать такую (недопустимую) последовательность токенов. После этого разбор продолжается.

Рассмотрим более подробно перечисленные выше возможности FRIS.

Подсветка синтаксиса (рис. 1, см. также цветную вкладку) — это выделение цветом различных смысловых частей исходного кода: ключевых слов, строковых литералов, комментариев и т. п. Отличительной особенностью FRIS является возможность выделения цветом элементов библиотек конечного пользователя, а также XML-комментариев документирования.

Выделение структурных элементов (см. рис. 1) позволяет наглядно представить структуру исходного кода путем выделения в самостоятельный блок текста программы между парными конструкциями, например подпрограммы, модуля, условного оператора, оператора цикла и т. п.

Панель навигации (в верхней части рис. 1) позволяет быстро переходить к интересующим элементам, расположенным в текущем файле. Она состоит из двух выпадающих списков. Левый предназначен для отображения информации об областях видимости (модулях и типах), правый — об их членах.

Поддержка XML-комментариев документирования является одной из ключевых возможностей FRIS. Такие комментарии используются для документирования программного кода, а их содержимое включается в выводимые средой разработки контекстные подсказки. При этом комментарий должен иметь следующий вид:

!!!<Имя тэга>
!!!Текст комментария...
!!!</Имя тэга>

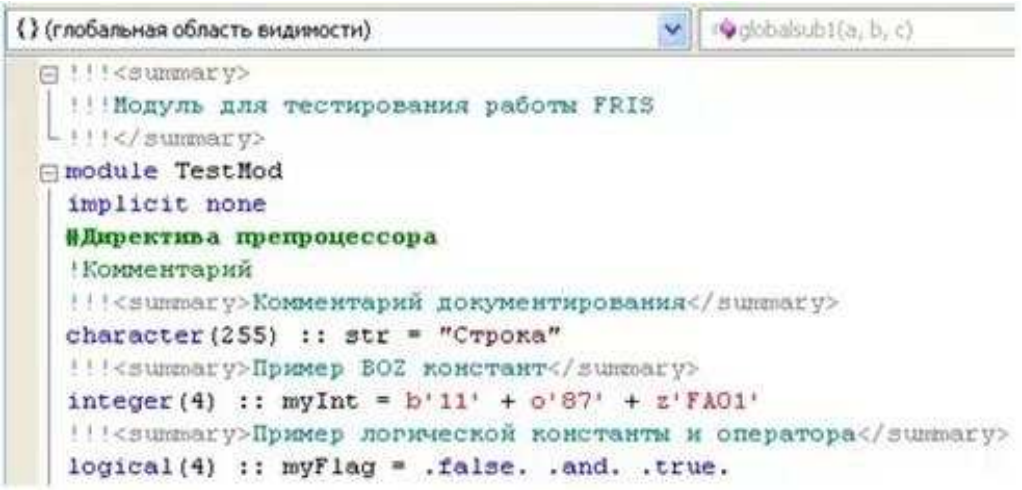

Рис. 1. Текст редактора с использованием языкового сервиса FRIS

FRIS трактует любой комментарий, начинающийся с последовательности из трех восклицательных знаков ("!!!") как комментарий документирования, за которым может следовать один или несколько XML-тэгов документирования. Например, тэг **summary** служит для краткого описания общей информации. Полный список поддерживаемых комментариев документирования приведен на рис. 2.

FRIS обеспечивает возможность *построения списка элементов*, определенных в различных типах данных, с учетом механизма наследования типов. Поддерживается работа со всеми компонентами производных типов данных, включая связанные с типом процедуры. На рис. 3 показано применение данной возможности. При этом в список элементов данных производного наследуемого типа включен особый компонент **secondtype** — ссылка на базовый тип данных. На рис. 4 показан пример работы с процедурами, связанными с типом данных.

Автодополнение является основополагающей возможностью для быстрой разработки приложений. Во FRIS реализованы возможности автодополнения:

- имен модулей;
- имен производных типов данных;
- имен переменных, доступных в текущей области видимости;
- имен подпрограмм.

Рис. 5 демонстрирует автодополнение имен переменных, доступных в текущей области видимости, для правой части оператора. В правой части оператора могут появиться переменные (**b, c, ex, i**), константы (**real_value**), в том числе члены перечислений (enum) (**blue, green, orange, red**), и функции (**fun1**). На принадлежность переменной к одной из перечисленных групп указывает пиктограмма, расположеннная слева от ее имени.

Отображение сведений о параметрах подпрограмм и функций (процедур) позволяет работать

Рис. 2. Поддерживаемые во FRIS комментарии документирования

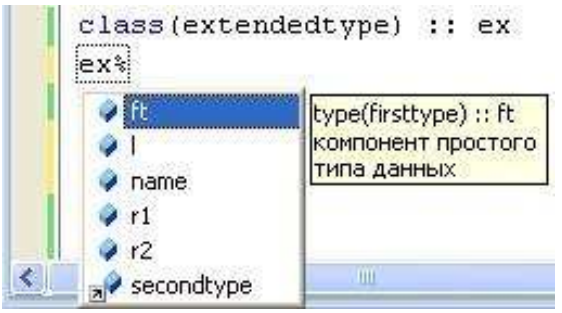

Рис. 3. Список компонентов производного наследуемого типа

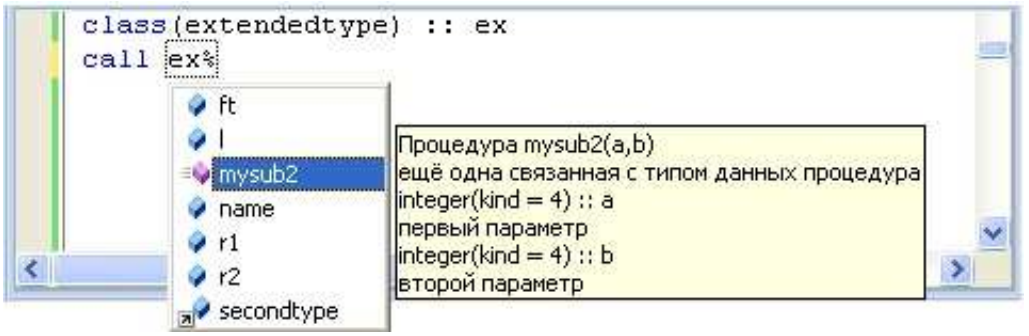

Рис. 4. Работа с процедурами, связанными с типом данных

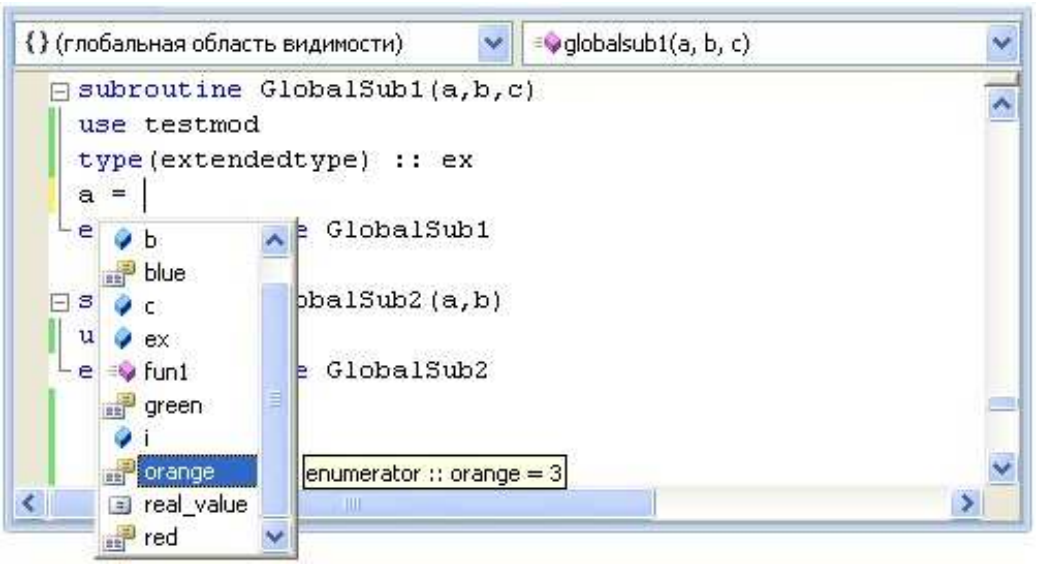

Рис. 5. Автодополнение имен переменных, доступных в текущей области видимости, для правой части оператора

с процедурами, точную сигнатуру которых Fortran-программист может не знать. Вся необходимая информация о наличии перегруженных процедур, их аргументах, их количестве, названиях и типах данных будет отображена во всплывающей подсказке (рис. 6).

FRIS поддерживает работу с родовыми (обобщенными) интерфейсами, когда для группы процедур объявляется одно имя (рис. 7).

К *отображению кратких сведений об элементе языка программирования* относится любая контекстная помощь, отображаемая в виде подсказок. В контекстные подсказки, выводимые FRIS, включается информация, полученная не только из определения элемента языка программирования, но и из комментариев документирования (рис. 8).

Довольно часто встречающейся на практике задачей является *переход к определению того или иного элемента языка программирования* для получения наиболее полной информации о нем (рис. 9).

FRIS поддерживает *работу со сниппетами (фрагментами) исходного кода*. Сниппет представляет собой некоторый шаблонный фрагмент исходного кода, в котором могут содержаться пустые, предназначенные для заполнения места. Создание и использование таких шаблонов существенно ускоряет процесс разработки программного обеспечения.

Необходимо отметить, что все доступные сниппеты включаются FRIS в списки автодополнения (рис. 10). Сниппеты снабжены заготовками для комментариев документирования. На рис. 11 показан результат вставки сниппета FRIS для модуля.

Сразу после вставки сниппета происходит переход в режим его редактирования. В этом ре-

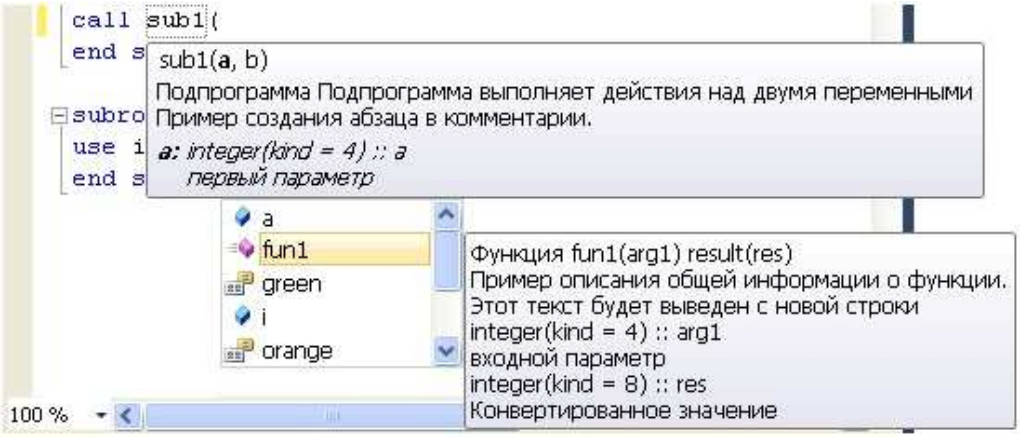

Рис. 6. Отображение параметров подпрограммы и автодополнение ее аргументов

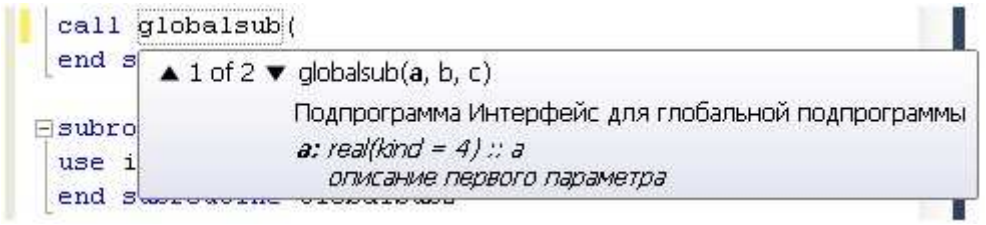

Рис. 7. Отображение параметров для родового интерфейса

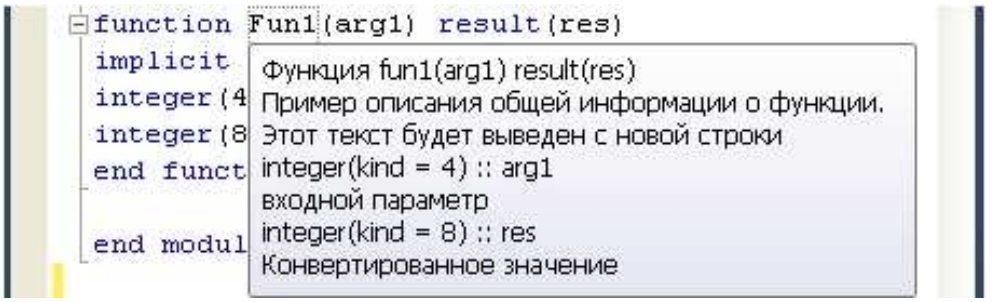

Рис. 8. Отображение информации о функции

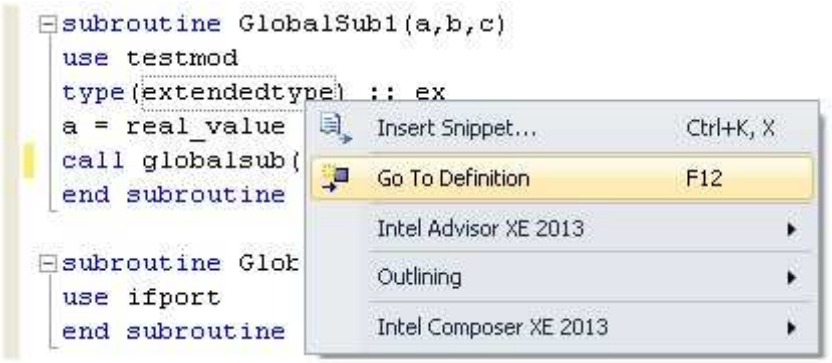

Рис. 9. Переход к определению производного типа данных

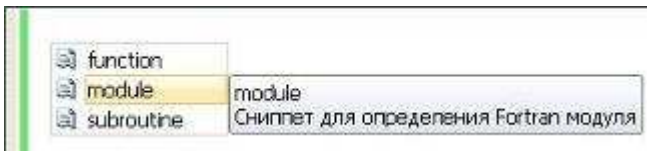

Рис. 10. Сниппеты исходного кода в списке автодополнения

```
!!!<summary>
!!!
!!!</summary>
module Module1
!Импортирование других модулей
implicit none
!Описания переменных
!и типов данных

contains
!Описания внутренних функций
!и подпрограмм

end module
```

Рис. 11. Вставка сниппета FRIS для модуля

жиме пользователь может заполнить шаблонные части нужными значениями. Переход между шаблонными частями осуществляется при помощи клавиши <**Tab**>, ввод шаблонной части заканчивается нажатием клавиши <**Enter**>.

FRIS поддерживает работу с панелью *Список ошибок (Error List)*, которая предназначена для вывода сообщений об обнаруженных ошибках. На данный момент *Список ошибок* содержит предупреждения, относящиеся к работе с комментариями документирования, а также сообщения о грубых синтаксических ошибках (рис. 12).

Особенностью FRIS является возможность *поддержки произвольных библиотек*, которая включает в себя:

1) предоставление контекстно-зависимой помощи для элементов библиотеки;
2) обозначение цветом элементов библиотеки.

Первая задача решается автоматически, если определения модулей, производных типов данных, интерфейсов, подпрограмм и функций библиотеки содержатся в файлах исходного кода, непосредственно подключаемых к программному проекту. В РФЯЦ-ВНИИЭФ таким способом подключаются библиотеки УРС-ОФ и ЕФР.

Если для подключения библиотек используются не файлы исходного кода, а, например, двоичные файлы *.mod, то используются специаль-

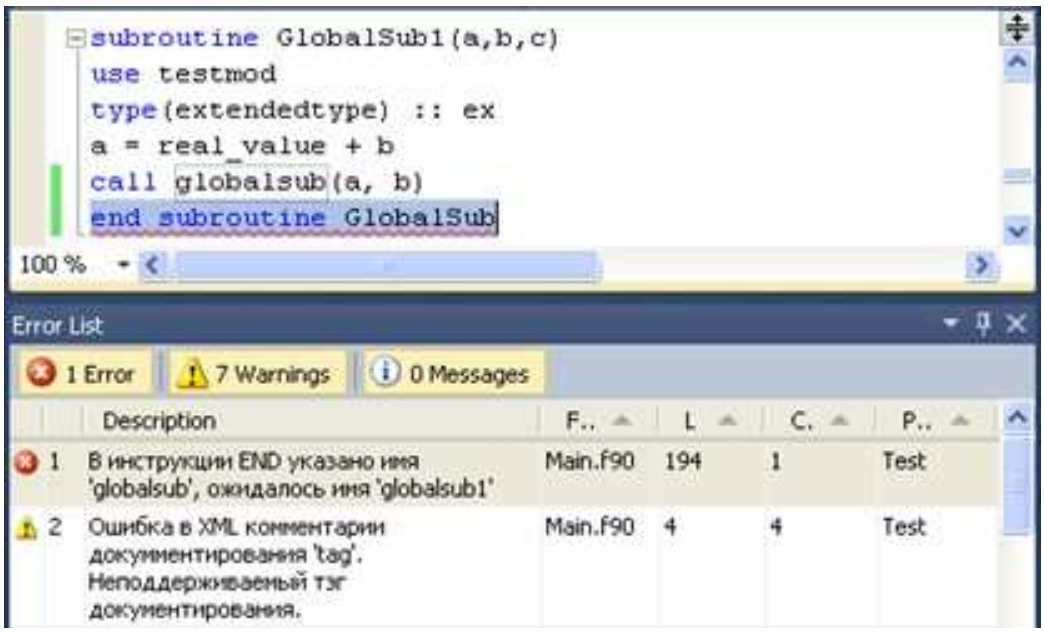

Рис. 12. Поддержка работы с панелью *Список ошибок*

ные XML-файлы, в которых содержатся описания необходимых программных элементов: модулей, подпрограмм, функций, типов данных и т. п. Заметим, что для описания каждой библиотеки используются два файла: первый содержит описание прикладных программных интерфейсов библиотеки в специально разработанном автором формате Fortran API, второй — документацию к элементам библиотеки. Реализована возможность по автоматическому созданию таких файлов для выбранного программного проекта. Еще одна задача, которая может быть решена с использованием данных файлов, — это генерация документации пользователя или разработчика библиотеки с использованием специализированных программ автоматической генерации документации, например Sandcastle [8].

Примером подключения библиотеки с использованием XML-файлов являются встроенные в язык подпрограммы и функции. На рис. 13

```
<!--Описание интерфейса функции-->
<api id="M:ACHAR(I, [KIND])">
 <!—Указываем, данные об элементе-->
 <apidata name="ACHAR" group="method" subgroup="function" />
 <!--указываем что это элементарный метод-->
 <methoddata elemental="true"/>
 <!--Указываем, что это функция-->
 <functiondata />
 <!--Описание аргументов функции-->
 <parameters>
  <!--Имя, использование - только входной-->
  <parameter name="I" intent="in">
   <!--Указываем, что это переменная-->
   <variabledata/>
   <!--Указываем тип данных. Целочисленный, с размерностью по умолчанию-->
   <type api="T:integer"/>
  </parameter>
  <!--Имя, использование - только входной, необязательный-->
  <parameter name="KIND" intent="in" optional="true">
   <!--Указываем, что это переменная-->
   <variabledata/>
   <!--Указываем тип данных. Целочисленный, с размерностью по умолчанию-->
   <type api="T:integer"/>
  </parameter>
 </parameters>
 <!--Указываем имя результирующего значения-->
 <result name="ACHAR">
  <!--Указываем, что это переменная-->
  <variabledata/>
  <!--Указываем тип данных возвращаемого значения. Символьный, единичной
длинны, с размерностью KIND, либо по умолчанию-->
  <type api="T:character(len=1,kind=KIND)"/>
 </result>
 <!--Указываем где содержится элемент, т.е. обратную связь-->
 <containers>
  <!--Имя библиотеки-->
  <library assembly="intrinsics"/>
  <!—Указываем имя родительского элемента - глобальная область видимости-->
  <element api="N:"/>
 </containers>
</api>
```

Рис. 13. Пример API-описания для функции ACHAR

представлен пример API-описания для встроенной функции ACHAR, а на рис. 14 — ее документации.

Расширенная поддержка библиотек конечного пользователя в части визуального выделения элементов библиотек демонстрируется на примере библиотеки УРС-ОФ (рис. 15), который показывает, что данная возможность позволяет визуально акцентировать внимание пользователя на наиболее важных элементах библиотеки.

Аналогично осуществляется поддержка библиотеки ЕФР. На рис. 16 приведен пример возможности получения информации об аргументах процедуры для перегруженной подпрограммы библиотеки ЕФР.

Сравнение возможностей FRIS и других языковых сервисов

В таблице сравниваются возможности FRIS и языковых сервисов от Intel и PGI.

Необходимо отметить, что языковой сервис FRIS не только осуществляет полную поддержку технологии IntelliSense, но и предоставляет существенно больше возможностей по расширению данной поддержки. Например, он позволяет пользователю документировать элементы своей программы и получать затем эти сведения во всех видах контекстной помощи. Немаловажна также возможность работы с произвольными библиотеками конечного пользователя. Благодаря разработанной модели Fortran API и ком-

```
<?xml version="1.0" encoding="utf-8" ?>
<doc>
  <!--Указываем все элементы, для которых есть документация-->
  <members>
    <!--Документация для одного элемента-->
    <!--13.7.2 ACHAR (I [, KIND])-->
    <member name="ACHAR(I, [KIND])">
      <summary>
        Возвращает символ, находящийся в указанной позиции в схеме упоря-
дочения ASCII. Это инверсия значения функции IACHAR.
      </summary>
      <param name="I">
        Номер позиции, типа integer
      </param>
      <param name="KIND">
        Необязательный параметр, размерность типа возвращаемого значения
      </param>
      <result>Символ character(len=1,kind=[kind])</result>
    </member>
  </members>
</doc>
```

Рис. 14. Пример документации к функции ACHAR

```
type(UrsOfData) ofdata
call ReleaseUrsOf(ofdata,ko,kan)
```

```
type(UrsOfData) ofdata
call ReleaseUrsOf(ofdata,ko,kan)
```

Рис. 15. Подсветка синтаксиса УРС-ОФ (сверху отсутствует, снизу присутствует)

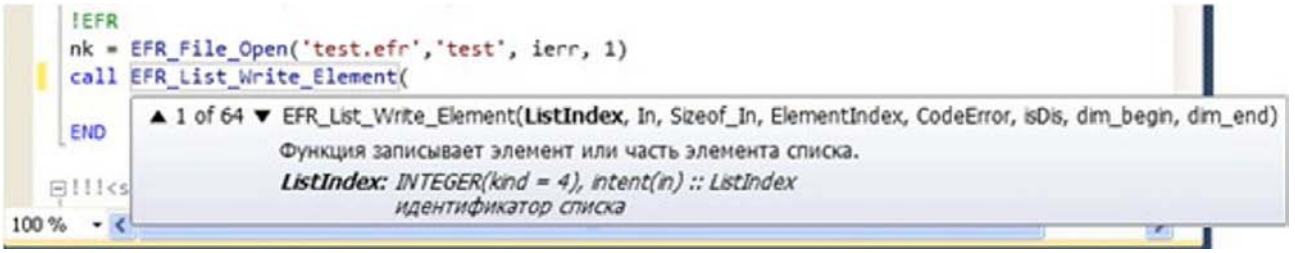

Рис. 16. Работа с перегруженными подпрограммами ЕФР

Сравнение возможностей FRIS и языковых сервисов от Intel и PGI

Возможность	Intel	PGI	FRIS
Построение списка членов типов данных	Отсутствует	Отсутствует	Есть
Отображение сведений о параметрах подпрограмм и функций	Есть, за исключением методов, связанных с типом данных, и перегруженных методов с использованием родового интерфейса	Только для встроенных подпрограмм и функций	Есть
Отображение кратких сведений об элементе языка программирования	Есть, за исключением полей и методов производных типов данных	Только для встроенных подпрограмм и функций	Есть
Автодополнение	Для имен модулей и подпрограмм, и функций	Только для ключевых слов	Есть
Поддержка сниппетов исходного кода	Есть, но только как команды меню	Нет	Есть. Сниппеты во FRIS включены в списки автодополнения
Поддержка комментариев документирования	Нет	Нет	Есть
Поддержка библиотек конечного пользователя	Нет	Нет	Есть

ментариям документирования можно получать оперативную помощь по функциям библиотеки, даже если тексты библиотеки недоступны.

Заключение

Создан языковой сервис FRIS, который предназначен для повышения производительности разработки Fortran-приложений. Реализованная во FRIS расширенная поддержка XML-комментариев документирования позволяет получать контекстно-зависимую помощь об используемых элементах языка программирования без обращения к специализированным справочным системам. С помощью механизма работы со сниппетами исходного кода пользователь может создавать и повторно использовать полноценные библиотеки шаблонов кода, а включение сниппетов в списки автодополения обеспечивает мгновенный доступ к ним.

Особым видом адаптации к требованиям конечного пользователя является обеспечение расширенной поддержки пользовательских библиотек программ. В частности, FRIS может предоставлять возможности визуального выделения тех или иных элементов в файле исходного кода для акцентирования внимания пользователя. Для этого реализовано два универсальных под-

хода: первый из них заключается в автоматической обработке файлов исходного кода на языке Fortran, второй — в создании специальных XML-файлов с описанием программных компонентов, экспортируемых библиотекой.

Разработанная модель прикладных программных интерфейсов Fortran-элементов (Fortran API) вместе с комментариями документирования, сохраняемыми в виде двух XML-файлов, позволяют в дальнейшем задействовать программы автоматической генерации документации пользователя или программиста библиотеки программ.

На основе средств расширенной поддержки библиотек реализована "интеллектуальная" поддержка использования библиотек УРС-ОФ и ЕФР, а в будущем планируется поддержка средств параллельного программирования MPI3.0 и OpenMP4.0.

Использование FRIS при разработке Fortran-приложений повышает производительность труда за счет предоставления различных видов контекстной помощи. Ранее Fortran-программист был вынужден знать наизусть имена всех используемых им функций, модулей, переменных и типов данных либо проводить полнотекстный поиск для их уточнения. Теперь в этом нет необходимости. Использование FRIS особенно полезно при командной работе над одним проектом.

Список литературы

1. Information technology. Programming languages. Fortran. Part 1: Base Language. ISO/IEC 1539-1:2004. ISO Publications Department.
2. Intel® Fortran Composer XE 2013 SP1 Release Notes. http://software.intel.com/enus/articles/intel-fortran-composer-xe-2013-sp1-release-notes.
3. PGI Visual Fortran. http://www.pgroup.com/products/pvf.htm.
4. Using IntelliSense. http://msdn.microsoft.com/enus/library/hcw1s69b(v=vs.110).aspx.
5. *Гордеев Д. Г., Голубкова Е. Ф., Гударенко Л. Ф. и др.* Современное состояние пакета программ УРС-ОФ для расчета термодинамических и механических свойств веществ // 12 Межд. конф. "Супервычисления и математическое моделирование". г. Саров, 11—15 ноября 2010 г.
6. *Волгин А. В., Тарасов В. И., Красов А. В., Кузнецов М. Ю.* Библиотека ЕФР для универсального представления расчетных данных // Труды РФЯЦ-ВНИИЭФ. Вып. 11. Саров: РФЯЦ-ВНИИЭФ, 2007. С. 130—135.
7. *Parr T.* Language Implementation Patterns. The Pragmatic Bookshelf, Raleigh, NC and Dallas, TX, 2009.
8. Eric Woodruff's Sandcastle Help File Builder Documentation. http://ewsoftware.github.io/SHFB/html/bd1ddb51-1c4f-434f-bb1ace2135d3a909.htm.

Статья поступила в редакцию 26.02.15.

FRIS LANGUAGE SERVICE FOR THE EFFECTIVE DEVELOPMENT OF FORTRAN APPLICATIONS: A REVIEW OF CAPABILITIES I. S. Ratkevich (FSUE "RFNC-VNIIEF", Sarov, Nizhny Novgorod region).

The FRIS language service capabilities providing an extended support for the effective development of Fortran applications in Microsoft Visual Studio are discussed. The FRIS capabilities are compared with those of the most commonly used analogs by Intel and PGI.

Keywords: FRIS, Fortran Intelligent Solutions, Visual Studio extension, language service, Fortran-2003.