DOI: 10.15514/ISPRAS-2023-35(2)-7

Программный комплекс SIO для работы со структурированными данными

А.О. Игнатьев, ORCID: 0000-0003-4902-2123 <a.o.ignatyev@vniitf.ru>
С.Ю. Мокшин, ORCID: 0000-0002-7454-6597 <s.yu.mokshin@vniitf.ru>

Всероссийский НИИ технической физики имени академика Е.И. Забабахина,
456770, Россия, г. Снежинск, Челябинская область, ул. Васильева, 13

Аннотация. Решение задач численного моделирования различных физических процессов предполагает использование вычислительных ресурсов на разных стадиях подготовки, проведения расчетов и обработки их результатов. При этом существует проблема передачи данных между различными прикладными программными комплексами, используемыми в том числе на гетерогенных вычислительных ресурсах с разной архитектурой. В статье рассматриваются основные подходы по разработке и использованию программного обеспечения для работы со структурированными данными прикладных программных комплексов на примере HDF и SIO.

Ключевые слова: вычислительная система; структурированные данные; высокопроизводительные вычисления; математическое моделирование; форматы данных; HDF; SIO

Для цитирования: Игнатьев А.О., Мокшин С.Ю. Программный комплекс SIO для работы со структурированными данными. Труды ИСП РАН, том 35, вып. 2, 2023 г., стр. 101-110. DOI: 10.15514/ISPRAS–2023–35(2)–7

SIO program suite for structured data processing

A.O. Ignatyev, ORCID: 0000-0003-4902-2123 <a.o.ignatyev@vniitf.ru>
S.Yu. Mokshin, ORCID: 0000-0002-7454-6597 <s.yu.mokshin@vniitf.ru>

E. I. Zababakhin All-Russian Scientific Research Institute of Technical Physics,
13, Vasilieva street, Chelyabinsk region, Snezhinsk, 456770, Russia

Abstract. Numerical modeling of various physical processes involves the use of computing resources at different stages of preparation, calculations and processing of their results. At the same time, there is a problem of data transfer between various application software used on heterogeneous computing resources with different architectures. The main approaches to the development and use of software for working with structured data of application software on the example of HDF and SIO discussed in this article.

Keywords: computing system; structured data; high-performance computing simulation; mathematical simulation subsystem; data format; HDF; SIO

For citation: Ignatyev A.O., Mokshin S.Yu. SIO program suite for structured data processing. Trudy ISP RAN/Proc. ISP RAS, vol. 35, issue 2, 2023. pp. 101-110 (in Russian). DOI: 10.15514/ISPRAS-2023-35(2)-7

1.Введение

Современные подходы по организации работы с генерируемыми на вычислительных системах и персональных компьютерах наборами структурированных данных предполагают использование некоторых, стандартизованных в мировом сообществе разработчиков

прикладного программного обеспечения, библиотек ввода-вывода. На данный момент основными требованиями к формату хранения данных можно назвать следующие:

- возможность хранения именованных данных разного типа и размера,
- возможность использования данных без дополнительного преобразования на вычислительных системах с гетерогенной аппаратной платформой и различными операционными системами.

Рассмотрим подходы к хранению данных с точки зрения удовлетворения данных требований.

2. Формат HDF

Безусловным лидером среди открытых проектов в области разработки стандартизованного формата хранения данных, удовлетворяющего вышеназванным требованиям, является проект HDF [1].

HDF – это многообъектный файловый формат с самоописанием, используемый для хранения и распространения научных данных. HDF был создан Национальным центром суперкомпьютерных приложений (NCSA) для удовлетворения потребностей различных групп ученых в различных инженерных проектах. Формат HDF обладает следующими свойствами:

- самоописание: для каждого объекта данных в файле HDF есть исчерпывающая информация (метаданные) о данных;
- универсальность: многие типы данных могут быть встроены в файл HDF;
- гибкость: HDF позволяет пользователям группировать связанные объекты данных вместе, помещать их в иерархическую структуру и добавлять описания и теги к объектам данных, позволяет пользователям помещать научные данные в несколько файлов;
- расширяемость: HDF может легко приспособиться к новым добавленным режимам данных в будущем и легко совместим с другими стандартными форматами;
- кроссплатформенность: HDF - это независимый от платформы формат файла (файлы HDF можно использовать на разных программно-аппаратных платформах без преобразования).

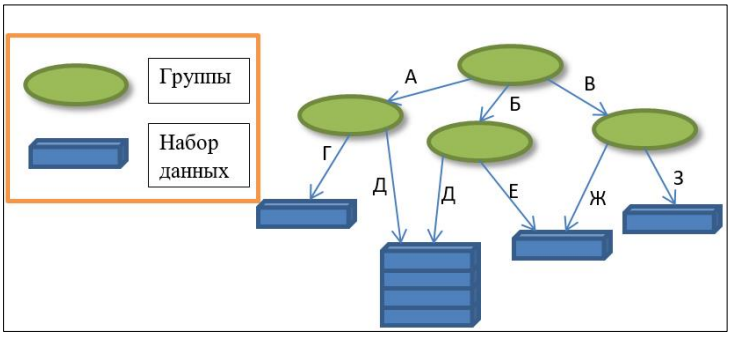

Рис. 1. Структура HDF-контейнера
Fig.1. HDF container structure

На данный момент разработчиками формата поддерживается версия HDF5. Для лучшего понимания логики использования HDF рассмотрим структуру файла HDF5. Формально файл HDF5 – это контейнер для хранения различных научных данных с помощью двух основных объектов данных – групп и наборов данных, как показано на рис. 1.

При этом набор данных HDF5 представляет собой многомерный массив элементов данных и некоторых вспомогательных метаданных, как показано на рисунке 2.

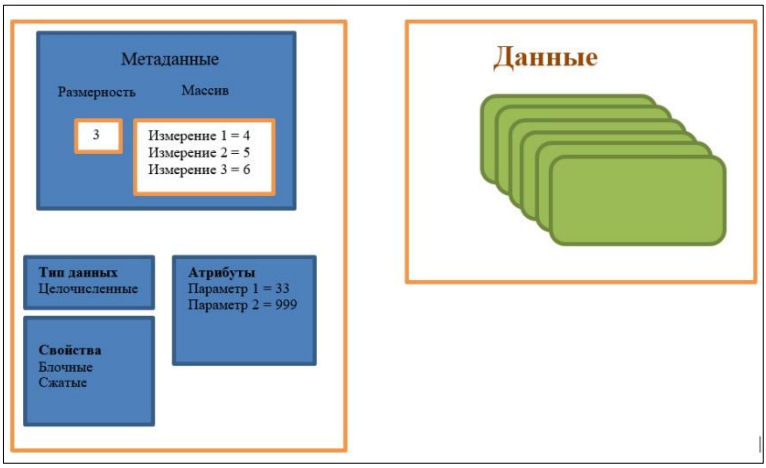

Рис. 2. Структура некоторого набора данных HDF
Fig.2. HDF dataset structure

В примере, приведенном на рис. 2, данные сохраняются, как трехмерный целочисленный массив размерностью 4x5x6. Он содержит некоторый набор атрибутов – параметров. Данные в массиве разделены на блоки и сжаты.

При всех преимуществах, такой подход обладает определенными недостатками:

- изначальная ориентация на максимальную универсализацию (хранение произвольных типов данных, допустимость сложной иерархической структуры данных) привела к излишнему усложнению как формата данных, так и интерфейса работы с данными,
- стремление к универсализации привело к многократному пересмотру спецификации HDF (за последние 20 лет были выпущены спецификации HDF3, HDF4, HDF5), приведшие к тому, что данные, записанные в старом формате, не читаются при использовании нового формата,
- поддержка новых форматов данных предъявляет специфические требования к системному окружению, в котором работает HDF, что, зачастую приводит к невозможности установки старых версий библиотек HDF на современные операционные системы.

Кроме того, многолетний опыт эксплуатации HDF в Российском Федеральном Ядерном Центре – Всероссийском научно-исследовательском институте технической физики имени академика Е.И. Забабахина (РФЯЦ-ВНИИТФ) показал постоянное наличие ошибок в данном программном обеспечении, в том числе влияющих как на информационную безопасность, так, иногда и на точность вычислений при важных научных расчетах. Эти ошибки, безусловно, оперативно выявляются и исправляются открытым сообществом разработчиков, но, в то же время, постоянно появляются новые.

3. Программный комплекс SIO

В то же время, в РФЯЦ-ВНИИТФ в 1998 году был разработан и внедрен в эксплуатацию программный комплекс SIO [2], полностью удовлетворяющий приведенным ранее требованиям к формату сохранения данных. Данный программный комплекс успешно применялся для хранения данных, совместно используемых на ЭВМ «Эльбрус-2» и другом вычислительном оборудовании вычислительного центра РФЯЦ-ВНИИТФ, а также фактически на всех ЭВМ с архитектурой x86-64. На данный момент программный комплекс SIO успешно развивается и интегрируется в прикладное программное обеспечение разработки РФЯЦ-ВНИИТФ.

SIO предназначен для хранения именованных объектов следующего вида:

- именованные записи (одномерные массивы) типа integer*2, integer*4, integer*8, real*4, real*8 и char произвольной длины;
- многомерные массивы типа integer*2, integer*4, integer*8, real*4, real*8;
- номерованные структуры (имя структуры и номер экземпляра), где элементом структуры (полем) являются те же типы данных.

Программный комплекс SIO состоит из библиотеки sio и программы для просмотра файлов формата SIO – sioview. Библиотека sio содержит функции для работы с файлами и хранимыми объектами. Интерфейс sio состоит из следующих основных функций:

- работа с файлами (открытие и закрытие, вывод элементов каталога);
- работа с именованными записями (чтение, запись);
- работа со структурами (описание полей, чтение/запись поля экземпляра группы);
- работа с многомерными массивами (определение массива, чтение/запись массива).

Рассмотрим основные особенности работы с этими типами данных.

3.1 Хранение данных

Программный комплекс SIO предназначен для хранения большого объема структурированных данных и не имеет ограничений как по размеру хранимых данных, так и по их количеству. Хранение структурированных данных требует ведение каталога, в котором для каждого объекта присутствует имя объекта, его атрибуты (тип, формат и количество элементов) и адрес в файле. Каталог в SIO имеет сегментно-списочную структуру, т.е. состоит из сегментов фиксированной длины, объединенных списком. Каждый сегмент состоит из описания блоков данных и, при необходимости, включает в себя описание следующего сегмента каталога, как показано на рис. 3.

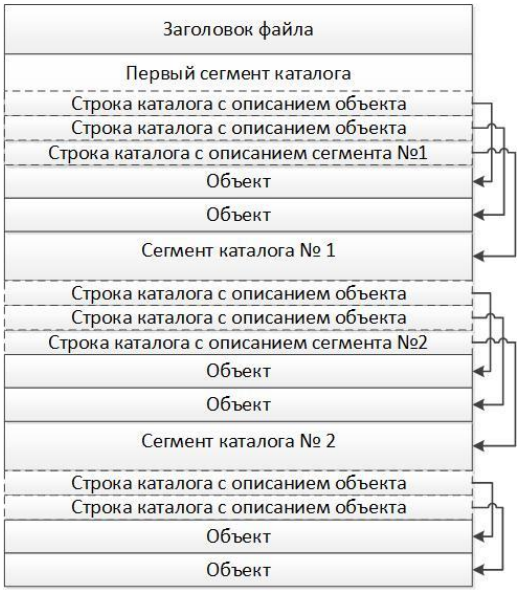

Рис. 3. Структура файла SIO
Fig. 3. SIO file structure

При создании файла в формате SIO создается первый сегмент каталога. При добавлении нового объекта в файл сначала проверяется, есть ли место в текущем сегменте. Если сегмент заполнен, то выделяется новый сегмент (ссылка на который включается в текущий сегмент)

и новый сегмент объявляется текущим. После этого новый объект включается в текущий сегмент каталога.

Набор функций для работы с каталогом включает в себя:

- получить количество объектов в каталоге файла,
- получить описание объекта из каталога по его порядковому номеру,
- получить описание объекта по имени.

Таким образом, SIO, как и HDF, является самоописываемым форматом хранения данных.

3.2 Именованные записи

Именованная запись представляет собой одномерный массив, имеющий имя, количество и формат элементов.

Набор функций для работы с именованными записями включает в себя:

- запись,
- чтение.

В параметрах этих функций указывается имя, формат элементов, количество элементов и, возможно, смещение относительно начала именованной записи в файле, что позволяет работать с именованной записью, как с подмассивом.

3.3 Структуры

Структуры предназначены для хранения множества экземпляров неоднородных элементов. Каждая структура имеет набор полей, идентичных по смыслу именованным записям.

Набор функций для работы с структурами включает в себя:

- определение структуры,
- описание полей структуры,
- запись экземпляра поля структуры,
- чтение экземпляра поля структуры.

При определении структуры указывается её имя и число полей в ней. Далее, для каждой структуры указывается имя поля, формат и количество элементов. Для доступа к объекту структуры в операциях чтения/записи указывается номер экземпляра структуры и имя поля.

3.4 Многомерные массивы

В последних версиях SIO появилась поддержка многомерных массивов. При этом размерность массива в SIO не ограничена ничем, кроме объема доступной памяти. В программном комплексе SIO реализованы функции, позволяющие:

- определить многомерный массив,
- получить информацию о размерности массива и длинах его по измерениям,
- записывать/читать массив как целиком, так и частями (подмассивами).

Необходимость работы с подмассивами связана с традиционной необходимостью обработки данных значительного объема на высокопроизводительных вычислительных системах, в этом случае часто возникает необходимость доступа только к части обрабатываемых данных. Это может быть связано как с недостаточностью ресурсов для обработки данных целиком, так и с распараллеливанием обработки данных для сокращения времени этой обработки.

Зачастую вместо понятия «подмассив» используется термин «вырезка» (slice). Под вырезкой из многомерного массива в общем случае понимается получение нового массива размерности той же или меньшей, элементы которого берутся из части исходного массива с отсеканием «ненужных» частей. Пример вырезки из двумерного массива показан на рис. 4.

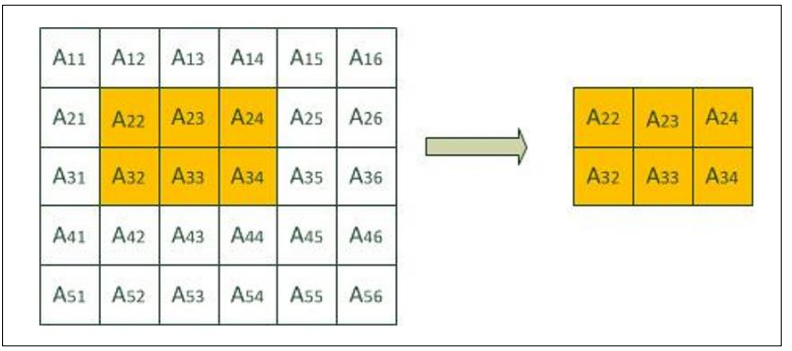

Рис. 4. Вырезка из двумерного массива
Fig.4. Slice from two-dimensional array

В SIO массивы в файле хранятся в виде одномерной последовательности элементов с построчной разверткой в стиле языка C (на рисунке 5 показан порядок хранения элементов для массива, приведенного на рис. 4).

Рис. 5. Порядок хранения элементов двумерного массива
Fig.5. The order of storing elements of a two-dimensional array

Выбранная вырезка определяется тремя параметрами, первый из которых определяет начальный индекс вырезки в массиве, второй описывает размерность вырезки, а третий – количество элементов по каждому измерению массива, составляющих вырезку.

В общем случае, для выполнения вырезки из массива требуется либо прочитать весь массив в оперативную память с последующей пересылкой элемента из исходного массива в массив для вырезки в программе, либо несколько раз производить процедуру чтения из файла требуемых частей массива и, опять же, пересылать полученные элементы в результирующий массив, увеличивая накладные расходы на выполнение операции. Поэтому предпочтительно использовать в качестве вырезок построчные сечения.

Сечение отличается от вырезки тем, что начальные и конечные значения индексов в сечении совпадает с теми же значениям в исходном массиве. При этом размерность сечения становится меньше, чем у исходного массива. Построчное сечение определяется фиксированным первым индексом массива. Таким образом, информация состоит из одной строки или набора смежных по памяти строк массива.

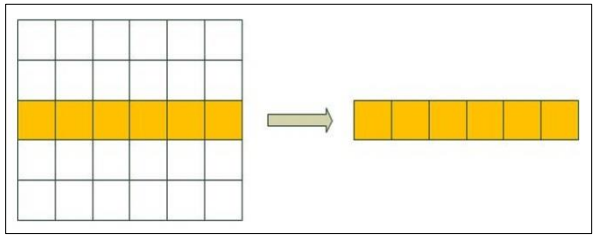

Рис. 6. Одномерное сечение двумерного массива
Fig.6. One-dimensional cross section of a two-dimensional array

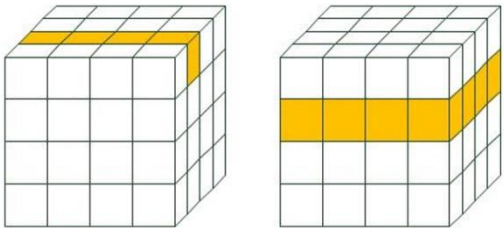

Рис. 7. Одно- и двумерное сечения трехмерного массива
Fig.7. One- and two-dimensional sections of a three-dimensional array

На рис. 6 показано одномерное сечение двумерного массива. На рис. 7 показаны одномерное и двумерное сечения трехмерных массивов. Выбором подходящего сечения можно добиться сокращения числа чтений из файла до одного и исключения необходимости перемещения информации по оперативной памяти.

Для двумерных массивов A_{ij} таким сечением является строка при фиксированном значении i. Для трехмерных массивов A_{ijk} таких сечений два типа:

- строка при фиксированных значениях i и j (одномерное сечение);
- плоскость при фиксированном значении i (двумерное сечение).

На рис. 8 приведены примеры хранения трехмерного массива с длинами по измерениям, равными двум, и выбираемые для двух сечений элементы массива.

Рис. 8. Порядок хранения элементов трехмерного массива и выбираемого сечения
Fig.8. The elements order of a three-dimensional array and its selected slice

3.5 Пример работы с библиотекой sio

Рассмотрим приведенный на листинге 1 пример исходных текстов программы, которая сначала заполняет массив по колонкам, а потом читает по строкам.

```
 1 #define SIO_REAL16 18 // формат элементов
 2  const int lcat = 8;
 3  int ko, i, j;
 4  char* aname="d2_2";  // имя массива
 5  int idx2[2];         // массив для индексов вырезки
 6  int lens2[2];        // массив для длин по измерениям вырезки
 7  double column[L1];   // массив для колонки
 8  double line[L2];     // массив для строки
 9
10  ko = siofop(fname,0,STNEW,lcat,0,&hf); // создание нового файла
11  ko = sioadef(hf, aname, 2, dims2, SIO_REAL16);// создание 2D массива
12
13  for (j = 0; j < L2; j++) {   // цикл по колонкам
14    for (i = 0; i < L1; i++) { // формирование элементов колонки
15      column[i] = - (double)(i * LM + j);
16    }
17    idx2[0]  =  0; idx2[1]  = j;// начальный индекс для колонки
18    lens2[0] = L1; lens2[1] = 1;// длины по измерением колонки
19    ko = sioawrs2(hf, aname, 1, idx2, lens2, column);// запись колонки
20  }
```

```
21
22  ko = siofcl(hf,0);
23
24  ko = siofop(fname,0,STOLD,lcat,0,&hf); // открытие файла
25
26  for (i = 0; i < L1; i++) {
27    idx2[0]  =  i; idx2[1]  = 0; // индекс строки
28    lens2[0] =  1; lens2[1] = L2;// длины по измерениям строки
29    ko = sioards2(hf, aname, 1, idx2, lens2, line); // чтение строки
30    printf("I=%2d: %7.2f %7.2f .. %7.2f %7.2f\n", i, line[0],
line[1],line[L2-2], line[L2-1]); // проверка содержимого строки
31  }
32
33  ko = siofcl(hf,0);
```

Листинг 1. Пример исходного текста программы
Listing 1. Program source code example

В строках 1-8 приводится описание требуемых для работы констант и переменных. Строка 10 содержит обращение к функции создания нового файла. Строка 11 содержит обращение к функции описания двумерного массива. В строках 13-20 производится поколоночная запись в массив сформированной информации. Строка 22 содержит обращение к функции закрытия файла. Строка 24 содержит обращение к функции открытия этого же файла на чтение. В строках 26-31 производится построчная выборка из массива и печать выбранных значений. Строка 33 содержит обращение к функции закрытия файла.

3.6 Дополнительные инструменты SIO

Для упрощения процедуры перехода с формата HDF на формат SIO специалистами РФЯЦ-ВНИИТФ был разработан преобразователь данных hdf2sio, который из указанного HDF файла выбирает необходимый набор данных и дописывает его в создаваемый файл SIO.

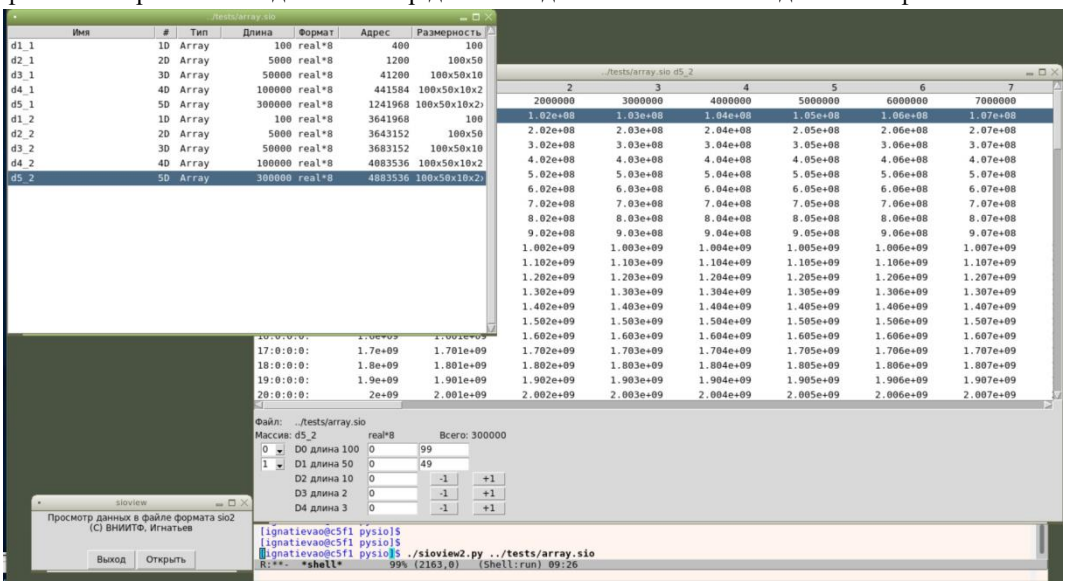

Рис. 9. Просмотр каталога файла и выбранных массивов
Fig.9. File directory and selected arrays viewing

Кроме того, для интерактивного просмотра содержимого файла в формате SIO реализована графическая утилита sioview, входящая в состав программного комплекса SIO и написанная на языке программирования Python [4] с использованием графической библиотеки Tk [5]. Эта

утилита открывает указанный в параметре или выбранный в диалоге файл, выдает его каталог и, после выбора именованной записи, в отдельном окне выдается содержимое записи в выбранном диапазоне.

На рис. 9 показан вывод каталога выбранного файла и выбранного массива. Поля ввода «от» и «длиной» определяют область выводимой для записи информации. Одномерный массив *M(I)* выдается в режиме построчной развертки. Двумерный массив *M(I,J)* выдается в виде таблицы, где строки соответствуют значению индекса *I*, а по столбцам выдается значение *J*. Трехмерный массив *M(I,J,K)* выдается в виде таблицы для выбранного значения индекса *I*, строки соответствуют значению *J*, а по столбцам выдается значение индекса *K*. Кнопки навигации позволяют изменять начальные индексы по измерениям.

4. Заключение

Рассмотренный в данной статье программный комплекс SIO обеспечивает выполнение сформулированных ранее требований к формату хранения данных. Так же, как и HDF, SIO является самоописываемым форматом хранения. Реализация SIO обеспечивает возможность работы с данными, полученными на разных вычислительных платформах и под различными операционными системами. SIO не обладает рядом возможностей HDF (иерархия и многообразие хранимых объектов), поэтому программный код SIO существенно проще и не требует при своей установке и использовании дополнительных программных продуктов. Это существенно облегчает его использование на различных вычислительных платформах и операционных системах.

Программный комплекс SIO успешно используется в РФЯЦ-ВНИИТФ уже более 24 лет, последняя версия SIO включена в реестр отечественного программного обеспечения РФ. Авторы выражают надежду, что изложенная в статье информация о структуре и функциональных возможностях SIO окажется полезной другим специалистам, занимающимся разработкой прикладного программного обеспечения в РФ.

Список литературы / References

[1] HDF Group. Available at: https://www.hdfgroup.org/solutions/hdf5/, accessed 10.04.2023.
[2] Программный комплекс SIO. / SIO program suite. Available at: http://vniitf.ru/article/programmniy-kompleks-sio, accessed 10.04.2023.
[3] GNU Bash. Available at: https://www.gnu.org/software/bash/, accessed 10.04.2023.
[4] Python. Available at: https://www.python.org/, accessed 10.04.2023.
[5] Tk Library. Available at: https://docs.python.org/3/library/tk.html, accessed 10.04.2023.

Информация об авторах / Information about authors

Алексей Олегович ИГНАТЬЕВ – начальник лаборатории. Сфера научных интересов: проектирование вычислительных систем, разработка параллельных программ численного моделирования, разработка операционных систем, методы и средства защиты информации.

Alexey Olegovich IGNATYEV – Head of Laboratory. Research interests: design of supercomputer systems, parallel numerical simulation programs development, operating systems development, methods and means of information security.

Сергей Юрьевич МОКШИН – начальник отдела. Сфера научных интересов: проектирование вычислительных систем, разработка функциональных подсистем для высокопроизводительных вычислительных систем, разработка операционных систем, методы и средства защиты информации.

Sergey Yurievich MOKSHIN – Head of Department. Research interests: design of supercomputer systems, development of functional subsystems for high performance supercomputing systems, operating systems development, methods and means for protecting information.