Разработка концепции новой системы мультифизического моделирования для решения задач гемодинамики*

Г.В. Копытов

Балтийский федеральный университет им. И. Канта, Институт вычислительной математики РАН

Рассматривается архитектура новой программной платформы мультифизического моделирования для решения задач гемодинамики. Предлагаемый подход использует принцип раздельного выполнения кодов моделей и сопряжения моделей через выделение областей связывания на сетках и передачи данных через сервер интеграции. Рассматриваются результаты совместного моделирования моделей 1D-0D-3D и дается оценка его эффективности.

Ключевые слова: мультифизическое моделирование, гемодинамика.

1. Введение

За последнюю декаду мультифизическое (междисциплинарное) моделирование становится быстро развивающимся трендом в области моделирования сложных систем и физических процессов. Оно позволяет разбивать общую задачу на ряд подзадач, каждая из которых описывается собственным набором уравнений и использует различные решатели и численные методы. Таким образом, удается совместно решать задачи газовой и гидродинамики и динамики твердого тела, объединять модели разной размерности и разной масштабности для моделирования гемодинамики и т.д. Существует много академических групп и коммерческих компаний, предлагающих различные библиотеки и платформы для решения задач мультифизического моделирования [1], некоторые из которых будут рассмотрены далее.

MpCCI (http://www.mpcci.de/mpcci-software.html) является коммерческим приложением для сопряжения мультидисциплинарных приложений и поддерживает интеграцию с большим числом коммерческих решателей (ANSYS, Abaqus, Star-CD, Fluent, FlowMaster), а также пакетов с открытым исходным кодом (OpenFOAM). Она построена на клиент-серверной архитектуре и имеет программный API для подключения решателей, разработанных третьими сторонами.

OpenPALM (http://www.cerfacs.fr/globc/PALM_WEB/) - программное обеспечение с открытым исходным кодом (с 2011 года), которое позволяет сопрягать как коммерческие решатели, так и собственные разработки. Это приложение имеет довольно сложный графический интерфейс для связывания приложений и позволяет динамически связывать несколько приложений. Связывание приложений происходит через зоны обмена разной размерности (1D, 2D и 3D) с возможностью интерполяции передаваемых данных.

MCT (http://www.mcs.anl.gov/research/projects/mct) – это разработка Арагонской национальной лаборатории и является программным обеспечением с открытым исходным кодом. Поддерживает программную модель, близкую к MPI и позволяет работать с неструктурированными сетками и планировщиками коммуникаций, обеспечивающие параллельные обмены между матрицами приложений MxN.

FEStudio [2] – это российская разработка для решения задач взаимодействия конструкции с жидкостью или газом. Программная модель представляет собой клиент-серверную архитектуру, использующую программное обеспечение ICE (https://zeroc.com). Сопрягаемые приложения могут работать на несогласованных сетках. Поддерживает итерационную процедуру согласования данных в пределах одного временного шага.

preCICE [3] – это программное обеспечение для решения задач взаимодействия твердого тела с жидкостью или газом. Может работать только на структурированных сетках. Для коммуникаций использует MPI.

* Работа поддержана грантом РНФ 14-31-00024.

FlowVision (https://flowvision.ru) – коммерческое программное обеспечение, разработанное компанией ТЕСИС, и предназначенное для решения задач взаимодействия жидкость-конструкция. В настоящее время обеспечивает сопряжение только программных продуктов Abaqus и FlowVision.

ПКСМ – Программный Комплекс Совместного Моделирования [4], разработанный в институте вычислительной математики РАН для совместного моделирования модели мирового океана ИВМ-ИО и модели глобальной атмосферы ПЛАВ. Эта система разрабатывалась с нуля для совместного выполнения моделей на сетках высокого разрешения. Отличительной особенностью этой системы является разработанный и реализованный алгоритм многоуровневой интерполяции данных для произвольных пар геофизических моделей на логически прямоугольных сетках. Он учитывает тот факт, то в задачах моделирования океан-атмосфера можно предварительно вычислить интерполяционную матрицу для заданных сеток, так как сетки в процессе моделирования не изменяются.

В данной работе представлена концепция новой платформы совместного моделирования, предназначенная для решения задач гемодинамики, как на последовательных, так и на массово-параллельных архитектурах, и позволяющая сопрягать модели, написанные на разных языках программирования (в настоящее время Фортран , Си и C++) и проводить моделирование в операционных системах Windows и Unix.

2. Программная модель разрабатываемой платформы

Разрабатываемая в Институте вычислительной математики РАН при поддержке гранта РНФ 14-31-00024 программная система является платформой для мультифизического моделирования, ориентированная в первую очередь на решение задач гемодинамики [5,6].

Типичная составная модель кровотока человека включает в себя одномерную модель кровотока 1D, модель сердца 0D и 3-х мерную модель, которая рассчитывает трехмерные уравнения Навье-Стокса движения крови в окрестностях установленного кава-фильтра, работу которой должна обеспечить разрабатываемая платформа.

В основе разрабатываемой платформы лежит интеграция разделенных кодов моделей, которые выполняются как отдельные процессы, и выполняют синхронизацию данных с помощью сервера интеграции, как показано на рисунке 1.

Программные коды моделей взаимодействуют с сервером интеграции по протоколу TCP/IP. Взаимодействие через MPI также рассматривалась как опция, однако было отвергнуто из-за возможной несовместимости различных версий MPI при интеграции моделей. Тем не менее, конкретная модель может использовать MPI как средство распараллеливания, как видно на рисунке 2, и при этом участвовать в общем процессе интеграции.

Код модели включает в себя библиотечный код, который содержит функции ядра (адаптера) интеграции. Адаптер в свою очередь состоит из менеджера синхронизации и драйвера данных. Каждая модель должна предоставить адаптеру набор функций по чтению и записи данных (драйвер данных). На рисунке 1 буквой А обозначен адаптер интеграции. Менеджер синхронизации и драйвер данных как составные части адаптера на рисунке не обозначены. В функции менеджера синхронизации входит взаимодействие с сервером интеграции (каплером), упаковка и распаковка данных при отправке и приеме сообщений, а также вызов функций драйвера данных. На рисунке 1 видно, что некоторые процессы кода могут не иметь адаптера интеграции (код А). В этом случае главный процесс кода выступает от имени всех подчиненных процессов при взаимодействии с сервером интеграции и отвечает за распределение данных по сеткам подчиненных процессов.

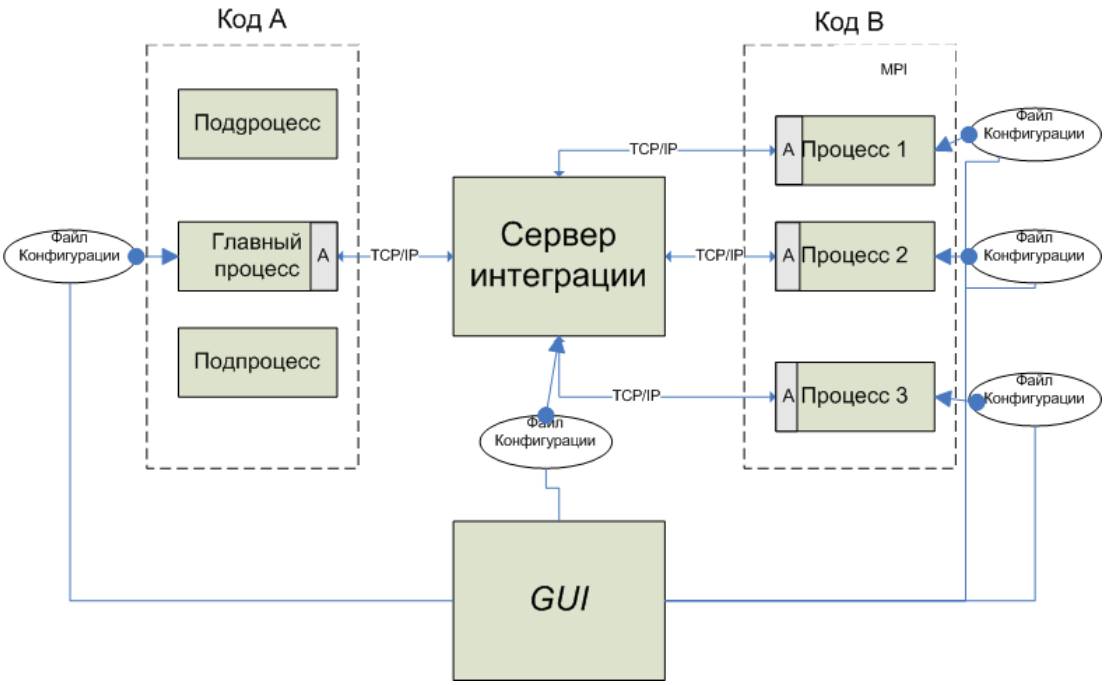

Рис. 1. Программная модель новой платформы

На рисунке 2 приведен API, который используют коды для работы в режиме совместного моделирования.

```
int  edml_init(int codeID, int pid);
int  edml_init_close();
void edml_config(DRIVER  *driver);
void edml_finalize();
int  edml_reached_syncpt(int id);
int  edml_def_mesh(int id);
int  edml_def_part(int mesh_id, int part_id);
int  edml_def_nodes(int mesh_id, int part_id, int globalDim, int nn, OID
nids[], double *coords);
struct DRIVER {
    void (*error)(int err);     // если функция не определена пользовате-
лем, то действует стандартная
    void (*setGlobalValue)(QID qid, void *value);
    void (*getGlobalValue)(QID qid, void *value);
    void (*setNodeValue)(int meshid, int partid, QID qid, OID oid, void
*value);
    void (*getNodeValue)(int meshid, int partid, QID qid, OID oid, void
*value);
    void (*setLineValue)(int meshid, int partid, QID qid, OID oid, void
*value);
    void (*getLineValue)(int meshid, int partid, QID qid, OID oid, void
*value);
    void (*setFaceValue)(int meshid, int partid, QID qid, OID oid, void
*value);
    void (*getFaceValue)(int meshid, int partid, QID qid, OID oid, void
*value);
    void (*setVolumeValue)(int meshid, int partid, QID qid, OID oid, void
*value);
    void (*getVolumeValue)(int meshid, int partid, QID qid, OID oid, void
*value);
};
```

Рис. 2. API для работы кодов в режиме совместного моделирования (Си интерфейс)

Для того, чтобы участвовать в совместном моделировании, каждая модель должна определить переменные, которыми она будет обмениваться с другими моделями и описать свои точки синхронизации. Это определение делается в общем конфигурационном файле, пример которого представлен рисунке 3.

```
code 1DTD
 T( no = 1, loc = global, dim = 1 );
 dT( no = 2, loc = global, dim = 1 )
 Smax( no = 3, loc = global, dim = 1 )
 Qin( no = 11, loc = global, dim = 1 )
 V(no = 4, loc = node, dim = 2);
end
code ETD
 T( no = 11, loc = global, dim = 1 );
 dT( no = 12, loc = global, dim = 1 )
 Smax( no = 13, loc = global, dim = 1 )
 V(no = 14, loc = node, dim = 2);
end
code Heart
 T( no = 31, loc = global, dim = 1 );
 Qout( no = 32, loc = global, dim = 1 )
end
matching
  1DTD.T = ETD.T;
  1DTD.dT = ETD.dT;
  1DTD.Smax = ETD.Smax;
  1DTD.V = ETD.V;
  1DTD.Qin = Heart.Qout
end
coupling
syncpt 1DTD( 1 ) : send( T ), recv( Qin );
syncpt Heart( 1 ) : recv( T );
syncpt Heart( 2 ) : send( Qout );
syncpt 1DTD( 2 ) : send( T, dT, Smax, V(7,2) );
syncpt 1DTD( 3 ) : recv( V(7,2) );
syncpt ETD( 2 ) : recv( T, dT, Smax, V(7,2) );
syncpt ETD( 3 ) : send( V(7,2) );
match_syncpt   1DTD( 1 ) = Heart( 1 ) = Heart(2);
match_syncpt   1DTD( 2 ) = ETD( 2 );
match_syncpt   1DTD( 3 ) = ETD( 3 );
global-matching
end
job 1DTD
# Windows
#bin=C:/Data/endovasc_last-2.0/blood.exe
# Linux
host=192.166.118.151
bin=/mnt/hgfs/Data/1DTD-2.0/main.exe
#args=4
end
job ETD
# Windows
#bin=C:/Data/endovasc_last-2.0/blood.exe
# Linux
host=192.166.118.151
bin=/mnt/hgfs/Data/ETD-2.0/main.exe
#args=4
#numproc=1
end
```

Рис. 3. Конфигурационный файл совместного моделирования 1D-0D-3D

В приведенном примере описаны коды одномерной модели кровотока (1DTD), трехмерной модели (ETD) и модели сердца (Heart). Код 1DTD определяет четыре глобальных скалярных переменных T – текущее время моделирования, dT – текущий шаг моделирования, Smax – рассчитываемый на каждом шаге параметр, определяющий соотношение шага по времени и по пространству в соответствии с правилом Куранта и Qin – объем крови, которое поступает в артериальное дерево на каждом шаге моделирования. Кроме того модель 1DTD определяет одну векторную сеточную переменную V, определенную в узлах сетки размерностью 2, которая является кортежем (s, u), где s - поперечное сечение сосуда, а u – скорость кровотока. Каждая переменная имеет атрибуты размерность ‘dim’ и ‘no’, являющимся номером переменной в данном коде. Атрибут ‘loc’ (расположение) может принимать значения ‘global’, означающее, что переменная определена не на сетке, а является глобальной, либо ‘node’, ‘line’, ‘face’ или ‘volume’, что означает, что переменная определена на соответствующем элементе сетки. Атрибуты переменной образуют уникальный идентификатор переменной QID, который передается в драйвер данных при операциях чтения и записи. Структура идентификатора переменной (QID –Quantity ID) и идентификатора объекта сетки (OID – Object ID) показана на рисунке 4.

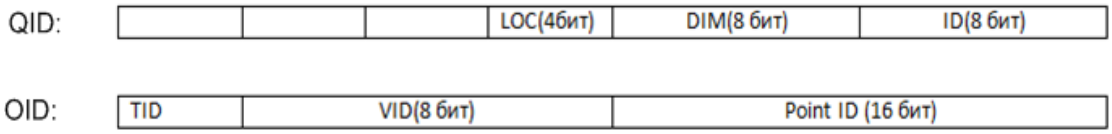

Рис. 4. Структура QID – уникального идентификатора переменной и OID – идентификатора узла 1D сетки

Секция ‘matching’ описывает соответствие переменных разных кодов. В секции ‘coupling’ присутствуют описания точек синхронизации моделей (‘syncpt’), с указанием переменных, которые будут посланы или приняты от других моделей. В приведенном примере модель 1DTD в точке синхронизации 2 посылает три глобальных переменных и одну векторную, определенную в области согласования (7,2), то есть на сетке с идентификатором 7 ее части 2. Ключевое слово ‘match_syncpt’ определяет соответствие точек синхронизации в разных кодах. Одна точка синхронизации в коде может соответствовать нескольким точкам синхронизации в одном или нескольких кодах. Единственным требованием при этом является полное соответствие посылаемых и принимаемых переменных в точке синхронизации. Если кто-то посылает переменную, то кто-то другой ее должен принимать. Сервер интеграции проверяет это условие при анализе конфигурационного файла.

Ключевое слово ‘global-matching’ означает, что коды моделей используют одинаковые идентификаторы элементов сетки и серверу интеграции не требуется вычислять геометрическое соответствие элементов сеток разных моделей. Это особенно удобно, если модели работают на одинаковых сетках, как в примере, приведенном ниже.

Секция ‘job’ описывает расположение и параметры запуска разделенных кодов. В этой секции может быть описано число запускаемых процессов данного кода (numproc, по умолчанию 1), хост, на котором будет выполняться модель, и путь до исполняемого файла.

Модель может иметь несколько сеток, каждая из которых имеет уникальный идентификатор. Каждая сетка может состоять из нескольких частей, также имеющих уникальный идентификатор. Пара {уникальный идентификатор сетки, уникальный идентификатор части} определяет конкретную область согласования.

Интеграция различных моделей происходит на основе выделения в сетках моделей областей согласования, обмен данными между которыми происходит при посредничестве сервера интеграции. Типы сеток и даже их размерность может быть совершенно разной в различных моделях. При этом возникает задача сопоставления сеток, то есть соответствия ячейки сетки одного кода ячейкам сетки другого кода, а также интерполяции данных при передаче их из одной сетки в другую.

Модели передают серверу интеграции описания своих областей согласования как множество узлов, имеющих координаты в пространстве, и множество элементов, объединяющих некоторое подмножество узлов и образующих ячейки сетки (одно-, двух- или трехмерной размерности). Узлы и ячейки могут иметь ассоциированные с ними данные, которыми обмениваются

модели. Данные могут быть как скалярного, так и векторного типа. Помимо данных на сетке, модели могут также иметь глобальные данные, например, время моделирования или шаг итерации, которыми они также могут обмениваться при взаимодействии.

Модификация кода для выполнения в режиме совместного моделирования выглядит следующим образом. Код модели сначала устанавливает соединение с сервером интеграции (edml_init), определяет одну или несколько областей согласования (edml_def_nodes) с перечислением элементов сетки, которые входят в данную область, сообщает адаптеру данных настройки драйвера данных (edml_config) и завершает фазу инициализации (edml_init_close).

Необходимость отдельного вызова для сообщения серверу о завершении фазы инициализации связана с тем, что любой код может определять произвольное число областей согласования и при этом он указывает, за какие элементы сетки в области согласования он отвечает. Это особенно важно, когда код запускается в режиме параллельного выполнения, когда разные части сетки в области согласования распределены по разным процессам. При этом прикладному программисту нет необходимости заботиться о том, каким процессам он должен послать свою часть сетки и тем самым нет необходимости знать, как сетка в области согласования распределена по процессам в чужом коде. Всю необходимую работу осуществляет сервер интеграции.

При достижении точки синхронизации код модели объявляет об этом адаптеру (edml_reached_syncpt) и тот в свою очередь собирает необходимые для передачи данные, вызывая функции драйвера данных, и передает их серверу интеграции, который в свою очередь организует обмен с другими кодами. Если в данной точке синхронизации данный процесс должен получить данные от нескольких процессов одного или разных кодов, то сервер интеграции будет дожидаться данных от всех процессов и, только собрав все данные, вернет их ожидающему процессу.

Взаимодействие с драйвером данных происходит с помощью набора функций чтения/записи глобальных переменных и сеточных переменных, как показано на рисунке 2. Параметры meshid и prated являются идентификаторами сетки и части области согласования. Уникальный идентификатор переменной QID содержит информацию о номере переменной, равный атрибуту ‘no’ из конфигурационного файла, размерность переменной и ее расположение ‘LOC’. Для сеточных переменных передается также уникальный идентификатор элемента сетки. В приведенном ниже примете он формируется из номера дерева сосудов ‘TID’(бронхиальное, и малые и большие артериальные и венозные деревья), номер сосуда ‘VID’ и номер точки (узла сетки 1D) в сосуде, как показано на рисунке 4.

3. Проведение совместного моделирования 1D0D3D

В качестве модели для совместного моделирования взята модель [5] кровеносной системы человека, состоящая из одного артериального и одного венозного дерева сосудов. В венозном дереве между первым и вторым сосудом стоит кава-фильтр, геометрия которого описывается трехмерной сеткой, которую использует в расчетах 3D модель [6]. Пораженных сосудов в данной модели нет. Расчетное модельное время составляет 1 секунду. Расчет 1D, 3D модели и модели сердца происходил на ОС Ubuntu 12.04.

Каждый сосуд кровеносного дерева в одномерной модели разбит на различное число точек, которые образуют узлы сетки. Модели 1D и 3D определяют область согласования (7,2), которая состоит из узлов сетки сосудов 2 и 1 венозного дерева, то есть входящих и выходящих из кава-фильтра. Одномерная модель рассчитывает значения в промежуточных узлах сетки и передает их значения в области согласования в точке синхронизации 2. Трехмерная модель рассчитывает движение крови в узле с кава-фильтром и передает модифицированные граничные условия в 1 и 2 сосуде назад в 1D модель.

Для сравнения результатов берутся значения скорости и сечения сосуда на выходе из кава фильтра при моделировании совместного кода и совместного моделирования разделенного кода.

Результаты моделирования представлены на рисунке 5. Как и ожидалось, в результате совместного моделирования разделенных кодов был получен тот же результат, что и при расчете модели с общим кодом.

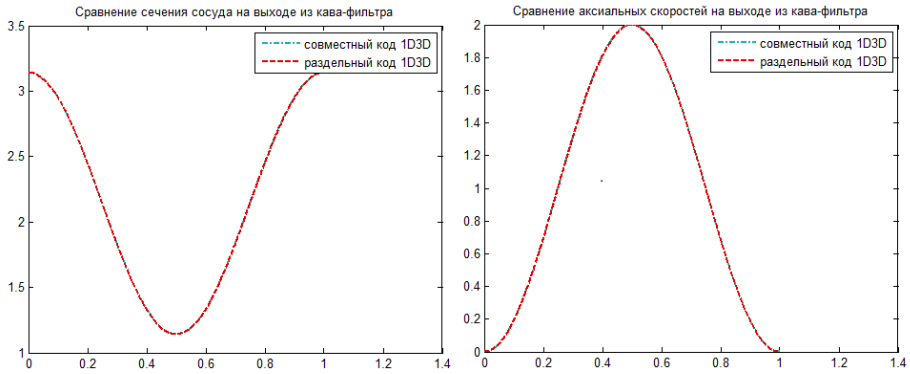

Рис. 4. Сравнительные графики сечения сосуда и аксиальной скорости при моделировании совместного кода и раздельных кодов

Была также проведена оценка эффективности работы модели в режиме совместного моделирования по сравнению с расчетом модели с совместным кодом, а также объем передаваемых между моделями данных. Результат сравнения представлен в таблице 1.

Таблица 1. Сравнение эффективности моделирования в режиме совместного и разделенного кодов

	Совместный код	Раздельный код
0,5 сек модельного времени	11 мин 42 сек	11 мин 50 сек
1,0 сек модельного времени	23 мин 15 сек	23 мин 31 сек
1,5 сек модельного времени	35 мин 03 сек	35 мин 22 сек

Суммарный объем переданных данных между кодами в режиме совместного моделирования на прогоне 1 секунда модельного времени составило 4.1 Мб, что, учитывая длительность расчета 23 минуты, составляет очень незначительную величину. Данные из таблицы 1 показывают, что накладные расходы на организацию совместного моделирования в режиме разделенных кодов не превышают 1% по сравнению с совместным кодом. Этот результат вполне объясним, так как в задачах гемодинамики не требуется осуществлять трудоемкую операцию интерполяции при переносе данных между сетками разных кодов.

Литература

1. O. Babur, T. Verhoeff, M. van den Brandt “Multiphysics and Multiscale Software Frameworks: An Annotated Bibliography”, ISSN 0926-4515, 2015.
2. И. М. Кузьмин, Л. Е. Тонков, С. П. Копысов “Алгоритмическое и программное обеспечение решения задач взаимодействия конструкции с жидкостью/газом на гибридных вычислительных системах», Компьютерные исследования и моделирование, 2013, Т.5, № 2, с. 153-164.
3. H. J. Bungartz, F. Lindnerb, B. Gatzhammer, M. Meh, K. Scheufeleb, A. Shukaev, B. Uekermann, « preCICE – A fully parallel library for multi-physics surface coupling », ELSEVIER, 2016.
4. Калмыков В.В., Ибраев Р.А. Программный комплекс совместного моделирования океан-лед-атмосфера-почва на массивно-параллельных компьютерах. // Вычислительные методы и программирование. 2013. **14** 88-95
5. С. С. Симаков, А. С. Холодов / Численное исследование содержания кислорода в крови человека при низкочастотных воздействиях/ Матем. моделирование, 20:4 (2008), 87–102.
6. T. Dobroserdova, M. Olshansky "A finite element solver and energy stable coupling for 3D and 1D fluid models", Comput. Methods Appl. Mech. Engrg. 259 (2013), 166-176.

Design and concept development of a new Multiphysics simulation tool for hemodynamics modeling

G.V. Kopytov

Baltic State Federal University, Kaliningrad, Russia, Institute of Numerical Mathematics of Russian Academy of Sciences, Moscow, Russia

A new multiphysics simulation tool for hemodynamics modeling is presented. It allows coupled simulation of various 1D and 3D models of human blood flow. The tool performs mesh coupling of separately running processes via external coupler based on TCP/IP communications.

Keywords: multiphysics and multiscale simulation, hemodynamics, external mesh coupling.

References

1. O. Babur, T. Verhoeff, M. van den Brandt “Multiphysics and Multiscale Software Frameworks: An Annotated Bibliography”, ISSN 0926-4515, 2015.
2. I. M. Kuz’min, L. E. Tonkov, S. P. Kopysov “Algorithms and Software for Solving Coupled Fluid-Structure Interaction Problems on Hybrid HPC Platform», Computer Research and Modeling, 2013, V.5, № 2, pp. 153-164.
3. H. J. Bungartz, F. Lindnerb, B. Gatzhammer, M. Meh, K. Scheufeleb, A. Shukaev, B. Uekermann, « preCICE – A fully parallel library for multi-physics surface coupling », ELSEVIER, 2016.
4. Kalmykov V.V., Ibrayev R.A. A framework for the ocean-ice-atmosphere-land coupled modeling on massively-parallel architectures. // Numerical Methods and programming. 2013. **14** 88-95
5. S. S. Simakov, A. S. Kholodov / Computational study of oxygen concentration in human blood under the low-frequency disturbances / Mathematical Models and Computer Simulations, 20:4 (2008), pp. 87–102.
6. T. Dobroserdova, M. Olshansky "A finite element solver and energy stable coupling for 3D and 1D fluid models", Comput. Methods Appl. Mech. Engrg. 259 (2013), 166-176.