Модел. и анализ информ. систем. Т. **19**, № **5** (2012) 131–141

УДК 004.042+004.45

Управление параллелизмом в задаче автоматической замены агента dataflow-сети на платформе Smart-M3 и в интернете вещей

Баландин С.И., Васильев А.М., Кожемякин Н.И., Лаурэ Д.А., Парамонов И.В. [1]

Технологический университет г. Тампере
Ярославский государственный университет им. П.Г. Демидова

e-mail: sergey.balandin@tut.fi, vamonster@gmail.com, enginegl.ec@gmail.com, den.a.laure@gmail.com, ilya.paramonov@fruct.org

получена 20 октября 2012

Ключевые слова: агент, Smart-M3, интернет вещей, управление параллелизмом

Рассматриваются механизмы и область применимости разрабатываемого авторами решения для автоматической замены обрабатывающих агентов в dataflow-сетях, построенных на базе платформы Smart-M3 и применяемых для сценариев, характерных для интернета вещей. Приведены описание предлагаемого решения и обзор проблем управления параллелизмом, специфичных для реализации данного механизма как на платформе Smart-M3, так и в более широком контексте интернета вещей.

1. Введение

Термин «интернет вещей» (Internet of Things) описывает сетевую инфраструктуру нового поколения, включающую помимо традиционной компьютерной техники широкую номенклатуру уникально идентифицируемых электронных устройств («вещей»), способных к активному взаимодействию друг с другом [1], [2]. «Вещами» в интернете вещей могут быть различные сенсоры (например, датчики температуры), компьютеры, смартфоны, бытовая техника и многие другие объекты. Все эти объекты соединены друг с другом и идентифицируются посредством некоторых уникальных адресов и/или меток (например, RFID). Архитектура интернета вещей подразумевает, что объекты могут выполнять не только функции прямого назначения, но при необходимости дублировать функции других «вещей». Например, компьютер может осуществлять телефонные звонки или выполнять функцию

[1]Работа выполнена при поддержке Федеральной целевой программы «Научные и научно-педагогические кадры инновационной России на 2009–2013 годы». Контракт № 14.B37.21.0876.

телевизора; холодильник может не только сохранять продукты, но и автоматически осуществлять покупки необходимых продуктов через интернет.

Согласно прогнозу WWRF [3] уже к 2017 году ожидается, что в среднем на одного жителя земли будет приходиться до 1000 различных электронных устройств, вовлеченных в предоставление персонализированных сервисов. Существенное увеличение масштабов сетевой инфраструктуры и степени её связности ставит актуальные задачи по выработке общего видения и стандартизации инфраструктурных решений в области интернета вещей. Крупнейшими проектами, в рамках которых ведутся данные исследования, являются EU FP7 IoT-i [4], TiViT IoT [5] и другие.

В данной работе рассматривается параллелизм в контексте сценария использования, обеспечивающего возможности автоматической замены вышедшего из строя обрабатывающего модуля, а также передачи контекста между обрабатывающими модулями при перемещении пользователя.

2. Основные составляющие референс-модели интернета вещей

Приведём описание архитектурной референс-модели интернета вещей в соответствии с документом, разработанным в рамках проекта IoT-A [6]. Данная модель включает в себя следующие составляющие:

- модель предметной области;
- информационную модель;
- функциональную модель.

Модель предметной области (Domain Model) определяет основные понятия, такие как «физическая сущность», «виртуальная сущность», «устройство», «пользователь» и т. д. Информационная модель (Information model) определяет структуру данных в контексте интернета вещей и связи между этими данными. Функциональная модель (Functional Model) является основой для описания способов взаимодействия элементов.

Центральными понятиями модели предметной области являются понятия физической сущности (Physical Entity) и виртуальной сущности (Virtual Entity). Пользователь (User) должен иметь возможность взаимодействовать с физической сущностью реального мира. В роли пользователя здесь может выступать как человек, так и сервис, приложение или программный агент. Физическая сущность может быть представлена как дискретная, идентифицируемая часть физической среды, которая представляет интерес для пользователя. В цифровом мире физические сущности представлены с помощью виртуальных сущностей (Virtual Entity). В контексте интернета вещей такие сущности должны обладать двумя фундаментальными свойствами.

- Виртуальная сущность должна быть цифровым объектом (Digital Artefact). Виртуальная сущность ассоциируется с единственной физической сущностью,

которую она представляет, а также имеет однозначно определяющий её идентификатор. В то же время возможно, что физический объект будет связан с несколькими виртуальными сущностями.

- Виртуальная сущность должна быть синхронизирована с соответствующей ей физической сущностью. Это означает то, что при изменении реального объекта его виртуальная сущность также должна меняться соответствующим образом и наоборот.

Ресурсы (Resources) — это программные компоненты, которые обеспечивают некоторую функциональность: предоставляют информацию о физических сущностях или позволяют ими управлять. Компоненты данного класса разделяются на два подкласса:

- Встроенные ресурсы (On-device Resources) — программные компоненты размещены непосредственно на устройствах. В частности, они ответственны за передачу информации с сенсоров, а также за управление исполнительными механизмами. Этот вид ресурсов может рассматриваться в качестве «моста» между цифровым и реальным миром.

- Сетевые ресурсы (Network Resources)— программные компоненты, исполняемые на выделенных серверах или в облаке.

Сервисы (Services) представляют собой программные компоненты, удовлетворяющие потребности пользователей средствами других компонентов интернета вещей. Все сервисы делятся на три категории.

- Сервисы ресурсов (Resource-level Service) предоставляют доступ к ресурсам, описанным выше. Каждый из таких сервисов обеспечивает доступ к одному ресурсу и отвечает также за надежность, безопасность, устойчивость и производительность.

- Сервисы виртуальных сущностей (Virtual Entity-level Services) обеспечивают доступ к информации о виртуальных сущностях.

- Интегрированные сервисы (Integrated Services) являются композицией сервисов ресурсов и сервисов виртуальных сущностей.

Связь между различными представлениями объекта достигается с помощью устройств (Devices), которые позволяют физическим сущностям быть частью цифрового мира. Таким образом, устройства выступают в роли посредников между физическими сущностями. Устройства делятся на три категории: сенсоры (Sensors), метки (Tags) и исполнительные устройства (Actuators). Главной функцией сенсоров является предоставление информации о контролируемых ими физических сущностях. Метки используются для идентификации ассоциируемых с ними объектов. Исполнительные устройства предоставляют возможность реально воздействовать на физические сущности. Также возможно существование устройств, одновременно относящихся к нескольким категориям.

3. Сценарии использования

В качестве базовой вычислительной модели мы используем модель dataflow-сети [7]. Данная модель представляется весьма удачной для описания класса сервисов в интернете вещей, осуществляющих поэтапную обработку данных, получаемых с сенсоров, для удовлетворения потребностей пользователя.

Входами dataflow-сети являются сенсоры, осуществляющие измерение некоторых параметров физических сущностей и предоставляющих результаты этих измерений для дальнейшей обработки объектами виртуального мира (цифровыми артефактами в терминах интернета вещей).

В узлах сети находятся агенты — обрабатывающие устройства сети. На входе каждый агент может принимать сигналы с одного или нескольких сенсоров и других агентов. На выходе агентом генерируется поток новых значений, полученный на основе входных потоков и заложенного алгоритма обработки. Например, в простейшем случае по температуре и влажности воздуха агент может определять вероятность дождя. Причём для вычисления результирующего значения агент может формировать и учитывать своё текущее состояние, в качестве которого может выступать информация о предыдущих изменениях. Таким образом, каждый агент осуществляет преобразование своего входа в выход и в то же время имеет внутреннее состояние, определяющее контекст вычислений.

Каждый агент, выполняя преобразование своих входных данных, генерирует новые данные, которые могут представлять интерес для других агентов. В свою очередь, из цепочек информационно-связанных агентов могут быть сформированы сервисы интернета вещей, представляющие интерес для конечного пользователя.

Для обеспечения стабильного функционирования рассмотренной структуры необходимо организовать передачу состояния (контекста) между различными обрабатывающими агентами. Можно выделить два основных сценария, связанных с передачей контекста в рамках интернета вещей:

1. Прекращение функционирования обрабатывающего устройства. Данная ситуация может быть связана с выходом конкретного агента из строя (например, в силу ограничений по энергии) или с нарушениями каналов связи, по которым осуществляется взаимодействие агента с другими агентами и сенсорами.

   В этом случае необходимо произвести замену обрабатывающего устройства с передачей состояния последнего таким образом, чтобы с точки зрения пользователя сервис не прерывался.

2. Физическое перемещение объекта, обслуживаемого сервисом. Данная ситуация связана с переходом пользователя из зоны действия одного обслуживающего устройства в зону действия другого. Примером аналогичного процесса является мягкая передача сессии абонента от одной базовой станции к другой (handover) в GSM-сетях.

   В этом случае необходимо использовать другие агенты для обслуживания данного объекта, организовав передачу состояния от старых агентов к новым. Заметим, что в более общем случае в результате перемещения пользователя изменение обслуживающего агента может сопровождаться полным или частичным

изменением входных и выходных потоков, например, в силу перехода обслуживаемого объекта в зону ответственности других сенсоров, предоставляющих доступ к информации, более релевантных с точки зрения местонахождения пользователя.

Выработка общего решения, обеспечивающего передачу контекста в интернете вещей, является задачей высокой сложности. На первом шаге мы рассмотрим решение аналогичной задачи на основе платформы Smart-M3, сфокусировавшись на первом из приведённых выше сценариев использования. Следующим шагом будет произведена оценка возможности переноса данного решения в контекст интернета вещей.

4. Автоматическая замена агента на платформе Smart-M3

Smart-M3 — это платформа с открытым исходным кодом, позволяющая реализовывать схемы обмена и хранения информации любой степени сложности и предоставляющая инфраструктуру для создания приложений в соответствии с парадигмой интеллектуальных пространств (smart spaces) [8]. Приведём краткое описание данной платформы.

Главным компонентом Smart-M3 является SIB (Semantic Information Broker), который управляет информационным хранилищем пространства и позволяет приложениям-агентам — KP (Knowledge Processor) — обмениваться информацией. Информация хранится в SIB в соответствии с моделью RDF (Resource Description Framework). Её базовым элементом является триплет, включающий в себя три составных элемента: «субъект», «предикат» и «объект». Каждый триплет позволяет описывать некоторое отношение между субъектом и объектом.

На низком уровне KP может управлять RDF-триплетами посредством операций добавления, удаления, обновления и запроса данных из SIB. SIB также может уведомлять KP об изменении некоторых данных, для чего соответствующий KP должен быть подписан на отслеживание таких изменений. Для высокоуровневого описания информации, обрабатываемой KP, могут быть использованы онтологии. Онтология не только определяет структуру данных, но и их семантику. Вследствие этого KP могут использовать не только предопределённые структуры данных, но и адаптироваться к различным источникам информации. Кроме того, семантическая информация может быть использована KP для предоставления интеллектуальных сервисов [9, 10].

Модель dataflow-сети является одной из возможных архитектур, объединяющих сенсоры и обрабатывающие модули (агенты). В терминах Smart-M3 в роли последних выступают KP, а взаимодействие между ними осуществляется средствами SIB. При этом поведение вычислительных KP достаточно сложно и может быть разбито на три стадии:

1. Инициализация. Во время этой фазы KP подключается к RDF-хранилищу и получает необходимую ему информацию. После этого KP подписывается

на триплеты, которые формируют входной поток данных. Если необходимые триплеты недоступны, KP может отключиться от хранилища.

2. Функционирование. Во время этой фазы KP генерирует выходные значения. Активация KP осуществляется по уведомлению подписки. Помимо триплетов, которые приходят «по подписке», KP также может запросить некоторые дополнительные данные. После того, как вычисления закончены, KP посылает результирующие значения в виде триплетов в RDF-хранилище.

3. Завершение работы. KP удаляет все активные подписки. Последние выходные данные также могут быть удалены из RDF-хранилища либо заменены значениями, указывающими на прекращение функционирования.

Для решения проблемы потери одним из KP соединения с RDF-хранилищем предлагается механизм замены агентов, основанный на расширении функционала SIB. Когда связь между некоторым KP и RDF-хранилищем обрывается, для замены данного KP используются резервные KP.

Для реализации механизма замены агентов в dataflow-сети, построенной на платформе Smart-M3, необходимо внести изменения в поведение KP. При этом модификации подвергаются только фазы инициализации и завершения работы, тогда как фаза функционирования остаётся неизменной, что позволяет сохранить все свойства сети. В соответствии с изменениями, инициализация KP начинается с его конфигурирования. Следующим шагом KP отправляет запрос на регистрацию в качестве узла dataflow-сети. В этом запросе KP указывает следующую информацию:

- URI описания функционала KP;
- тип резервной программы, которая должна быть запущена для замены KP;
- программа для замены KP.

Результатом данного запроса может быть как регистрация KP в качестве нового dataflow KP, так и разрешение на продолжение прерванной операции. В первом случае KP продолжает процесс инициализации как обычный узел dataflow-сети. Во втором случае KP возвращается в сеть и получает обратно управление и текущий контекст от резервного KP. В этом случае вместе с ответом на запрос регистрации SIB также возвращает список подписок и данные, которые не были использованы заменяющим KP.

В системе, по возможности, должно быть несколько замещающих KP, которые могут выполнять программы различных типов. Резервный KP начинает свою работу с отправки запроса в SIB, чтобы зарегистрироваться в качестве замещающего KP. Вместе с запросом KP отправляет сведения о типах программ, которые он может выполнять. После регистрации резервный KP переходит в состояние ожидания и не выполняет никаких действий.

Когда один из узлов dataflow-сети выходит из строя, SIB посылает замещающему KP резервную программу, список активных подписок и данные, которые не были обработаны основным узлом.

В общем случае КР в соответствии с резервной программой читает информацию о конфигурации из интеллектуального пространства, затем обрабатывает данные, полученные напрямую из SIB, и переходит к фазе функционирования.

Когда вышедший из строя вычислительный узел возвращается обратно в dataflow-сеть, SIB сообщает об этом замещающему КР с помощью сообщения о прекращении функционирования. При получении данного сообщения резервный КР прекращает выполнение программы и освобождает все используемые ресурсы. При этом он не посылает никаких ответных сообщений.

5. Управление параллелизмом для механизма замещения КР

Рассмотренный в предыдущем разделе механизм замещения КР порождает ряд задач управления параллелизмом, которые необходимо решить для обеспечения корректной реализации предложенного решения. В частности, к ним относятся:

1. Проблема конкуренции агентов за выходные потоки данных. Поскольку агент осуществляет запись информации в триплеты, ассоциированные с его выходными потоками в рамках dataflow-модели, возможна ситуация, когда два агента пытаются записывать свои выходные данные в одни и те же триплеты.

2. Проблема рассогласования входных потоков. Предположим, что сенсор передаёт некоторое значение в выходной поток. Это значение обрабатывается двумя различными агентами, выходы которых передаются на вход третьего агента. В зависимости от порядка поступления данных на входы последнего агента его состояние и значения на выходе могут различаться. При этом в общем случае невозможно гарантировать определённый порядок прихода входных значений в связи с вариативностью скорости обработки информации агентами и возможными задержками при передаче информации между ними.

3. Проблема непрерывности временного потока во время замещения агента. Поскольку процесс замены обрабатывающего агента требует некоторого времени, необходимо предотвратить возможную потерю информации во входных потоках замещаемого агента и агента-заместителя.

Перейдём теперь к обсуждению возможных решений перечисленных выше проблем, как в контексте приведённой выше реализации на платформе Smart-M3, так и в рамках интернета вещей.

Первая из перечисленных проблем на платформе Smart-M3 может быть решена с использованием средств блокировки, предоставляемых SIB. В фазе инициализации КР может заблокировать выходные триплеты в RDF-хранилище, чтобы они не могли быть изменены другими КР. По окончании функционирования КР эта защита снимается.

Вторая проблема может быть решена путём наложения дополнительных ограничений на функционирование агентов и окружения. Например, в некоторых случаях

вариативность выхода и состояния агента в промежутке времени между получением данных по каждому из входных каналов может быть непринципиальна и можно потребовать, чтобы выход агента после получения всей необходимой информации не зависел от порядка подачи информации на входы. В более общем случае задача может быть решена посредством введения механизмов управления транзакциями, которые рассмотрены в следующем разделе.

Третья проблема, как и первая, может быть решена на уровне SIB с помощью специального «арбитра», являющегося расширением SIB. В простейшем случае после «выпадения» обрабатывающего элемента из dataflow-сети, арбитр сохраняет все сгенерированные данные, а после ввода в строй замещающего элемента передаёт ему эту информацию. Следует также отметить, что данная проблема тесно связана с предыдущей, поскольку как выявление выхода из строя агента, так и передача управления агенту-заместителю требуют времени, что также может приводить к рассогласованию выходных потоков.

6. Управление параллелизмом в интернете вещей

Как уже отмечалось, интернет вещей является системой, включающей множество объектов и множественные связи между ними. С ростом количества объектов возрастает и количество пересылаемых данных, а следовательно, и количество одновременных обращений к одним и тем же данным. Это, а также сложность и непостоянство структуры взаимодействующих объектов, делают задачи управления параллелизмом в рамках интернета вещей особо актуальными. Следует также отметить, что решения этих проблем должны учитывать множество различных аспектов, таких как особенности управления ресурсами, возможность выхода из строя компонентов, энергоэффективность.

Проблемы управления параллелизмом, приведённые в предыдущем пункте, сходны с проблемой пересекающихся транзакций в распределённых базах данных, а именно с ситуациями, когда несколько транзакций пытаются одновременно изменить значения одних и тех же данных в БД. Решения данной проблемы связаны с выполнением требований ACID, включающих атомарность, согласованность, изолированность и долговечность. Стратегии реализации принципов ACID разделяются на 2 типа: пессимистичные (использующие блокировку данных) и оптимистичные (не использующие блокировку данных).

В работе [11] описан алгоритм оптимистичного управления параллелизмом для управления транзакциями в базах данных, расположенных на устройствах с ограниченными ресурсами. Основными целями данного алгоритма являются высокая энергоэффективность и низкое время задержек, что делает его эффективным при использовании на мобильных платформах. Для достижения этих целей данный алгоритм обеспечивает уменьшение откатов транзакций. В основе алгоритма лежат следующие идеи:

- Forward Oriented Optimistic Concurrency Control (FOCC) [12] — стратегия валидации транзакций, при которой сравнивается набор данных, которые подлежат записи валидируемой транзакцией, и наборы данных, которые читаются

или были недавно прочитаны всеми выполняемыми в данный момент транзакциями.

- Виртуальное выполнение транзакций — механизм, позволяющий отменённым транзакциям выполняться повторно, используя локально сохранённые данные.

Отличительной особенностью предложенного авторами алгоритма является смена порядка фаз чтения, записи и валидации транзакции. В отличие от традиционного порядка, когда после фазы чтения идет фаза валидации и затем фаза записи, в указанном алгоритме авторов порядок фаз записи и валидации изменён так, что после фазы чтения идет фаза записи и лишь затем фаза валидации. Это позволяет отказаться от блокирования транзакций во время фаз записи и валидации.

Высокая энергоэффективность и низкое время задержек, обеспечиваемые предложенным алгоритмом, являются критичными и для интернета вещей. Однако в том виде, в котором алгоритм описан в статье [11], его затруднительно применять для решения задач, приведённых в предыдущем разделе. Виртуальное выполнение транзакций требует, чтобы каждая сущность модели интернета вещей имела локальное хранилище временных данных, что может вызвать сложности при реализации модели. Кроме того, данный алгоритм предполагает наличие в системе некоторого менеджера транзакций или даже нескольких таких менеджеров, которые должны управлять параллелизмом и разрешать конфликты, однако архитектура интернета вещей также не предусматривает наличия таких компонентов в системе.

В работе [13] дан обзор решений проблемы, когда несколько мобильных устройств пытаются получить доступ к одним и тем же данным, расположенным на некотором сервере, одновременно.

Решение проблемы одновременного доступа к данным в мобильных системах с помощью пессимистичных стратегий в чистом виде невозможно, поскольку разрывы соединений являются обычным явлением для мобильных устройств и вероятны ситуации, когда разрыв происходит во время транзакции. В этом случае блокировка с заблокированных этой транзакцией данных не снимается и они остаются недоступными для остальных процессов. Для решения этой проблемы каждую транзакцию ассоциируют с таймером. Если транзакция не завершается за время, отведенное таймером, она прерывается, считается неудачной, а блокировка с данных снимается. Несмотря на то, что подход, основанный на таймерах, позволяет избежать проблемы «голодания» (когда данные оказываются заблокированы навсегда), он повышает количество транзакций, поскольку неудачные транзакции должны выполняться снова. Это в свою очередь ведет к увеличению количества данных, передаваемых по сети, что ведет к уменьшению пропускной способности канала. У оптимистических стратегий в мобильных системах также появляются свои недостатки. Так, например, Concurrency Control Mechanism должен предполагать увеличение размеров кэша при увеличении количества мобильных устройств в системе. В случае Sequential Order with Dynamic Adjustment недостатком является необходимость каждого мобильного устройства в системе иметь свою локальную базу данных. Недостатком всех оптимистичных стратегий является то, что они увеличивают количество пе-

редаваемых данных, что увеличивает нагрузку на сеть и уменьшает пропускную способность.

Недостатки решений, описанные в работе [13], остаются актуальными и в интернете вещей. Причем в последнем случае эти недостатки являются более критичными, чем для мобильных систем, поскольку, в отличие от мобильных систем, в интернете вещей каждый объект является не только клиентом, но и «сервером», поэтому количество передаваемых между объектами данных гораздо больше, чем в мобильных системах. Уменьшение пропускной способности каналов передачи данных приведет к значительным задержкам в системе и отрицательно скажется на её производительности и качестве работы.

Список литературы

1. Atzori L., Iera A., Morabito G. The internet of things: A survey // Computer Networks. — 2010. — Vol. 54, no. 15. — P. 2787–2805.

2. Internet of Things Strategic Research Roadmap / O. Vermesan, P. Friess, P. Guillemin et al. // Internet of Things — Global Technological and Societal Trends. — 2011. — P. 9–52.

3. User Scenarios 2020 — a Worldwide Wireless Future / L. Sørensen, K. E. Skouby, D. Dietterle et al. // WWRF Outlook: Visions and Research Directions for the Wireless World. — 2009. — July. — no. 4.

4. The Internet of Things Initiative. — URL: http://www.iot-i.eu/public/front-page.

5. TiViT Internet of Things. — URL: http://www.internetofthings.fi.

6. Internet-of-Things Architecture. Deliverable D1.3 — Updated reference model for IoT v1.5. — URL: http://www.iot-a.eu/public/public-documents/documents-1/1/1/copy_of_d1.2/at_download/file.

7. Kok J. A fully abstract semantics for data flow nets // PARLE Parallel Architectures and Languages Europe / Springer. — 1987. — P. 351–368.

8. Smart-M3 information sharing platform / J. Honkola, H. Laine, R. Brown, O. Tyrkkö // 2010 IEEE Symposium on Computers and Communications (ISCC) / IEEE. — 2010. — P. 1041–1046.

9. Luukkala V., Honkola J. Integration of an answer set engine to Smart-M3 // Smart Spaces and Next Generation Wired/Wireless Networking. — 2010. — P. 92–101.

10. Case study: Context-aware supervision of a smart maintenance process / S. Pantsar-Syvaniemi, E. Ovaska, S. Ferrari et al. // Proceedings of the 11th IEEE/IPSJ International Symposium on Applications and the Internet (SAINT 2011) / IEEE. — 2011. — P. 309–314.

11. Solaiman K., Morgan G. Later validation/earlier write: Concurrency control for resource-constrained systems with real-time properties // Proceedings of 2011 30th IEEE Symposium on Reliable Distributed Systems Workshops (SRDSW) / IEEE. — 2011. — October. — P. 9–12.

12. Härder T. Observations on optimistic concurrency control schemes // Information Systems. — 1984. — Vol. 9, no. 2. — P. 111–120.

13. Concurrency control in mobile environments: Issues & chalenges / S. A. Moiz, S. N. Pal, J. Kumar et al. // International Journal of Database Management Systems (IJDMS). — 2011. — November. — Vol. 3, no. 4. — P. 147–159.

Concurrency Control in the Problem of Automatic Substitution of the Dataflow Network Agent on Smart-M3 Platform and in the Internet of Things

Balandin S.I., Vasilev A.M., Kozhemyakin N.I., Laure D.A., Paramonov I.V.

Keywords: Agent, Smart-M3, Internet of things, Parallelism control

The paper describes implementation of dataflow networks based on Smart-M3 platform for use cases related to the Internet of Things. The mechanism for automatic substitution of computational agents created on top of Smart-M3 platform is described. The paper reviews concurrency issues of the developed solution regarding Smart-M3 platform, as well as in the broader context of the Internet of Things.

Сведения об авторах:
Баландин Сергей Игоревич,
Технологический университет г. Тампере (Финляндия),
канд. техн. наук, адъюнкт-профессор,
президент ассоциации открытых инноваций FRUCT.
Васильев Андрей Михайлович,
Ярославский государственный университет им. П.Г. Демидова,
аспирант каф. теоретической информатики.
Кожемякин Никита Ильич,
Ярославский государственный университет им. П.Г. Демидова,
студент факультета ИВТ.
Лаурэ Денис Александрович,
Ярославский государственный университет им. П.Г. Демидова,
студент факультета ИВТ.
Парамонов Илья Вячеславович,
Ярославский государственный университет им. П.Г. Демидова,
канд. физ.-мат. наук, старший преподаватель каф. компьютерных сетей.