2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom)

Росистые вычисления: дополнение облачных вычислений

Andy Rindos
Emerging Technology Institute (Cloud)
International Business Machines Corporation (IBM)
Raleigh-Durham, North Carolina, USA
Email: rindos@us.ibm.com

Yingwei Wang
School of Mathematical and Computational Sciences
University of Prince Edward Island
Charlottetown, Prince Edward Island, Canada
Email: ywang@upei.ca

На горизонте появились росистые вычисления (Dew computing) – как новая парадигма и как новая область для исследований. В данной статье мы изучим сущность росистых вычислений, обсудим их потенциал и трудности применения этой технологии. При использовании росистых вычислений традиционные компьютерные приложения переходят на такой уровень, когда они постоянно взаимодействуют с облачными сервисами; по сути, росистые вычисления – это развитие настольных приложений. Основываясь на возможностях и требованиях этой технологии, мы предлагаем и новый вид компьютеров: росистые компьютеры. Несмотря на то, что мы уже представляем себе некоторые характеристики этих росистых компьютеров, их точные возможности ещё необходимо исследовать и описать. Мы уже упомянули облачные сервисы: росистые вычисления тесно увязаны с облачными, и являются их дополнением.

1. ВВЕДЕНИЕ

Тема этой статьи касается новой области исследований: росистых вычислений [1] [2]. В настоящее время мы постоянно с большим количеством умных слов. Когда звучит термин «росистые вычисления», люди могут подумать, что это ещё один модный термин без особого смысла. Хорошо, так это или нет, мы хотим спросить: нужен ли этот новый термин? Несёт ли он что-то новое?

В этой статье мы обсудим значение росистых вычислений и их историю (Глава 2), потенциал применения росистых вычислений (Глава 3), технические проблемы применения росистых вычислений (Глава 4) и связь между росистыми и облачными вычислениями (Глава 5). Из всех этих рассуждений читателю предлагается самому

сделать вывод о том, является ли росистые вычисления модным словечком, и стоит ли запомнить эту фразу.

2. ЧТО ТАКОЕ РОСИСТЫЕ ВЫЧИСЛЕНИЯ?

Несколько авторов рассматривали вопрос росистых вычислений под разными углами [1] [2] [3] [4] [5] [6] [7] [8] [9]. Некоторые из этих статей описывали Р.В. или давали им определение. Мы сравним эти описания/определения для выявления общих функций и ключевых особенностей.

Ванг (Wang) [1] [2] предложил следующее определение: *«Росистые вычисления – это локальная программно-аппаратная организационная парадигма облачного окружения, когда локальный компьютер предоставляет функциональность, которая не зависит от облачных сервисов, сотрудничая при этом с облачными сервисами. Цель росистых вычислений - в полной мере реализовать потенциал локальных компьютеров и облачных сервисов».*

Скала (Skala) [8] описывает росистые вычисления следующей фразой: «*Росистые вычисления (Р.В.) выходят за рамки концепции сети/хранилища/сервиса – они основываются на концепции микросервиса в вертикально-распределённой иерархии вычислений». «Р.В. толкают границы приложений и низкоуровневых сервисов от централизованных виртуальных узлов к конечным пользователям».*

Ристов [9] обсуждает функции РВ в следующей фразе: «*Идея, стоящая за росистыми вычислениями — использовать ресурсы раньше, чем они будут обработаны на облачном сервере. В архитектуре росистых вычислений используются микросервисы, взаимодействующие с макросервисами, т.е. росистые сервисы, взаимодействующие с облачными сервисами*».

Мы отметили в определении Ванга две ключевые функции: независимость и сотрудничество; определение Скала подчёркивает положение росистых вычислений в связи другими компонентами, и это примерно эквивалентно независимости; описание Ристова содержит и функции независимости, и функции совместной работы (*сотрудничества*), что практически идентично [2].

Суть в том, что все эти три определения/описания не противоречат друг другу. Поэтому, далее мы будем использовать определение Ванга.

Это определение относится к локальным компьютерам. Концепция локальных компьютеров часто используется в облачных вычислениях; локальные компьютеры – это необлачные компьютеры. Локальные компьютеры включают в себя персональные

компьютеры (десктопы или ноутбуки), планшеты, мобильные телефоны, сервера и кластеры.

Это определение подчёркивает две ключевые функции: независимость и сотрудничество. *Независимость* означает то, что локальные компьютеры предоставляют функциональность без облачных сервисов и соединения с сетью Интернет. Другими словами, это означает, что приложение не полностью онлайн или основанное на облачных сервисах. *Сотрудничество* означает то, что росистые приложения во время работы автоматически обмениваются информацией с облачными сервисами.

Такое сотрудничество включает в себя синхронизацию, корреляцию или другие виды взаимодействия.

Для полного понимания концепции РВ, мы приведём несколько небольших примеров. Некоторые примеры являются росистыми приложениями, а некоторые нет.

Первый пример — это **Dropbox** [10]. Файлы и папки в **Dropbox** всегда доступны пользователям, несмотря на то, что сервера **Dropbox могут быть** недоступны. Это реализация функции *независимость.* Файлы и папки автоматически синхронизируются с серверами Dropbox; это реализация функции *сотрудничество.* Таким образом, **Dropbox** – росистое приложение.

Второй пример — это **Google Drive** [11]. Несмотря на то, что **Google Drive** очень похож на **Dropbox**, это не приложение с использованием росистых вычислений. Причина, по которой **Google Drive** не удовлетворяет критерию независимости: файл не может быть открыт, если сервера **Google Drive** недоступны.

Третий пример — это **TurboTax** [12], Канадский сервис подоходного налога. Это программное обеспечение имеет десктопную версию, которая устанавливается локально, а также доступна онлайн-версия. По-видимому, настольная версия удовлетворяет критерию независимости, но не удовлетворяет критерию *сотрудничества*, потому что не обменивается информацией с онлайн-сервисом. Это программное обеспечение — не приложение с использованием росистых вычислений.

Из этих примеров мы можем сделать следующие наблюдения:

Первое, росистые приложения (РП) — это специальная группа приложений, удовлетворяющих критериям *независимости* и *сотрудничества*. Не все онлайн приложения и доступные по двум каналам (локально и онлайн) являются росистыми приложениями.

Второе, хотя росистые вычисления — это новый термин и новая концепция, приложения, удовлетворяющие требованиям росистых вычислений, существовали на

протяжении многих лет. Подробный обзор существующих росистых приложений доступен в [1].

Слово *роса* было связано с вычислениями в статье [3], общедоступной в онлайн режиме с января 2015 года; то, что было предложено в этой статье, основывалось на Web, и было эквивалентно росистым вычислениям (категория WiD, Web in Dew была раскрыта в [1]). Позже широкое определение росистых вычислений было предложено в [2]. Помимо трёх определений и описаний [1] [8] [9], которые мы представили выше в этом разделе, другие крупные работы в области росистых вычислений включают некоторые статьи по облачно-росистой архитектуре [4] [5] [7], масштабируемой архитектуре распределённых иерархических вычислений, включающей облачные вычисления, туманные вычисления и росистые вычисления [8] и отношения между облачными, туманными и росистыми вычислениями [6].

Мы обсудили концепцию росистых вычислений. Следующий вопрос, который мы ходим задать: что росистые вычисления принесут нам? В Главе 3 мы увидим, что могут росистые вычисления принести пользователям. В Главе 4, мы увидим, что принесут росистые вычисления технологиям разработки.

3. ПОТЕНЦИАЛЬНЫЕ ВОЗМОЖНОСТИ РОСИСТЫХ ВЫЧИСЛЕНИЙ

Ценность концепции росистых вычислений может быть отражена по-разному. В этой главе мы покажем, что росистые вычисления, как новая парадигма, могут вдохновлять на создание новых приложений.

Можем ли мы предложить новые приложения, удовлетворяющие определению росистых вычислений? Во-первых, рассмотрим пример росистых вычислений, показанный в Главе 2: **Dropbox.** Мы заметили несколько росистых приложений, очень похожих на **Dropbox**, такие как **OneDrive** [13], **Google Drive Offline** [14] и так далее. Все эти приложения предоставляют сервисы хранения файлов; все они удовлетворяют критерию *независимости* и *сотрудничества*. Мы можем использовать категорию для описания всех этих приложений и назвать её «Хранение в Росе» (Storage in Dew, STiD) [1].

Чтобы обобщить эту идею мы можем создавать другие категории в формате X-в-Росе, где X – вид ресурса или услуги; основная суть X-в-Росе заключается в том, что X как-то присутствует на локальном компьютере и X обменивается информацией с облачным сервисом. Цель этой статьи не исследовать все возможные категории.

Подробное обсуждение различных категорий можно найти в [1]. Тут мы обсудим только несколько категорий, чтобы показать вдохновляющую силу росистых вычислений.

Мы начинаем с категории, называемой Web in Dew (WiD). Буквальное значение WiD – Всемирная паутина, помещённая в локальный компьютер. Поскольку эта задача невозможна, мы всё же можем захотеть, чтобы хотя бы часть Сети была помещена на локальный компьютер. Чтобы реализовать WiD, нам необходимо установить на компьютер веб-сервер, создать веб-сайты на локальном компьютере, расширить маппинг сетевых имён на локальный компьютер и решить некоторые другие технические проблемы. Мы также должны помнить, что WiD должен не только размещать дублированные веб-сайты на локальном компьютере, но и организовать взаимодействие с облачным веб-сайтом. Все детали WiD отражены в документе, описывающей облачно – росистую архитектуру [3] [5] [7]. Рисунок 1 показывает центральную идею облачной – росистой архитектуры, которая реализует WiD.

WiD делает возможным веб-сёрфинг без соединения с Интернетом. Когда отсутствует соединение, пользователь по-прежнему имеет доступ к вебсайтам, развёрнутым на локальном компьютере. Мы проиллюстрируем эту идею на примере.

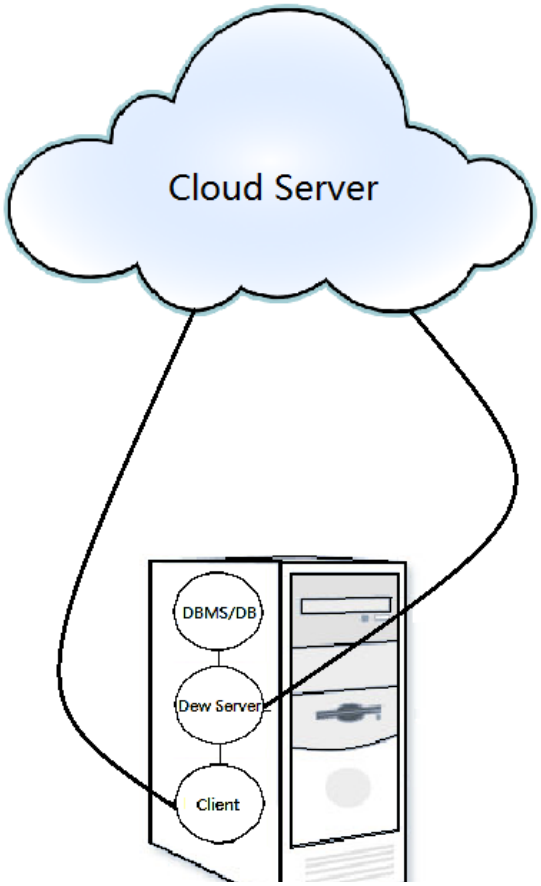

Рисунок 1 — Архитектура облако-роса. новый тип сервера, росистый сервер. Росистый сервер - это веб-сервер, который находится на локальном компьютере пользователя. Росистый сервер и связанные с ним базы данных имеют две функции: во-первых, он предоставляет клиенту те же сервисы, что и облачный сервер; во-вторых, он синхронизирует базы данных росистого сервера с СУБД в облаке

Предположим, что пользователь посещает вечеринку, где нет подключения к Интернету. Пользователь не может открыть http://www.facebook.com в своём браузере, чтобы показать фотографии или найти какую-либо информацию. Если http://www.facebook.com принял идею росистых вычислений и создал WiD-приложение, то веб-сайт был бы продублирован на локальном компьютере. Дублирование — это не совсем копирование: дублированный веб-сайт не должен иметь дело с глобальной большой нагрузкой, и он должен быть гораздо проще чем веб-сайт; дублированный сайт будет реализован на общедоступных технологиях, чтобы не допустить утечки корпоративных

секретов; дублированная база данных веб-сайта не будет слишком большой, поскольку она содержит только данные, относящиеся к пользователю. Мы можем использовать http://mmm.facebook.com для ссылки на дублированный веб-сайт на локальном компьютере, где mmm является индикатором того, что это локальный веб-сайт. Когда доступно подключение к Интернету, пользователь может использовать оба веб-сайта http://www.facebook.com и дублированный http://mmm.facebook.com; когда Интернет-соединение не доступно, то веб-сайт http://www.facebook.com недоступен, но пользователь по-прежнему может использовать http://mmm.facebook.com для доступа к своим фотографиям и другой информации или выполнять необходимые обновления.

Эти обновления будут синхронизированы с веб-сайтом, когда снова будет доступно подключение к Интернету. На локальном компьютере может размещаться множество дублированных веб-сайтов, чтобы веб-сёрфинг был частично доступен, когда нет подключения к Интернету.

WiD может использоваться для различных целей. Интернет-соединение не всегда доступно. Даже когда подключение к Интернету станет доступно в любом месте, все равно будут возникать издержки и риски, связанные с подключением к Интернету. Тут мы пропустим приложения, связанные со снижением стоимости услуг, с использованием в военных и политических целях и т.д., а сосредоточимся на одной области применения WiD: Интернете вещей (IoT).

Интернет вещей (IoT) [15] [16] – быстрорастущая область исследований и применения. С точки зрения «вещей» есть различные технологии, такие как RFID, NFC и беспроводные сенсорные сети (то есть IEEE 802.15.4, ZigBee, 6LoWPAN, Wireless M-Bus), которые могут использоваться для объединения их в сети и, в конечном итоге, подключению к Интернету. Некоторые из этих технологий уже используются в приложениях. Но мы больше беспокоимся об информации, сгенерированной такими «вещами» и обработке такой информации.

Web of Things (WoT) [17] предоставляет уровень приложений IoT, который упростит создание приложений обработки информации IoT. Хотя «вещи» будут генерировать большие объёмы потоковых данных и могут, в свою очередь, коммуницировать между собой автоматически, важным требованием является то, что большая часть данных должна быть сохранена и обработана локально. Таким образом, необходимы локальные пользовательские интерфейсы и центры управления.

Например, хотя сотни и тысячи датчиков и устройств внутри дома физически подключены к IoT, большая часть данных должна быть сохранена и обработана внутри дома. Только некоторые необходимые и разрешённые данные должны быть отправлены из дома во внешний мир. Пользовательский интерфейс или центр управления необходимы для того, чтобы пользователь мог установить или изменить настройки и правила. WiD идеально подходит для таких приложений: локальным компьютером с приложением WiD будет центр управления; пользователи могут контролировать и управлять всеми устройствами и датчиками внутри дома с веб-сайтов, находящихся на локальном компьютере; локальный компьютер, при необходимости, будет обмениваться информацией с облачными серверами, когда такие коммуникации будут разрешены пользователем. На рисунке 2 показан типичный сценарий, в котором росистая архитектура используется для организации системы IoT.

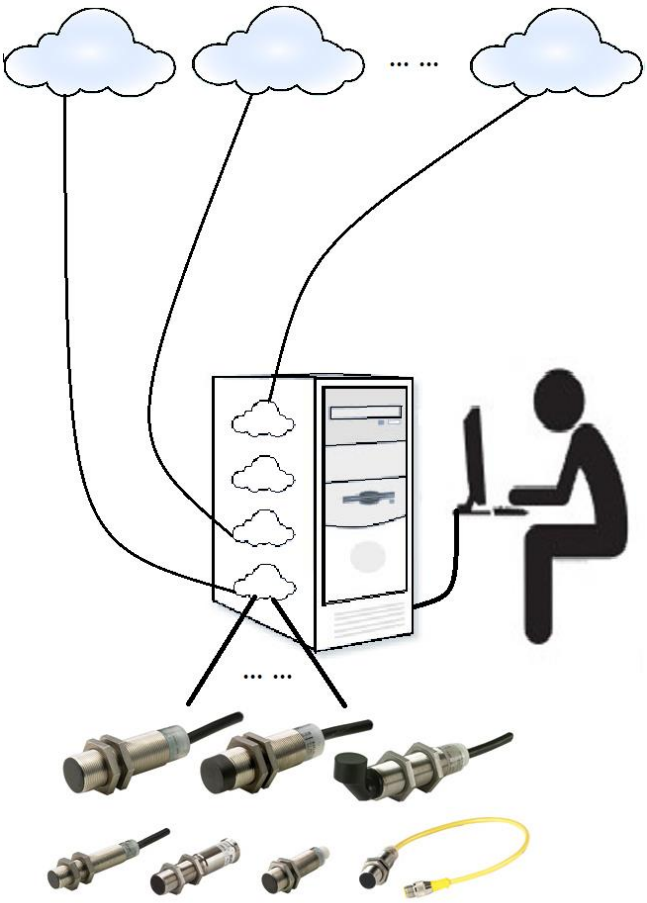

Рисунок 2 — Локальный веб-сайт, созданный в соответствии с типом WiD, используется для управления устройствами и датчиками. Большая часть данных будет храниться локально. Весь обмен данных с облаком контролируется пользователем через локальный веб-сайт так же, как и через облачный центр управления. Данные отправляются в облако только тогда, когда это необходимо и разрешено пользователем

Такой механизм может быть использован в доме, в автомобиле, на ферме, на военном корабле и т.д. В общем и целом, WiD с его структурой и росистая архитектура представляют иерархическую структуру серверов, которая напоминает иерархическую структуру мира. С развитием IoT и WoT, WiD и росистая облачная архитектура будут играть важную роль в тех местах, где необходима иерархическая структура серверов. Следует отметить, что роль росистых вычислений в IoT отличаются от ролей туманных вычислений там же. Парадигма туманных вычислений расширяет облачные вычисления на уровне сети. Подобно облаку, туман предоставляет конечным пользователям данные,

вычисления, хранилища и приложения. Отличительными характеристиками туманных вычислений является их близость к конечным пользователям, географическая плотность и поддержка мобильности [16].

Туманные вычисления могут способствовать IoT к более эффективному использованию устройств на границе сети. Росистые вычисления — к более эффективному использованию локальных компьютеров.

После обсуждения WiD мы хотели бы обсудить ещё одну категорию росистых вычислений: росистую инфраструктуру (Infrastructure as Dew, IaD).

Росистая инфраструктура (IaD) [1] — это категория, которая противоположна инфраструктуре как сервису (IaaS) в облачных вычислениях. В IaaS (виртуальная) машина находится в облаке; в IaD (реальная) машина – это локальный компьютер.

Если концепция IaD является просто псевдонимом локального компьютера, нам не нужно вводить такой новый термин. Новая функция, которую привносит IaD, в том, что локальный компьютер динамически поддерживает облачные сервисы. Другими словами, данные локального компьютера должны динамически контролироваться и регистрироваться облачными службами; в случае утери данных на локальном компьютере, мы должны иметь возможность восстановить потерянные данные из облака.

IaD может быть реализован в различных формах. Мы просто обсудим одну из них: полное разделение данных и устройства. Локальный компьютер может хранить все свои настройки/данные в облачном сервисе. Такие настройки/данные не только включают в себя системные настройки/данные, но также включают в себя все настройки / данные приложений. Если IaD полностью реализован, то данные и устройство могут быть полностью разделены. Если ноутбук или сотовый телефон утерян или повреждён, пользователю нужно только получить новое устройство, и все настройки/данные будут полностью восстановлены на нем. В настоящее время некоторые компании-производители сотовых телефонов предоставляют функциональность резервного копирования/восстановления, однако не все настройки/данные могут быть восстановлены. (*спорное утверждение! Прим. переводчика).* С развитием IaD такое полное восстановление будет возможно.

Выше мы обсудили две категории росистых вычислений: WiD и IaD, а также их возможные применения.

Мы можем сформулировать два следующих наблюдения:

Во-первых, приложения росистых вычислений, такие как автономный веб-сёрфинг и полное восстановление сотовых телефонов, очень привлекательны с точки зрения пользователя. У них особый аромат: аромат реального распределения.

Во-вторых, росистые вычисления и их категории очень полезны при разработке новых приложений. Парадигма росистых вычислений систематизирует мышление в поиске новых приложений с возможностями *независимости* и *сотрудничества*. С разработкой росистых вычислений появится ещё много новых категорий и приложений, созданных в рамках этой парадигмы. Потенциал росистых вычислений ограничен только воображением.

4. ТЕХНИЧЕСКИЕ ТРУДНОСТИ РОСИСТЫХ ВЫЧИСЛЕНИЙ

Росистые вычисления, как новая парадигма, приводят к многим техническим проблемам. Эти сложности охватывают широкий спектр областей информатики, включая аппаратное обеспечение, операционные системы, сети, базы данных, браузеры/серверы и т.д. Поговорим немного об этом предметно.

4.1. Аппаратное обеспечение

Росистые вычисления накладывают проблемы в аппаратных технологиях и дизайне компьютерных систем. Поговорим об одной из них.

Одним из требований к росистым вычислениям является то, что локальным компьютерам необходимо постоянно обмениваться информацией с облачными сервисами, чтобы реализовать функцию сотрудничества. Такой обмен всегда должен происходить, даже если пользователи не работают с компьютером. Поддержка работы локального компьютера 24/7 расходует слишком много энергии. Для всех видов локальных компьютеров, если они используют росистые вычисления, необходим режим ожидания, аналогичный режиму сотового телефона. Быстрое развитие твердотельных накопителей (SSD) и связанных с ними технологий, может быть решением данной проблемы.

Другая аппаратная проблема связана с многоядерными процессорами. В режиме ожидания может потребоваться понизить процессора до работы в одноядерном режиме.

4.2. Операционные системы

Как мы обсуждали ранее, некоторые приложения росистых вычислений были разработаны до того, как была предложена сама концепция росистых вычислений, но с развитием росистых вычислений появится гораздо больше росистых приложений. Все эти приложения функционируют поверх операционных систем. На каком-то этапе было бы неплохо интегрировать механизмы росистых вычислений непосредственно в операционную систему. Эффективность локальных компьютеров и связанных с ними

облачных сервисов может быть повышена, если функция сотрудничества росистых вычислений будет управляться ядром операционной системы. Управление росистыми вычислениями может быть основной функций ядра или расширенной функциональностью операционной системы. Росистый сервер и другие компоненты росистых вычислений могут быть частью операционной системы, аналогично тому, как сервер IIS (Internet Information Services) является частью системы Windows.

4.3. Сети

В настоящее время каждое росистое приложение использует собственные механизмы *сотрудничества;* реализация каждого приложения содержит все функции от пользовательского интерфейса до коммуникаций. Если несколько росистых приложений работает на компьютере, то этот подход будет работать. В будущем, когда многие росистые приложения будут работать на одном локальном компьютере, есть вероятность, что они будут конфликтовать друг с другом в отношении коммуникационных портов и других ресурсов. Даже если не будет конфликта между приложениями – этот путь не является наиболее эффективным, так как каждое приложение в некоторой части дублирует другое. Избыточно. Необходимо разработать новый протокол связи для передачи информации между росистым приложением и облачными сервисам. Такой протокол – часть фундаментальной инфраструктуры росистых вычислений.

4.4. Базы данных

Как указывает [1], все типы росистых вычислений имеют одну общую особенность: некоторые объёмы данных присутствуют на локальном компьютере и автоматически синхронизируются с облачными сервисами. Данные и базы данных являются ключевыми элементами росистых вычислений. Хотя технологии баз данных являются вполне зрелыми, некоторые сложности возникают, когда базы данных используются в росистых приложениях.

В росистой вычислительной среде многие серверные технологии мигрируют с серверов на локальные компьютеры. Серверы находятся в строго контролируемой среде, но локальные компьютеры находятся в гораздо более сложном окружении. То, что не было проблемой безопасности на сервере, вполне может стать таковой в росистых вычислениях. Например, на сервере вполне безопасно, что бы файл кода содержал учётные данные пользователя базы данных (*Хотя это и плохая практика! Прим.переводчика),* потому что доступ к серверу находится под высоким контролем; для росистого приложения такой файл кода, который содержит учётные данные, не будет

являться безопасным по многим причинам: вредоносное ПО на локальном компьютере, ошибки пользователей, умышленный взлом и т.д.

Проблемы с безопасностью баз данных должны решаться с разных точек зрения, если эти базы данных планируются использовать в росистых приложениях.

4.5. Браузеры и серверы

Браузеры создавались для получения и отображения гипертекста из сети Интернет. Веб-серверы создавались для доставки гипертекста в Интернет. В росистых вычислениях типа WiD, браузер используется для получения и отображения гипертекста с локальных компьютеров; веб-серверы используются для доставки гипертекста на тот же локальный компьютер.

Веб-серверы хорошо работают в среде WiD, однако браузеры могут выполнять задачи WiD только с помощью специальных технологий [3] [7]. Это связано с тем, что мы хотим, чтобы пользователи работали с URL в WiD аналогично тому, как они работают в сети Интернет, в качестве решения мы предлагаем механизм локальных доменных имён (Local Domain Names System LDNS) [3].

Текущее решение по реализации LDNS является временным и далеко не идеальным. Более простым является решение по редизайну браузеров с учётом требований WiD. Хотя мы рассчитываем, что браузеры будут переработаны с учётом требований LDNS, другая сторона проблемы – переработать облегчённые браузеры и серверы для целей WiD. В среде WiD сервер и браузер находятся на одном компьютере, и можно переработать сервер и браузер так, чтобы они были намного проще и использовали гораздо меньше ресурсов системы.

4.6. Росистые компьютеры

Из разделов 4.1 – 4.5 мы видим, что росистые вычисления создают технические проблемы для локальных компьютеров по всем аспектам организации системы. Традиционную организацию компьютерной системы необходимо изменить для устранения проблем росистых вычислений. Этот новый тип компьютеров можно называть готовыми-к-росе или просто росистыми компьютерами. Все возможности и функции росистых компьютеров пока не ясны, но следующие характеристики мы можем спрогнозировать:

- Он (*компьютер)* имеет режим ожидания для обмена информацией с облачными сервисами 24/7, используя при этом технологии энергосбережения.
- Его операционная система поддерживает росистые вычисления.

- Он имеет унифицированную сетевой фреймворк / протокол для выполнения операций *сотрудничества*.
- Его базы данных управляются для решения проблем безопасности;
- Его браузеры поддерживают приложения WiD.

5. РОЛЬ РОСИСТЫХ ВЫЧИСЛЕНИЙ

Из главы 3 и 4 мы видим, что росистые вычисления могут порождать новые приложения и способствовать дальнейшему развитию технологий; росистые вычисления привносят новый контент в информатику и новые приложения для пользователей. У росистых вычислений есть определённая ценность. Мы можем сказать, что росистые вычисления – новая перспективная исследовательская область, а не бесполезное модное слово.

Изучив ценность и потенциал новой парадигмы, мы можем захотеть сделать шаг назад и охватить картинку целиком, особенно связь между облачными и росистыми вычислениями.

Рассматривая взаимосвязь между росистыми вычислениями и приложениями, росистые вычисления способствуют тому, что, если это возможно, то все локальные компьютерные программы получат поддержку облачных сервисов. Без росистых вычислений локальные компьютерные приложения останутся изолированными. Таким образом, росистые вычисления являются будущим локальных компьютерных приложений.

Рассматривая связь между облачными и росистыми вычислениями, мы можем обобщить следующие моменты:

Первое. Росистые вычисления тесно связаны с облачными вычислениями. Определение росистых вычислений показывает, что росистые вычисления основаны на облачных вычислениях. Без облачных вычислений не было бы росистых вычислений.

Второе. Росистые вычисления не являются частью облачных вычислений. Область росистых вычислений выходит за рамки облачных вычислений. Росистые вычисления охватывают локальные компьютеры, которые не покрыты облачными вычислениями. Хотя облачные вычисления станут всё более популярными, пользователи всегда (Спорное утверждение. Прим. переводчика) будут использовать локальные компьютеры.

Третье. Росистые вычисления помогают раскрыть потенциал облачных вычислений. Р.В. способствуют тому, что бы все приложения как-то использовали

облачные сервисы. С росистыми вычислениями облачные вычисления могут достигнуть наивысшей популярности.

6. ВЫВОДЫ

В этой статье мы обсудили различные аспекты росистых вычислений. Росистые вычисления — это программно-аппаратная организационная парадигма компьютера в среде облачных вычислений. Они подчёркивают две ключевые особенности: независимость и сотрудничество. Различные типы росистых вычислений очень интересны с точки зрения создания новых привлекательных приложений. Эта новая парадигма приводит ко многих техническим проблемам. Эти технические сложности охватывают широкий спектр областей информатики, включая аппаратное обеспечение, операционные системы, сети, базы данных, браузеры/серверы и т.д. При использовании росистых вычислений традиционные компьютерные приложения переходят на такой уровень, когда они постоянно взаимодействуют с облачными сервисами; по сути, росистые вычисления – это развитие настольных приложений.

Основываясь на возможностях и требованиях этой технологии, мы предложили новый вид компьютеров: росистые компьютеры. Несмотря на то, что мы уже представляем себе некоторые характеристики этих росистых компьютеров, их точные возможности ещё необходимо исследовать и описать. Росистые вычисления тесно связаны с облачными вычислениями; росистые вычисления не являются частью облачных вычислений; росистые вычисления помогают реализовать потенциал облачных вычислений.

Суммируя вышесказанное – росистые вычисления являются дополнением облачных вычислений.

7. ССЫЛКИ

[1] Y. Wang, “Definition and categorization of dew computing,” Open Journal of Cloud Computing (OJCC), vol. 3, no. 1, pp. 1–7, 2016.

[2] Y. Wang. (2015, Nov.) The initial definition of dew computing. Dew Computing Research. [Online]. http://www.dewcomputing.org/index.php/2015/11/10/the-initialdefinition-of-dew-computing/

[3] Y. Wang, “Cloud-dew architecture,” International Journal of Cloud Computing, vol. 4, no. 3, pp. 199–210, 2015.

[4] D. Bradley. (2015, Sep.) Dew helps ground cloud services. Science Spot. [Online]. Available: http://sciencespot.co.uk/dew-helps-groundcloud-services.html

[5] Y. Wang and Y. Pan, "Cloud-dew architecture: realizing the potential of distributed database systems in unreliable networks," in Proc. The 21st International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA15), Las Vegas, USA, Jul. 2015, pp. 85–89.

[6] Y. Wang. (2015, Nov.) The relationships among cloud computing, fog computing, and dew computing. Dew Computing Research. [Online]. http://www.dewcomputing.org/index.php/2015/11/12/therelationships-among-cloud-computing-fog-computing-and-dewcomputing/

[7] Z. Kang, "A new method to implement ldns in the cloud-dew architecture," University of Prince Edward Island, CSIT Research Report CS-21, Nov. 2015.

[8] K. Skala, D. Davidovic, E. Afgan, I. Sovic, and Z. Sojat, "Scalable distributed computing hierarchy: Cloud, fog and dew computing," Open Journal of Cloud Computing, vol. 2, no. 1, pp. 16–24, 2015.

[9] S. Ristov, K. Cvetkov, and M. Gusev, "Implementation of a horizontal scalable balancer for dew computing services," Scalable Computing: Practice and Experience, vol. 17, no. 2, pp. 79–90, 2016.

[10] I. Drago, M. Mellia, M. Munafo, A. Sperotto, R. Sadre, and A. Pras, «Inside dropbox: Understanding personal cloud storage services» in Proc. the ACM conference on Internet measurement conference, Boston, Massachusetts, USA, Nov. 2012, pp. 481–494.

[11] Google drive. Google Inc. [Online]:. https://drive.google.com

[12] Turbotax. Intuit Inc. [Online]: http://turbotax.intuit.ca

[13] Onedrive. Microsoft Corporation. [Online].: https://onedrive.live.com

[14] Work on Google files offline, Google Inc., 2016.

[15] M. Palattella, M. Dohler, A. Grieco, G. Rizzo, J. Torsner, T. Engel, and L. Ladid, "Internet of things in the 5g era: Enablers, architecture and business models," IEEE Journal on Selected Areas in Communications, vol. 34, no. 3, pp. 1–17, 2016.

[16] M. Abdelshkour. (2015, Mar.) Iot, from cloud to fog computing. Cisco Blog. [Online]: http://blogs.cisco.com/perspectives/iotfrom-cloud-to-fog-computing

[17] Web of things at w3c. W3C. [Online].: https://www.w3.org/WoT/