Научный журнал «Костюмология» / Journal of Clothing Science https://kostumologiya.ru

2021, №1, Том 6 / 2021, No 1, Vol 6 https://kostumologiya.ru/issue-1-2021.html

URL статьи: https://kostumologiya.ru/PDF/04TLKL121.pdf

**Ссылка для цитирования этой статьи:**
Гусев А.О., Костылева В.В., Разин И.Б., Муртазина А.Р. Контроль версий в облачной системе автоматизированного проектирования обуви // Научный журнал «Костюмология», 2021 №1, https://kostumologiya.ru/PDF/04TLKL121.pdf (доступ свободный). Загл. с экрана. Яз. рус., англ.

**For citation:**
Gusev A.O., Kostyleva V.V., Razin I.B., Murtazina A.R. (2021). Version control for cloud-based footwear computer aided design system. *Journal of Clothing Science*, [online] 1(6). Available at: https://kostumologiya.ru/PDF/04TLKL121.pdf (in Russian)

**УДК 685.34.01, 658.512.2, 658.512.26:004.9**

**ГРНТИ 64.41.14, 64.01.85**

**ББК 37.255**

**Гусев Александр Олегович**
ФГБОУ ВО «Российский государственный университет имени А.Н. Косыгина (Технологии. Дизайн. Искусство)», Москва, Россия
Аспирант
E-mail: alex@gusev.xyz

**Костылева Валентина Владимировна**
ФГБОУ ВО «Российский государственный университет имени А.Н. Косыгина (Технологии. Дизайн. Искусство)», Москва, Россия
Заведующая кафедрой
Доктор технических наук, профессор
E-mail: kostyleva-vv@rguk.ru
РИНЦ: https://elibrary.ru/author_profile.asp?id=353612

**Разин Игорь Борисович**
ФГБОУ ВО «Российский государственный университет имени А.Н. Косыгина (Технологии. Дизайн. Искусство)», Москва, Россия
Заведующий кафедрой
Кандидат технических наук, доцент
E-mail: razin-ib@rguk.ru
РИНЦ: https://elibrary.ru/author_profile.asp?id=850439

**Муртазина Альфия Рустямовна**
ФГБОУ ВО «Российский государственный университет имени А.Н. Косыгина (Технологии. Дизайн. Искусство)», Москва, Россия
Преподаватель
Кандидат технических наук, доцент
E-mail: murtazina-ar@rguk.ru
РИНЦ: https://elibrary.ru/author_profile.asp?id=906389

# Контроль версий в облачной системе автоматизированного проектирования обуви

**Аннотация.** Сегодня, для удовлетворения потребностей клиентов и сохранения конкурентоспособности, производители обуви должны решать две основные задачи: быстро реагировать на рыночные изменения и соответствовать новым потребительским тенденциям. Одним из решений этих задач является применение систем автоматизированного

проектирования (САПР), позволяющих упростить процессы конструирования и моделирования, сокращая время на разработку новой продукции.

Современные САПР обуви отрыты для совершенствования. Ввиду того, что САПР обуви призваны выполнять большие объемы сложных математических вычислений и графических построений, предъявляются высокие требования к аппаратным средствам, что повышает стоимость всей системы. Министерство цифрового развития, связи и массовых коммуникаций утвердило методические рекомендации по импортозамещению программного обеспечения. В рамках этих рекомендаций, отечественные разработчики предлагают операционные системы на базе ядра Linux. Все современные отечественные САПР обуви работают только в операционной системе Windows, и не могут быть запущены в системе Linux без сторонних эмуляторов.

Решением проблем стали облачные технологии, которые сегодня активно развиваются. Облачные приложения распространяются по модели Software as a Service. В рамках этой модели программное обеспечение располагается на удаленном облачном сервере поставщика, а пользователи получают к нему доступ через веб-интерфейс с любого устройства и операционной системы. Модель экономически выгодна как для поставщика, так и для пользователя. А централизация данных, являющаяся следствием применения облачных технологий, открывает новые возможности для организации данных.

Цель исследования состоит в представлении специфики создания системы контроля версий данных для облачной системы автоматизированного проектирования обуви. Для этого в статье рассматриваются основные функциональные возможности современных систем контроля версий, а также научные исследования в области сравнения и объединения данных. Предложена структура организации данных для описания моделей обуви, применение которой позволит выполнить все задачи системы контроля версий. Описан интерфейс взаимодействия с системой контроля версий в облачной среде. Такой подход разрабатывается в рамках диссертации на соискание ученой степени кандидата технических наук Гусевым А.О. и предполагается к внедрению в учебный процесс кафедр «Художественное моделирование, конструирование и технология изделий из кожи», «Информационные технологии» РГУ им. А.Н. Косыгина (Технологии. Дизайн. Искусство) в виде учебных пособий.

**Ключевые слова:** системы автоматизированного проектирования обуви; структуры данных; алгоритмы; централизация данных; системы контроля версий; программный интерфейс обмена данными; распределенные и облачные вычисления

## Преимущества централизации данных для их обслуживания и организации

В процессе использования систем автоматизированного проектирования обуви создается большой объем информации, которая сохраняется на локальных компьютерах. При этом файлы могут быть перенесены между локальными компьютерами по электронной сети или через переносные устройства, в результате чего создается множество копий проекта. После внесения корректировок копии становятся неактуальными и их непредусмотренное применение может привести к проектным ошибкам. Одним из решений этой проблемы может быть централизация данных, которая может быть обеспечена облачным хранилищем.

Централизованное хранение данных позволяет автоматизировать процесс обслуживания данных [1]. Например, для обеспечения возможности быстрого и недорогого восстановления информации необходимо проводить резервное копирование данных. Любой компьютер подвержен риску поломки аппаратного обеспечения и многим другим опасностям, которые могут повлечь потерю проектных данных. Однако резервное копирование эффективно только тогда, когда оно выполняется регулярно, особенно если данные часто меняются. Это вызывает

необходимость автоматизации процесса. Резервное копирование децентрализованных данных невозможно. Пользователи должны самостоятельно предоставить актуальные данные системе для последующего копирования, то есть фактически произвести сбор данных для централизации [2]. Другим примером является архивация. Процесс архивации данных заключается в их сжатии для уменьшения занимаемого пространства в хранилищах данных. Производится данная операция над данными, которые более не являются актуальными, но могут понадобиться в будущем, а также над резервными копиями данных. Алгоритмы сжатия работают с отдельным потоком байтов информации. Соответственно, для применения алгоритмов на множестве различных данных, их необходимо объединить. Чаще всего для этого применяют специальные файлы-контейнеры. Также, как и в случае с резервным копированием, при децентрализованном хранении данных пользователь должен самостоятельно поместить нужную информацию в файл-контейнер для дальнейшей архивации [2].

Помимо автоматизации обслуживания данных, централизация предлагает новые способы работы с данными. Системы организации изменяющейся информации в централизованных хранилищах данных получили наибольшее распространение в сфере разработки программного обеспечения, и называются системами контроля версий. Они позволяют хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение и др. [3]. Наиболее распространенной системой контроля версий является Git, которая была создана в 2005 году Линусом Торвальдсом и сообществом разработчиков. Система контроля версий Git хранит всю информацию в «репозиториях». Работа с Git начинается с центрального репозитория, в котором происходит копирование данных в локальный репозиторий (операция Clone). Далее с помощью операций добавления, изменения и удаления локальная копия преобразуется в переработанный проект. После этого проверяется работоспособность проекта, подтверждаются внесенные изменения (операция Commit), которые переносятся из локального репозитория изменений в центральный (операция Push). В процессе коллективной работы принято использовать ветвление. При создании новой ветки, активная версия полностью копируется, и тогда работа с одной веткой никак не влияет на другую. Также несколько веток можно объединить в одну (операция Merge), при этом происходит слияние всех изменений от основной ветки. Большинство современных систем контроля версий работают по схожему алгоритму [4].

Сегодня подобные системы с успехом применяются в различных областях, где ведется работа с большим количеством непрерывно изменяющихся электронных документов. В частности, системы управления версиями применяются в САПР общего назначения, обычно в составе систем управления данными об изделии (PDM) [5]. Однако многие PDM лишены всех возможностей современных систем контроля версий, применяемых для разработки программного обеспечения. Документы САПР, как правило, являются бинарными файлами. Сравнение векторной информации в бинарном виде, а тем более его автоматическое слияние, как правило, либо затруднено, либо невозможно. Для решения этой проблемы структуры данных, представляющие графическую информацию, должны быть разработаны с учетом работы в системах контроля версий.

## Разработка структуры проектных данных, оптимальной для применения в системе контроля версий, интегрированной в облачную САПР

Большая часть алгоритмов сравнения и объединения данных работает с линейными и древовидными структурами данных. Связанно это преимущественно с решением проблем автоматизации бизнес-процессов, электронного документооборота, коллективной работой. Алгоритмы сравнения линейных данных чаще необходимы для сравнения текстовых данных,



04TLKL121
Издательство «Мир науки» \ Publishing company «World of science» http://izd-mn.com

и реже – бинарных. Одна из первых программ для сравнения текстов «diff» была разработана в начале 1970-х годов для операционной системы Unix. Ее алгоритм основан на решении задачи поиска наибольшей общей подпоследовательности (LCS) [6]. Сегодня разработано множество алгоритмов для решения задачи LCS. Например, группа исследователей из австрийского Института логики и вычислений смогли преобразовать исходную задачу в проблему поиска наикратчайшего пути, и решили ее классическим алгоритмом «A*» [7]. Современные системы сравнения файлов также используют в основе алгоритмы, основанные на возможностях современных информационных технологий. Так, группа исследователей из университетов Мэрилэнда и Иллинойса разработали алгоритм параллельного многопоточного сравнения данных [8].

При автоматизации бизнес-процессов, в качестве формата обмена данными наибольшее распространение получил язык разметки XML. Он разрабатывался как язык с простым формальным синтаксисом, удобный для создания и обработки документов программами и человеком одновременно. XML-документы формируют древовидную структуру, поэтому множество алгоритмов сравнения и объединения данных направленны именно на работу со структурами данных в виде дерева. Ввиду сложности задачи, для ее решения создаются целые фреймворки. Примером такого фреймворка является разработка исследователей из Института Бургундии и Университета Сан-Паулу [9], а также фреймворк «XChange» [10]. Применение древовидных структур данных в условиях контроля версий частое явление. Существует множество исследований, посвященных оценке эффективности различных алгоритмов сравнения и объединения древовидных данных в системах контроля версий [11; 12].

В системах автоматизированного проектирования обуви для описания данных можно применять оба подхода. Линейно данные могут быть представлены в виде цепи. В памяти каждый элемент однонаправленной цепи содержит ссылку на следующий элемент. В двунаправленной цепи элементы содержат ссылку и на предыдущий элемент. Физическим представлением таких данных в файле будет являться последовательность элементов цепи от начала до конца, с разделителями, отражающими начало и конец элемента. Однако, применение линейных структур данных осложнено их поддержкой. Это касается процесса внесения в систему новых элементов и изменение существующих [13].

Наиболее подходящей формой данных для САПР обуви является древовидная структура, основанная на различных уровнях, получивших в сфере разработки программного обеспечения название «уровни абстракции». Самый нижний, первый уровень абстракции – это графическая информация, описывающая кривые и прямые линии. В САПР обуви наибольшее распространение получили такие кривые третьего порядка, как последовательно соединенные кривые Безье, сплайн Эрмита, B-сплайн и натуральный сплайн [14–16]. Вторым уровнем абстракции являются независимые основные и вспомогательные линии. Третий уровень абстракции составляют элементы, зависимые от второго уровня – это линии припусков, наколы, гофры, перфорации. Четвертый уровень абстракции – это симметрия. Последний, пятый уровень абстракции – это деталь обуви, объединяющая множество элементов второго, третьего и четвертого уровней абстракции. Таким образом каждый следующий уровень объединяет предыдущие. Связи между уровнями наглядно представлены на рисунке 1. Такая структура физически может быть представлена в виде файлов: XML, YAML, JSON, ProtoBuf и других.

Для повышения эффективности выполнения операции сопоставления деревьев, каждый узел должен иметь уникальный идентификатор [11]. Для определения уникальности узла в рамках одного документа идентификатором может стать целое беззнаковое число, увеличивающееся на единицу после создания нового узла. Максимальное значение такого числа размером в 4 байта будет 235, что будет являться максимальным количеством элементов в одном документе.

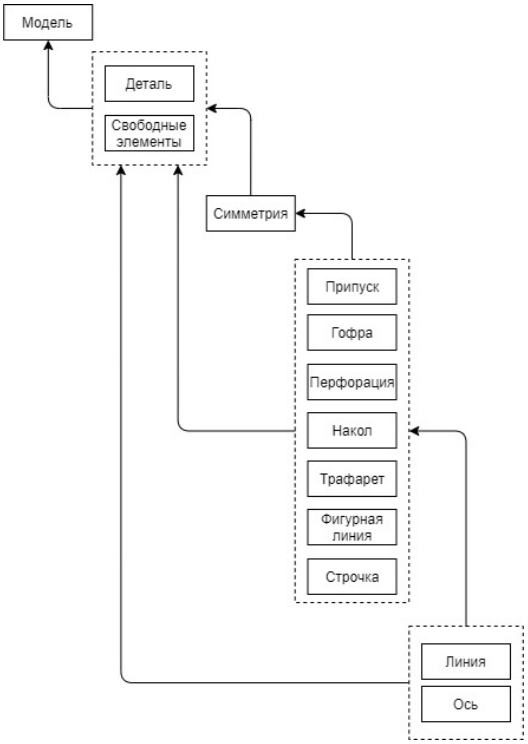


***Рисунок 1.*** *Разработанная авторами диаграмма связей элементов модели обуви*

Для обеспечения большей уникальности чаще всего генерируется UUID (universally unique identifier) – это стандарт идентификации, используемый в создании программного обеспечения, стандартизированный Open Software Foundation, как часть среды распределённых вычислений. Основное назначение UUID предоставление возможности распределённым системам уникально идентифицировать информацию без центра координации и необходимости разрешения конфликта имен. UUID представляет собой 16-байтное число, что обеспечивает 25616 возможных вариантов ключей [17].

Применяя такую структуру данных, а также существующие алгоритмы сравнения и объединения данных, можно построить систему контроля версий, интегрированную в облачную САПР обуви. Сущностями такой системы станут: репозиторий – место хранения данных, это может быть файловая система или база данных, Репозиторий хранит множество проектов; проект – модель обуви, содержит как минимум одну ветку; ветка – вариант разрабатываемой модели, состоит из набора версий; версия – набор данных, описывающих модель обуви.

## Концепция программного интерфейса облачного приложения для организации доступа к данным

Основным подходом для разработки облачного приложения является создание серверных компонентов, не имеющих состояние. Под stateless-серверной частью понимается, что на сервере не сохраняются промежуточные данные, а только их конечное состояние. Каждый запрос передает необходимые параметры для выполнения операции. Stateless-реализация серверной части существенно упрощает разработку и поддержку приложения, особенно серверной части. За счёт такой реализации серверной части достигается высокая гибкость горизонтального масштабирования приложения [18]. Применяя этот подход, система будет иметь несколько больше операций, чем классическая система контроля версий.

Работа начинается с создания проекта. Запрос, отвечающий за данную операцию должен создать не только проект, но и пустую главную ветку. Также работа может начинаться с выбора проекта, с фильтрацией или без. Запрос на получение списка проектов должен возвращать множество идентификаторов проекта с описательной информацией о проекте. Так как проект содержит ветки, то на следующем шаге необходимо выбрать нужную ветку. Для этого отправляется запрос на получение веток проекта, а в качестве параметра запроса передается идентификатор проекта. Ответом на этот запрос будет список веток с их идентификаторами в выбранном проекте. Пользователь может получить данные любой версии ветки, соответственно запрос на получение списка версий, аналогично описанным ранее запросам, должен принимать в качестве параметров идентификаторы проекта и ветки. Ответом станет список идентификаторов версий. Запрос на чтение, предоставляющий геометрию модели обуви, требует полный набор идентификаторов: проекта, ветки и версии. После работы с данными, пользователю необходимо сохранить свой результат. Новые данные передаются в запросе на запись вместе с идентификаторами проекта и ветки, после чего создается новая версия в выбранной ветке проекта, с измененными данными. Запрос на создание новой ветки должен принимать геометрические данные, вместе с идентификаторами проекта и оригинальной ветки. Полученные данные становятся первой версией в новой ветке. Запрос на слияние веток принимает идентификаторы проекта и двух объединяемых веток. В ответе на запрос содержится информация о результате сравнения последних версий обрабатываемых веток совместно с версией, на которой произошло разветвление. Успешное автоматическое слияние возможно только в том случае, если после разветвления в ветках изменились разные части геометрических данных или эти изменения были одинаковыми. В противном случае, пользователь, объединяющий ветки, должен выбрать предпочтительный вариант самостоятельно. Для этого клиентское приложение должно предоставить пользователю соответствующий интерфейс ручного разрешения конфликта, а серверная часть – все данные о сравнении.

## Выводы

В результате централизации данных в системе автоматизированного проектирования обуви за счет облачных вычислений появляется возможность их организации при помощи систем контроля версий. Рассмотренные алгоритмы сравнения и слияния хотя и предъявляют строгие требования к форме данных, но допускают создание такую систему. Показано, что разработанная древовидная структура данных позволяет не только быть объектом применения этих алгоритмов, но и является наиболее подходящей формой представления элементов САПР обуви. Помимо данных детальной проработки интерфейс требует доступа к ним. Для масштабирования облачных систем применяется stateless-подход, при котором каждый компонент системы принимает в запросе ровно столько информации, сколько ему необходимо для выполнения действия. При таком подходе предложенная концепция программного интерфейса описывает операции системы контроля версий.

# ЛИТЕРАТУРА

1. Giachetti R.E. Design of enterprise systems: Theory, architecture, and methods. – CRC Press, 2016.
2. Zhao Y., Lu N. Research and Implementation of Data Storage Backup // 2018 IEEE International Conference on Energy Internet (ICEI). – IEEE, 2018. – С. 181–184.

3. Шурыгин В.Н., Сиваченко Д.А. Системы контроля версий // Вестник Московского государственного университета печати. – 2015. – №. 5.
4. Ольховская А.А., Усенко Р.С. Системы контроля версий и их применение на различных этапах жизненного цикла ИС // Теория и практика экономики и предпринимательства. – 2020. – С. 213–214.
5. Scheidel W., Mozgova I., Lachmayer R. Product data management in the context of Industry 4.0 // 59th Ilmenau Scientific Colloquium. – 2017. – С. 11–15.
6. Spinellis D. A repository of Unix history and evolution //Empirical Software Engineering. – 2017. – Т. 22. – №. 3. – С. 1372–1404.
7. Djukanovic M. et al. An A⋆ search algorithm for the constrained longest common subsequence problem // Information Processing Letters. – 2020. – С. 106041.
8. Hajiaghayi M.T., Seddighin S., Sun X. Massively parallel approximation algorithms for edit distance and longest common subsequence // Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms. – Society for Industrial and Applied Mathematics, 2019. – С. 1654–1672.
9. Tekli J., Chbeir R. A novel XML document structure comparison framework based-on sub-tree commonalities and label semantics // Journal of Web Semantics. – 2012. – Т. 11. – С. 14–40.
10. Oliveira A. et al. XChange: A semantic diff approach for XML documents // Information Systems. – 2020. – Т. 94. – С. 101610.
11. Thao C., Munson E.V. Using versioned trees, change detection and node identity for three-way XML merging // SICS Software-Intensive Cyber-Physical Systems. – 2019. – Т. 34. – №. 1. – С. 3–16.
12. Asenov D. et al. Precise version control of trees with line-based version control systems // International Conference on Fundamental Approaches to Software Engineering. – Springer, Berlin, Heidelberg, 2017. – С. 152–169.
13. Пасечников К.А. Модели структур данных с векторной, списковой и древовидной организацией элементов // Наука и образование: научное издание МГТУ им. НЭ Баумана. – 2008. – №. 10. – С. 8–8.
14. Муртазина А.Р., Миронов В.П., Разин И.Б., Тихонова К.Н. Интерполяция точек кубического сплайна методом половинного деления // Научный журнал «Дизайн и технологии». – 2010. – с. 36–39.
15. Муртазина А.Р., Миронов В.П., Разин И.Б. Приближение к классическому сплайну в 2D // Научный журнал «Дизайн и технологии». – 2011. – с. 41–46.
16. Мазиков А.В., Миронов В.П., Муртазина А.Р. Четырёхточечная интерполирующая кубическая схема для проектирования кривых // Научный журнал «Дизайн и технологии». – 2013. – с. 75–79.
17. Богомья Д.И. Оптимизация хранения UUID // материалы международной научной конференции «Информационные технологии и системы». – Минск, 2015. – С. 218–219.
18. Хохряков И.А. Разработка распределённого Web-приложения // RSDN Magazine. – 2011. – №. 4. – С. 49–57.

**Gusev Alexander Olegovich**
Russian state university named A.N. Kosygin (Technologies. Design. Art), Moscow, Russia
E-mail: alex@gusev.xyz

**Kostyleva Valentina Vladimirovna**
Russian state university named A.N. Kosygin (Technologies. Design. Art), Moscow, Russia
E-mail: kostyleva-vv@rguk.ru
РИНЦ: https://elibrary.ru/author_profile.asp?id=353612

**Razin Igor Borisovich**
Russian state university named A.N. Kosygin (Technologies. Design. Art), Moscow, Russia
E-mail: razin-ib@rguk.ru
РИНЦ: https://elibrary.ru/author_profile.asp?id=850439

**Murtazina Alfiya Rustyamovna**
Russian state university named A.N. Kosygin (Technologies. Design. Art), Moscow, Russia
E-mail: murtazina-ar@rguk.ru
РИНЦ: https://elibrary.ru/author_profile.asp?id=906389

# Version control for cloud-based footwear computer aided design system

**Abstract.** Footwear manufacturers need to respond quickly to market changes and new trends in order to meet customer needs and remain competitive. One of the solutions to these problems is the use of computer-aided design (CAD) systems that simplify the design processes, reducing the time for developing new products.

Modern CAD footwear has the potential for improvement. For example, hardware requirements are high due to the fact that CAD systems for shoes are designed to perform large amounts of complex mathematical calculations and graphics. This increases the cost of the entire system. Another example is the cross-platform capability. The Ministry of Digital Development, Communications and Mass Media of the Russian Federation approved guidelines for import substitution of software. As part of these guidelines, developers are proposing operating systems based on the Linux kernel. All modern CAD shoes run only on the Windows operating system, and cannot be run on a Linux system without third-party emulators.

Cloud technologies, which are actively developing today, became the solution to the problems. Cloud applications are distributed according to the Software as a Service model. Under this model, the software is hosted on a remote cloud server of the vendor, and users can access it via a web interface from any device and operating system. The model is cost effective for both the supplier and the user.

The article discusses the specifics of creating a version control system for a cloud-based automated shoe design system. The main functionality of modern version control systems, as well as research in the field of data comparison and merging, are presented. A data structure is proposed for describing shoe models, the use of which will allow performing all the tasks of the version control system. The interface for interaction with the version control system in the cloud is described. This system is being developed by Gusev A.O. for the dissertation for the degree of Candidate of Engineering Sciences and it is supposed to be introduced into the educational process of the departments "Artistic modeling, design and technology of leather goods" and the department of "Information technology" Kosygin RSU (Technology. Design. Art) in the form of teaching aids.

**Keywords:** footwear computer aided design systems; data structures; algorithms; data centralization; version control systems; application programming interface; distributed and cloud computing

## REFERENCES

1. Giachetti R.E. Design of enterprise systems: Theory, architecture, and methods. – CRC Press, 2016.
2. Zhao Y., Lu N. Research and Implementation of Data Storage Backup // 2018 IEEE International Conference on Energy Internet (ICEI). – IEEE, 2018. – S. 181–184.
3. Shurygin V.N., Sivachenko D.A. Sistemy kontrolya versiy // Vestnik Moskovskogo gosudarstvennogo universiteta pechati. – 2015. – №. 5.
4. Ol′khovskaya A.A., Usenko R.S. Sistemy kontrolya versiy i ikh primenenie na razlichnykh ehtapakh zhiznennogo tsikla IS // Teoriya i praktika ehkonomiki i predprinimatel′stva. – 2020. – S. 213–214.
5. Scheidel W., Mozgova I., Lachmayer R. Product data management in the context of Industry 4.0 // 59th Ilmenau Scientific Colloquium. – 2017. – S. 11–15.
6. Spinellis D. A repository of Unix history and evolution //Empirical Software Engineering. – 2017. – T. 22. – №. 3. – S. 1372–1404.
7. Djukanovic M. et al. An A⋆ search algorithm for the constrained longest common subsequence problem // Information Processing Letters. – 2020. – S. 106041.
8. Hajiaghayi M.T., Seddighin S., Sun X. Massively parallel approximation algorithms for edit distance and longest common subsequence // Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms. – Society for Industrial and Applied Mathematics, 2019. – S. 1654–1672.
9. Tekli J., Chbeir R. A novel XML document structure comparison framework based-on sub-tree commonalities and label semantics // Journal of Web Semantics. – 2012. – T. 11. – S. 14–40.
10. Oliveira A. et al. XChange: A semantic diff approach for XML documents // Information Systems. – 2020. – T. 94. – S. 101610.
11. Thao C., Munson E.V. Using versioned trees, change detection and node identity for three-way XML merging // SICS Software-Intensive Cyber-Physical Systems. – 2019. – T. 34. – №. 1. – S. 3–16.
12. Asenov D. et al. Precise version control of trees with line-based version control systems // International Conference on Fundamental Approaches to Software Engineering. – Springer, Berlin, Heidelberg, 2017. – S. 152–169.
13. Pasechnikov K.A. Modeli struktur dannykh s vektornoy, spiskovoy i drevovidnoy organizatsiey ehlementov // Nauka i obrazovanie: nauchnoe izdanie MGTU im. NEH Baumana. – 2008. – №. 10. – S. 8–8.
14. Murtazina A.R., Mironov V.P., Razin I.B., Tikhonova K.N. Interpolyatsiya tochek kubicheskogo splayna metodom polovinnogo deleniya // Nauchnyy zhurnal «Dizayn i tekhnologii». – 2010. – c. 36–39.
15. Murtazina A.R., Mironov V.P., Razin I.B. Priblizhenie k klassicheskomu splaynu v 2D // Nauchnyy zhurnal «Dizayn i tekhnologii». – 2011. – c. 41–46.
16. Mazikov A.V., Mironov V.P., Murtazina A.R. Chetyryokhtochechnaya interpoliruyushchaya kubicheskaya skhema dlya proektirovaniya krivykh // Nauchnyy zhurnal «Dizayn i tekhnologii». – 2013. – c. 75–79.
17. Bogom′ya D.I. Optimizatsiya khraneniya UUID // materialy mezhdunarodnoy nauchnoy konferentsii «Informatsionnye tekhnologii i sistemy». – Minsk, 2015. – S. 218–219.
18. Khokhryakov I.A. Razrabotka raspredelyonnogo Web-prilozheniya // RSDN Magazine. – 2011. – №. 4. – S. 49–57.