Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт математики и фундаментальной информатики
Базовая кафедра вычислительных и информационных технологий

УТВЕРЖДАЮ
Заведующий кафедрой
_____ / В.В. Шайдуров
«____» _____ 2018 г.

БАКАЛАВРСКАЯ РАБОТА

Направление 02.03.01 Математика и компьютерные науки

АВТОМАТИЗИРОВАННОЕ ОБНОВЛЕНИЕ ИЗМЕНЕННЫХ КОНФИГУРАЦИЙ 1С

Научный руководитель
кандидат физико-математических наук,
доцент _____ / Д.А. Цыганок

Выпускник _____ / М.В. Гладких

Красноярск 2018

РЕФЕРАТ

Выпускная квалификационная работа на тему «Автоматизированное обновление измененных конфигураций 1С» содержит 45 страниц текста, 9 иллюстраций, 4 приложения, 13 использованных источников.

ОБНОВЛЕНИЕ КОНФИГУРАЦИЙ 1С, АВТОМАТИЗАЦИЯ, ПРОГРАММИРОВАНИЕ, ТЕХНОЛОГИЧЕСКАЯ ПЛАТФОРМА 1С.

Цель работы – разработка программного продукта, позволяющего автоматизировать обновление измененных типовых конфигураций. Основная задача данного приложения: перенос изменений из кода старого модуля измененной конфигурации в новый модуль типовой конфигурации поставщика.

В результате исследований и разработки было реализовано приложение со всем необходимым функционалом, а также средством для редактирования кода старого модуля. Данное программное решение может быть полезно для программистов 1С, которые хотят ускорить переход на новую конфигурацию и при этом минимизировать ошибки.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Большинство предприятий при осуществлении своей деятельности стремятся упростить и автоматизировать различные бизнес-процессы, а также бухгалтерский и управленческий учет. В решении этой задачи активно помогают информационные технологии, в частности технологическая платформа «1С:Предприятие».

Очевидно, что деятельность предприятий может быть довольно разнообразной, поэтому система 1С может «приспосабливаться» к особенностям конкретной области деятельности. Это достигается благодаря тому, что «1С: Предприятие» существует как совокупность различных программных инструментов, с которыми работают разработчики и пользователи. Но как бы 1С ни старалась быть гибкой, в процессе использования типовых конфигураций неизбежно возникает потребность вносить изменения и дорабатывать конфигурацию под индивидуальные нужды использующего ее предприятия. В этом случае ее становится гораздо сложнее обновить до новой версии из-за утраты совместимости с оригинальной типовой конфигурацией. В самой 1С механизм обновления несовершенен и неудобен, из-за чего программистам приходится вручную переносить большинство изменений.

Таким образом, главной целью данной работы является написание внешнего приложения для системы 1С, которое позволит частично автоматизировать процесс обновления измененных типовых конфигураций и перенос изменений из старой конфигурации в новую.

Для достижения данной цели были поставлены и решены следующие задачи:

1) Изучение встроенных в «1С:Предприятие» возможностей по упрощению и автоматизации перевода конфигурации на новую версию;

2) Рассмотрение имеющихся на данный момент внешних решений, автоматизирующих процесс обновления конфигураций;

3) Разработка и программная реализация собственного алгоритма для автоматического сравнения и объединения конфигураций на языке программирования C++;

4) Реализация взаимодействия приложения и системы 1С для получения необходимых для объединения данных;

5) Реализация пользовательского интерфейса;

6) Тестирование полученного программного продукта на реальных конфигурациях в системе «1С:Предприятие 8.3»;

Поскольку до сих пор не существует широко распространенного и дешевого решения для автоматизации обновления измененных типовых конфигураций, решение поставленной проблемы является актуальным. Оно поможет программистам 1С сэкономить время и нервы, а также избавит их от ошибок из-за человеческого фактора.

При написании данной работы были использованы труды следующих авторов: Нуралиев, Б.Г., Байдаков, В, Радченко, М.Г, Низамутдинов И., Пахомов Б., Толстобров А.А., Хромых В.Г., Шилдт, Г., Хоггенсон, Г., Темплеман, Дж., Рихтер Дж., Хортон, А.

Структурно выпускная квалификационная работа состоит из реферата, введения, двух глав, восьми пунктов, девяти подпунктов, заключения, списка используемых источников, четырех приложений.

Глава 1 Основные понятия системы 1С и способы обновления измененных типовых конфигураций

1.1 Основные определения

В данном пункте вводятся основные понятия из технологической платформы 1С, необходимые для дальнейшей работы.

Платформа - среда исполнения приложений «1С:Предприятие» и набор средств для разработки приложений и администрирования [1].

Конфигуратор - один из режимов работы платформы, в котором разрабатываются, администрируются и обновляются прикладные решения [1].

Конфигурация - в системе «1С:Предприятие» это прикладное решение, созданное с помощью конфигуратора и исполняемое платформой, состоящее из взаимосвязанных составных частей: подсистем, структур учетных данных, набора процедур, функций, макетов и т.п [1, 5]. Все эти части тесно связаны между собой и требуют согласованного внесения изменений. В сущности, структура конфигурации является проекцией предметной области в информационное пространство.

В информационной базе системы «1С:Предприятие» хранятся конфигурации двух видов:

- основная (редактируемая) конфигурация;
- конфигурация базы данных.

Конфигурация базы данных определяет структуру таблиц базы данных и всю функциональность, с которой работают пользователи. Основная конфигурация используется только для внесения изменений.

В процессе редактирования конфигурации могут быть созданы новые, изменены или удалены существующие объекты, в том числе и подчиненные объекты (формы, реквизиты и т. д.).

Метаданные – это структура конфигурации базы данных, в которой описываются объекты прикладного решения.

Объект конфигурации – формальное описание группы понятий со сходными характеристиками и одинаковым предназначением [1]. Проще говоря, это элементы, из которых складывается прикладное решение.

Как правило, объекты конфигурации являются компьютерными аналогами реально существующих на предприятии видов документов. Например, справочника сотрудников или номенклатуры товаров, однако могут использоваться и для организации конструкций, не имеющих явных физических аналогов. Важно понимать, что объект конфигурации описывает не конкретное значение, а только его вид.

Модуль - это программа на встроенном языке системы 1С [1].

Модули располагаются в заданных точках структуры конфигурации и вызываются для выполнения в заранее известные моменты работы системы.

В конфигурации существует несколько видов модулей. Это модуль управляемого приложения, модуль обычного приложения, модуль внешнего соединения, модуль сеанса, общие модули, модули форм т.д. Некоторые объекты (например, константы) не имеют модуля.

Именно с данным объектом конфигурации будет вестись работа в реализуемом автором данной работы приложении.

Типовая конфигурация - это универсальное прикладное решение, разработанное компанией 1С и ее представителями (далее поставщиками) для конкретной предметной области, например, бухгалтерии. Данный тип конфигураций находится на постоянной поддержке поставщика. Это означает, что клиент может пользоваться новыми версиями типовой конфигурации после обновления.

В случае, если в типовую конфигурацию не вносились изменения, обновление происходит максимально комфортно и практически не требует участия пользователя. С обновлением конфигураций данного типа 1С успешно справляется сама [1].

Нетиповая конфигурация - это конфигурация, полностью разработанная программистом 1С для какой-то специфической или

узкоспециализированной области деятельности, либо в целях обучения программированию на 1С.

Прикладные решения такого рода не поддерживаются поставщиком, и, соответственно, могут дорабатываться только самим программистом посредством внесения изменений вручную.

Измененная типовая конфигурация – типовая конфигурация, в которую были внесены какие-либо сторонние изменения.

Часто необходимость таких изменений диктуется спецификой конкретной предметной области и личными пожеланиями заказчика. В этом случае конфигурация снимается с поддержки поставщика и обновление до нового релиза требует участия программиста. Это трудоемкая и кропотливая работа, постоянно возникают ошибки из-за невнимательности. Поэтому процесс обновления таких конфигураций требует автоматизации. Именно для данного типа конфигураций и предназначено разрабатываемое приложение.

1.2 Способы обновления измененных типовых конфигураций

Поставщик постоянно совершенствует прикладные решения и добавляет изменения в типовые конфигурации в соответствии с новыми стандартами, законами и т.д. В определенный момент он выпускает так называемые «новые релизы». На эти новые версии типовой конфигурации и обновляется заказчик.

В случае с типовыми конфигурациями никаких проблем при обновлении не возникает, и система 1С все делает автоматически [2]. Но если были внесены какие-то изменения, особенно если были затронуты глубокие структуры конфигурации, например, модули или формы, то обязательно возникнут сложности при обновлении и потребуются ручные доработки обновленной конфигурации. Задача программиста в этом случае состоит в том, чтобы перенести сторонние изменения в новую конфигурацию на свои места в целости и сохранности, но при этом все изменения поставщика, содержащиеся в файле обновления, тоже должны быть применены. В случае большого числа

доработок, решение этой задачи становится очень затратным по времени и трудоемким.

1.2.1 Обновление с помощью 1С

Как уже было сказано выше, 1С справляется с обновлением полностью автоматически только в случае типовых конфигураций. Если же в старую конфигурацию были внесены сторонние изменения, при обновлении такой конфигурации появляется дерево сравнения основной конфигурации (с изменениями) и новой конфигурации поставщика (рис. 1), где знаком отмечаются объекты с изменениями [2]. В этом же дереве сравнения конфигураций можно выбрать те объекты, которые нужно объединить и установить режим их объединения. Кроме того, для каждого объекта можно просмотреть детальную информацию об отличиях. Например, можно увидеть различия в коде модулей объекта (рис. 2).

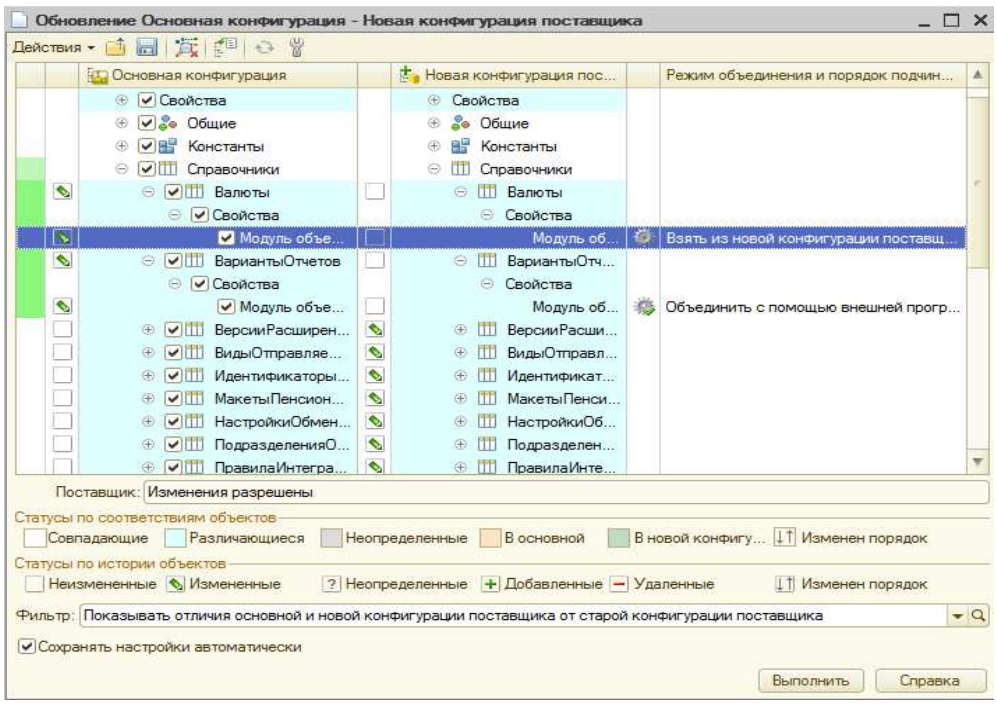

Рисунок 1 - Окно сравнения конфигураций

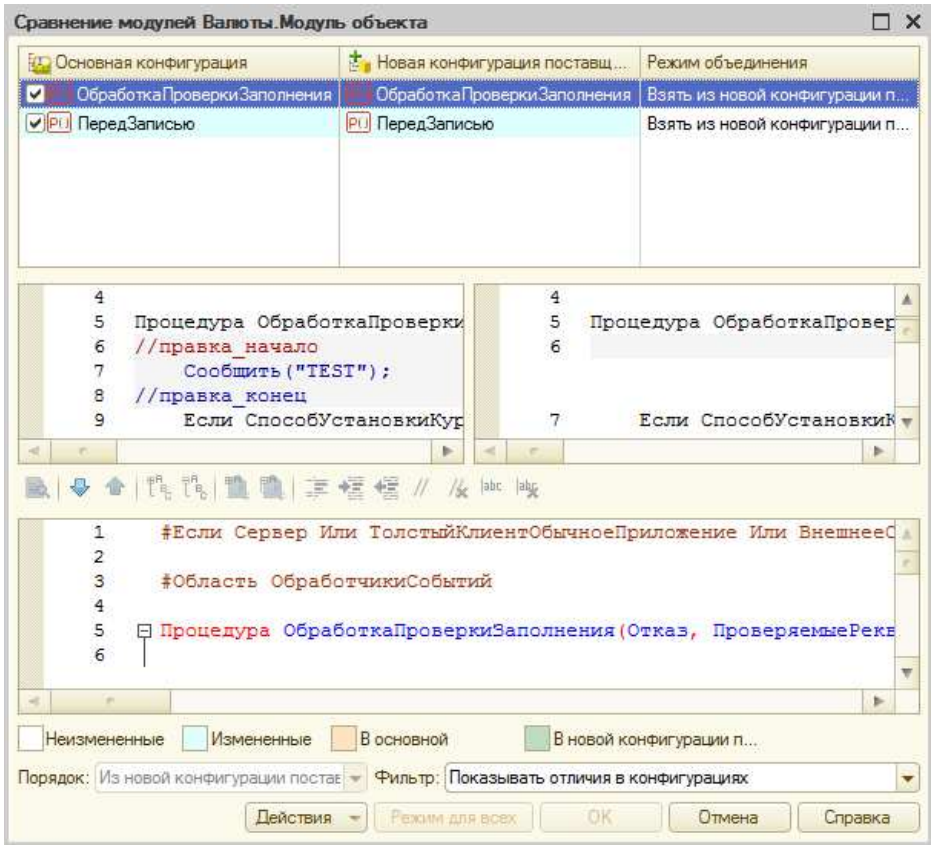

Рисунок 2 - Различия в модулях объекта

К недостаткам такого способа обновления конфигурации относится то, что в качестве режима объединения можно выбрать либо приоритет новой конфигурации, либо приоритет старой. В первом случае теряются все собственные изменения и их потом необходимо будет добавлять вручную. Во втором случае теряются изменения поставщика, которые тоже впоследствии нужно вносить вручную. То есть приходится выбирать меньшее из двух зол в каждом конкретном случае, в зависимости от того каких изменений меньше – собственных или поставщика. И тот и другой случаи требуют много последующей ручной работы.

1.2.2 Обновление с помощью внешних программ

1.2.2.1 Программы для сравнения и объединения модулей

Для сравнения и объединения модулей объектов измененной конфигурации и конфигурации поставщика существует ряд сторонних приложений, таких как Araxis Merge, DiffMerge, Kdiff3, TortoiseMerge, Perforce P4Merge. По сути своей они предназначены для сравнения текстовых файлов. Функционал у них примерно такой же, как и у встроенного в 1С. Эти программы можно подключить в настройках конфигуратора, если по каким-то причинам не устраивают встроенные возможности системы «1С Предприятие». Они получают доступ к модулям объектов, сравнивают их и объединяют.

К недостаткам такого решения можно отнести:

1) Программы не предназначены для сравнения модулей на языке 1С;
2) Сложный интерфейс на английском языке, нет российских аналогов;
3) Медленная работа, особенно при работе с большими модулями;

1.2.2.2 Программный продукт для комплексного сравнения и объединения

На данный момент автору данной работы удалось найти только один программный продукт, который позволяет автоматически выявить и корректно внести 99% изменений типовой конфигурации, причем не только код модулей, но и практически любой другой объект. Данный продукт под названием «1С: Автоматизированное обновление измененных конфигураций» на протяжении 8 лет разрабатывала компания «1С-ИжТиСи», которая тесно сотрудничает с фирмой «1С» [3].

В этот программный комплекс заложены такие функции, как возможность сравнения до 4-х конфигураций одновременно, просмотр всех

изменений, учет и проверка синтаксиса, проверка метаданных и тестирование обновленной конфигурации.

Несомненно, это лучшее решение для максимального облегчения и ускорения процесса обновления конфигураций на данный момент, но и у него есть недостатки. Во-первых, это цена. Стоимость профессиональной версии составляет 117 600 рублей, что далеко не каждому предприятию по карману. Во-вторых, у продукта очень высокие системные требования к компьютеру, на котором будет выполняться обновление. Для сравнительно быстрой работы требуются 2 накопителя SSD объемом не менее 120 Гб, процессор intel Core i7, оперативная память не менее 16 Гб, а также специальные настройки операционной системы. В-третьих, сложный интерфейс из-за обилия функционала.

Резюмируя, можно сказать, что данный программный продукт идеально подходит для крупных предприятий, у которых сложнейшие конфигурации с тысячами доработок. Они могут позволить себе приобрести его и дорогостоящее оборудование. Для небольших предприятий данный продукт использовать нецелесообразно.

1.2.3 Использование внешних отчетов и обработок

Чтобы не затрагивать своими изменениями объекты типовой конфигурации, можно отдельно от прикладного решения создавать внешние отчеты и обработки. В этом случае при обновлении конфигурации внешние файлы остаются нетронутыми, обновление проходит в автоматическом режиме, после чего эти файлы просто используются с новой обновленной конфигурацией [4, 5].

Недостатками такого подхода является его малая гибкость, не все возможные изменения можно вынести в отчеты и обработки, часто значительно проще, а иногда и невозможно по-другому, внести исправления в существующие объекты типовой конфигурации. Также все эти внешние файлы

будут занимать дополнительное место на компьютере, что негативно скажется на быстродействии системы 1С и компьютера в целом.

1.2.3 Механизм расширения конфигураций

Чтобы как-то бороться с трудностями обновления измененных типовых конфигураций, кроме стандартных средств полнотекстового поиска и ручной замены на стадии анализа объединяемых конфигураций, разработчиками 1С был предложен механизм расширения конфигураций [6, 7]. Расширение это по сути копия типовой конфигурации, в которую можно вносить изменения, при этом оставляя основную конфигурацию без изменений и, что самое главное, на поддержке. В результате обновление типовой конфигурации происходит полностью в автоматическом режиме. А при запуске обновлённого прикладного решения платформа автоматически объединит изменённую типовую конфигурацию с расширением (рис. 3).

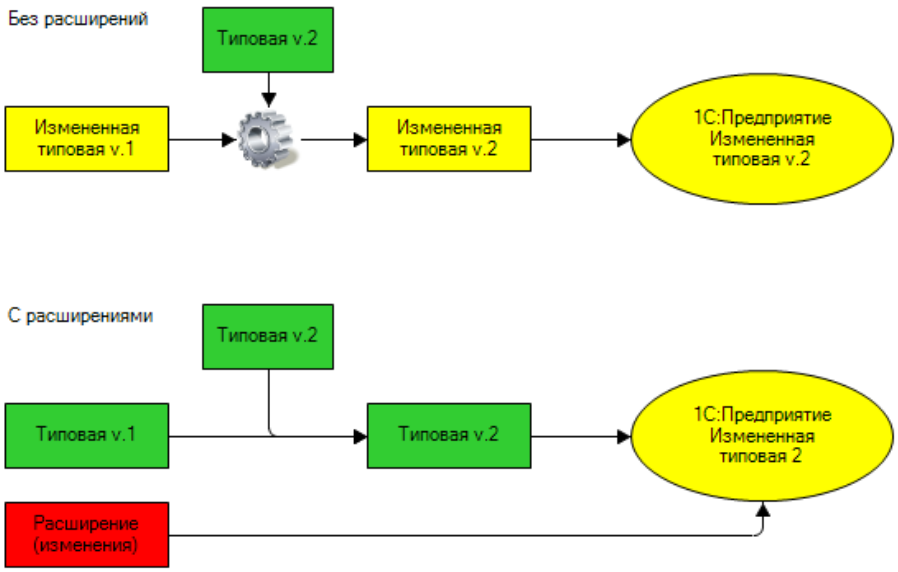

Рисунок 3 – Механизм расширения конфигурации

Этот механизм был предложен в конце 2014 года. Им мало пользовались, потому что на первоначальном этапе функционал был ограничен. Однако

сейчас использование этого варианта растет, так как фирма 1С активно развивает его, и с каждой новой версией платформы механизм расширений становится все более удобным и функциональным.

Но и у этого варианта есть недостаток. В расширении можно частично изменить только те объекты, которые уже есть в конфигурации, и то далеко не все.

Считается, что будущее платформы 1С как раз-таки за расширениями, которые позволят избегать ошибок и потерь информации при обновлении.

Глава 2 Авторский подход к автоматизации обновлений

Очевидно, что решение проблемы переноса изменений из старой конфигурации в новую со временем решат сами разработчики 1С с помощью механизма расширений. Но, по мнению автора дипломной работы, на данный момент программистам нужно решение, которое позволит хотя-бы частично автоматизировать процесс обновления. Выше приведен обзор подобных решений, в ходе которого у всех них выявились недостатки. Поэтому было принято решение написать собственное внешнее приложение для технологической платформы «1С: Предприятие», которое сможет автоматически переносить изменения программного кода модулей из измененной типовой конфигурации в новую.

2.1 Требования к программе

На основе анализа вышеперечисленных способов обновления был составлен список требований к разрабатываемой программе. Итоговое приложение должно удовлетворять следующим условиям:

- Простота интерфейса;
- Доступность и дешевизна;
- Быстродействие;
- Минимальные системные требования;
- Работа с модулями объектов;
- Учет синтаксиса языка 1С;
- Возможность указывать свои комментарии;
- Возможность вносить изменения в код старого модуля во время сравнения;

2.2 Выбор среды разработки и языка программирования

Для написания приложения был выбран профессиональный объектно-ориентированный язык программирования C++/CLI, используемый для разработки высокопроизводительного программного обеспечения. Данная версия языка программирования C++ позволяет создавать приложения для среды программирования Microsoft.NET с интерфейсом Windows Forms [8]. Это важно для разрабатываемого приложения, так как пользователю будет удобнее взаимодействовать с 1С через продуманный интерфейс.

В качестве интегрированной среды разработки (IDE) была использована платформа Microsoft Visual Studio 2017 Community Edition. Эта IDE бесплатна, обладает достаточно мощным функционалом, довольно удобна в управлении, обладает редактором форм и позволяет подключать сторонние библиотеки и плагины [9].

2.2 Структура программы

Прежде чем приступать к написанию кода, необходимо определиться с методологией программирования. Так как программа не предусматривает большого числа объектов, нет необходимости использовать объектно-ориентированное программирование, наличие классов только усложнит разработку. В связи с относительной простотой кода был выбран метод структурного программирования. В основу структурного программирования положены следующие достаточно простые положения [10]:

- алгоритм и программа должны составляться поэтапно (по шагам);
- сложная задача должна разбиваться на достаточно простые части, каждая из которых имеет один вход и один выход;
- логика алгоритма и программы должна опираться на минимальное число достаточно простых базовых управляющих структур.

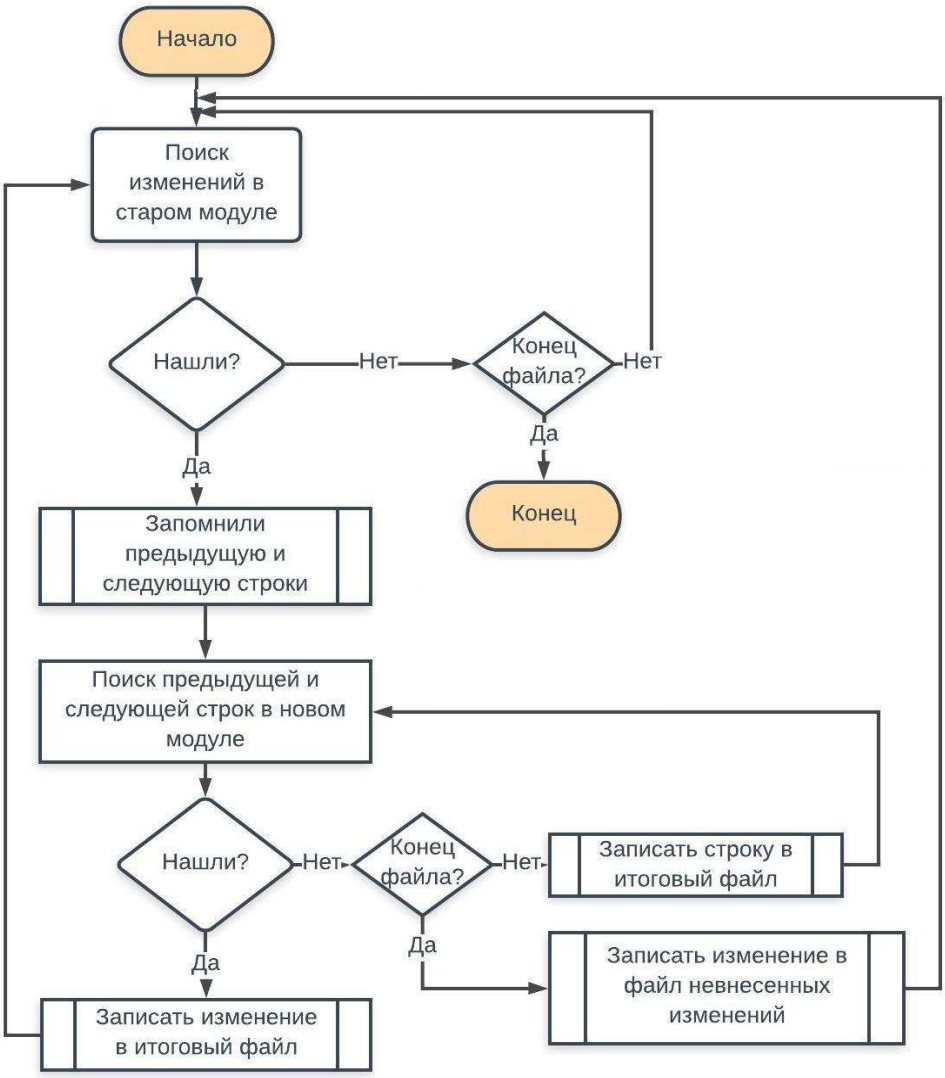

Рисунок 4 - Блок-схема алгоритма сравнения/объединения

Итак, структурно программа будет состоять из 3 частей:

- Главная форма
- Форма вывода невнесенных изменений
- Файл с функциями, выполняющими сравнение и объединение

2.3 Описание программного продукта

Здесь будет лишь описание функций и процедур, а исходный код прикреплен в приложениях из-за его громоздкости.

Приложение занимается сравнением и объединением текстов модулей старой конфигурации и новой конфигурации поставщика. Текстовые файлы с кодом модулей достаточно легко получить через параметры командной строки. Авторская идея заключается в том, что нужно учитывать синтаксис языка 1С и комментарии программистов. Дело в том, что во многих компаниях у программистов принято все вносимые в код модулей правки заключать в комментарии с указанием, например, имени программиста и даты внесения изменения. Таким образом, можно находить эти комментарии в модуле старой конфигурации и переносить в модуль новой конфигурации все, что лежит между ними. Но тут возникает проблема – как найти нужное место для вставки.

Разработанный алгоритм находит с помощью комментариев изменение, запоминает предыдущую и следующую строки, затем эти строки находит в том блоке (процедуре, функции, области переменных) кода модуля новой конфигурации и вставляет правку в нужное место. Комментарии, между которыми лежат изменения, указывает программист в специальных полях интерфейса авторского приложения. По умолчанию они такие: «//правка_начало» и «//правка_конец».

2.3.1 Основные функции

Файл **MyForm.cpp** содержит главную функцию **main,** которая получает параметры командной строки, содержащие все необходимы для работы файлы с модулями конфигураций.

Заголовочный файл **FuncComparation.h** содержит 7 основных функций.

Функция **Compare_And_Search_Changes** с помощью параметров командной строки, полученных функцией main, считывает код старого измененного модуля. Затем с помощью комментариев находит в нем изменение, а также две строки, между которыми оно находится. Далее вызывается функция Print_Combined_Moudle. Чтобы избежать вставки изменения не в то место в случае, если искомая пара строк встретится раньше,

ведется счет количества совпадений предыдущей строки от начала файла до изменения. После того, как все изменения найдены и/или достигнут конец модуля, функция завершает свою работу, вызвав функцию Print_Combined_Moudle для вывода в итоговый файл оставшихся строк.

Процедура **Print_Combined_Moudle** находит нужное место в модуле новой конфигурации, ориентируясь на количество совпадений предыдущей строки, копирует все строки, предшествующие изменению в итоговый файл и вставляет туда изменение, найденное в предыдущей функции. В случае, если это не удается, функция вызывает процедуру Print_Not_Inserted.

Процедура **Print_Not_Inserted** выводит правку в файл с невнесенными изменениями с указанием номера строки, предыдущей и следующей строки в старом модуле и названия процедуры или функции, в котором оно содержится.

Функция **Delete_Spaces** удаляет пробелы и знаки табуляции из начала и конца строки для правильной работы алгоритма, а именно для корректного сравнения строк.

Функция **to_lower** приводит все символы в строке к нижнему регистру, опять же, для корректного сравнения.

Функция **Converter_to_1251** возвращает строку, преобразованную из кодировки UTF-8 в кодировку Windows-1251 [12].

Функция **Converter_to_UTF8** возвращает строку, преобразованную из кодировки Windows-1251 в кодировку UTF-8 [12].

2.3.2 Главная форма

Форма **Объединение модулей (MyForm.h)** представляет собой главное окно программы, которое появляется после ее вызова из окна сравнения в конфигураторе 1С. Форма содержит 3 больших текстовых поля, в которые выводятся новый модуль поставщика, старый измененный и итоговый файлы соответственно. Здесь же в малых текстовых полях можно указать путь для сохранения невнесенных изменений, а также комментарии, между которыми

лежит добавленный код. Если пользователь оставляет эти поля пустыми, обновление происходит с настройками по умолчанию.

Обработку заполнения текстовых блоков при загрузке производит функция **MyForm_Load**. Она также инициализирует переменные, необходимые для загрузки модулей, используя параметры командной строки, полученные в функции main [13].

В средний текстовый блок, содержащий старый модуль, можно в реальном времени вносить дополнения. Для сохранения этих дополнений нужно нажать кнопку «Сохранить изменения», которую обрабатывает функция **SaveChange_Click**.

Обработку нажатия кнопки «Обновить» производит функция **Combine_Click.** Она обрабатывает строку с именем модуля, выделяя его название, чтобы в дальнейшем использовать его для сохранения не вставленных изменений. Считывает данные из текстовых полей, получая путь для сохранения невнесенных изменений и комментарии. Вызывает функцию Compare_And_Search_Changes, в которой происходит сравнение и объединение текстов. Результат отображается в третье текстовое поле. Все внесенные изменения выделяются зеленым цветом, все не внесенные выделяются красным в блоке старого модуля. Также показывается общее количество изменений и число невнесенных доработок.

При нажатии на кнопку «Посмотреть» вызывается процедура **OpenNotIns_Click**, которая вызывает окно просмотра невнесенных изменений.

Интерфейс главной формы показан на рисунке 5.

Рисунок 5 - Интерфейс главного окна

2.3.3 Форма для вывода невнесенных изменений

Вторая форма **Невнесенные изменения (FormNotInsChange.h)** вызывается нажатием кнопки «Посмотреть». В ней отображаются только не внесенные в итоговый файл изменения с указанием номера строки, предыдущей и следующей строки в старом модуле и название процедуры или функции, в котором оно содержится.

В коде ничего особенного нет, только перегруженный конструктор класса формы, который получает в качестве параметра путь к файлу с невнесенными изменениями.

Интерфейс данной формы показан на рисунке 6.

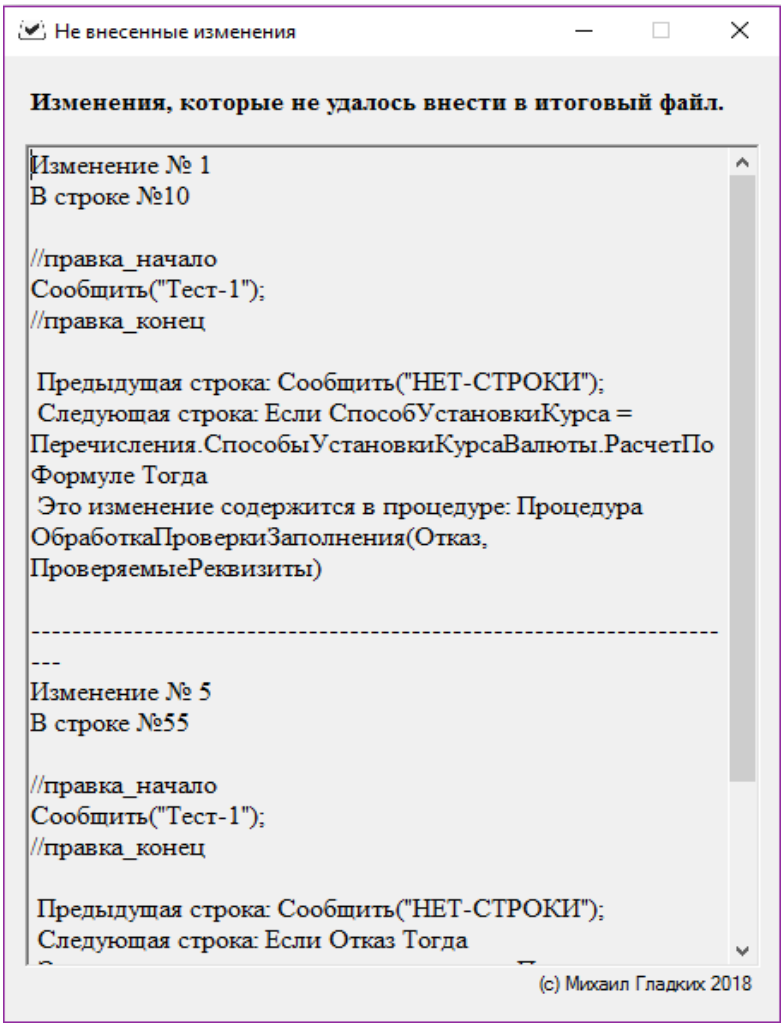

Рисунок 6 - Интерфейс окна невнесенных изменений

2.4 Руководство пользователя

Программа представляет из себя один исполняемый файл merge.exe размером всего 357 Кб. Чтобы использовать программу, в меню конфигуратора Сервис – Параметры необходимо выбрать на вкладке Сравнение/Объединение пункт Добавить. Затем в открывшемся окне нужно указать путь к файлу merge.exe и параметры командной строки. %baseCfg – модуль из старой конфигурации %secondCfg – модуль из новой конфигурации %baseCfgTitle – название модуля %merged – путь к системному каталогу, в который на время запишется результат объединения (рис. 7-8).

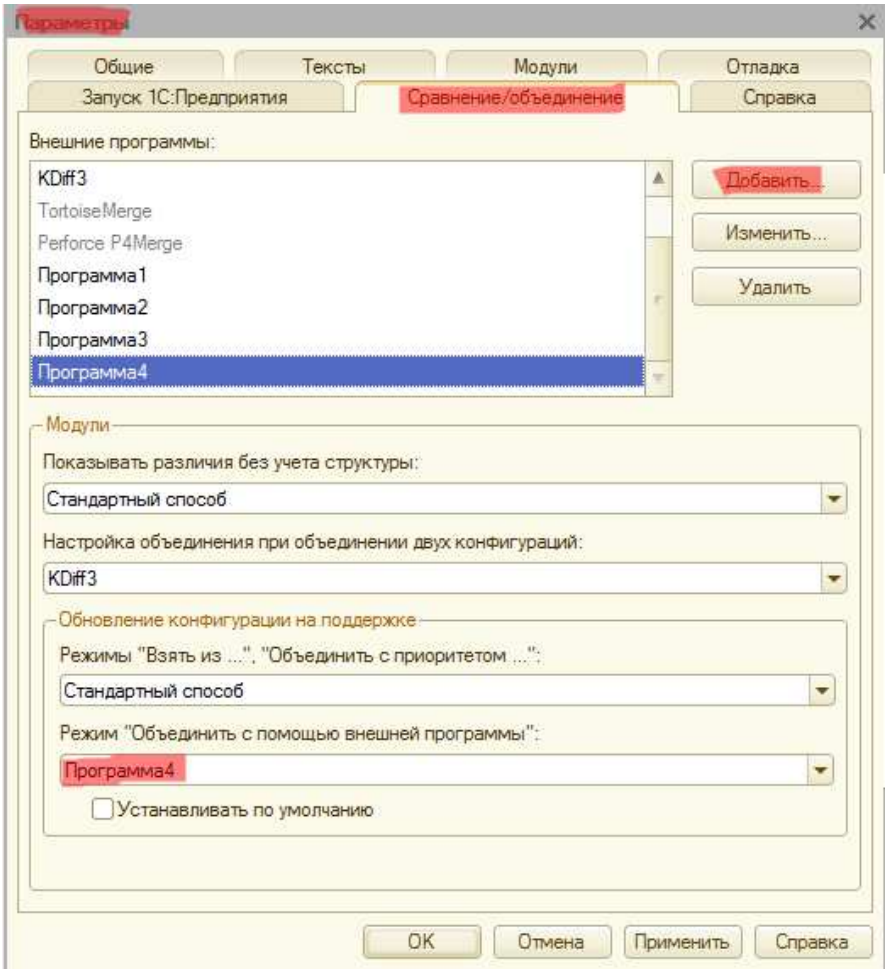

Рисунок 7 – Окно настройки программы

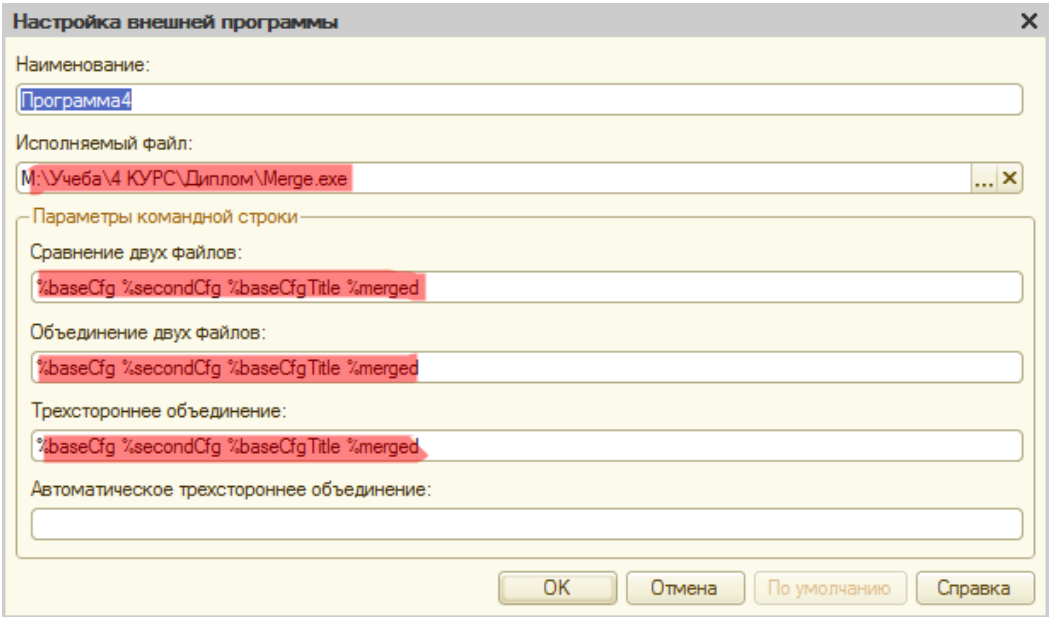

Рисунок 8 – Второе окно настройки программы

Далее при обновлении в дереве сравнения конфигураций поставить галочку напротив измененного модуля, и в колонке «Режим объединения и порядок подчинения» выбрать пункт «Объединить с помощью внешней программы» (рис. 9). Теперь при нажатии на шестеренку откроется главное окно программы (рис. 5). В нем можно указать путь для сохранения файла с не внесенными изменениями и комментарии, между которыми заключены дополнения. Если оставить эти поля пустыми, объединение будет происходить с настройками по умолчанию: файл с невнесенными изменениями сохранится в директорию «Невнесенные», которая автоматически создастся в каталоге с исполняемым файлом merge.exe, а комментарии будут иметь вид: «//правка_начало» и «//правка_конец». Также в этом окне можно редактировать старый модуль, для сохранения которого нужно нажать кнопку «Сохранить изменения».

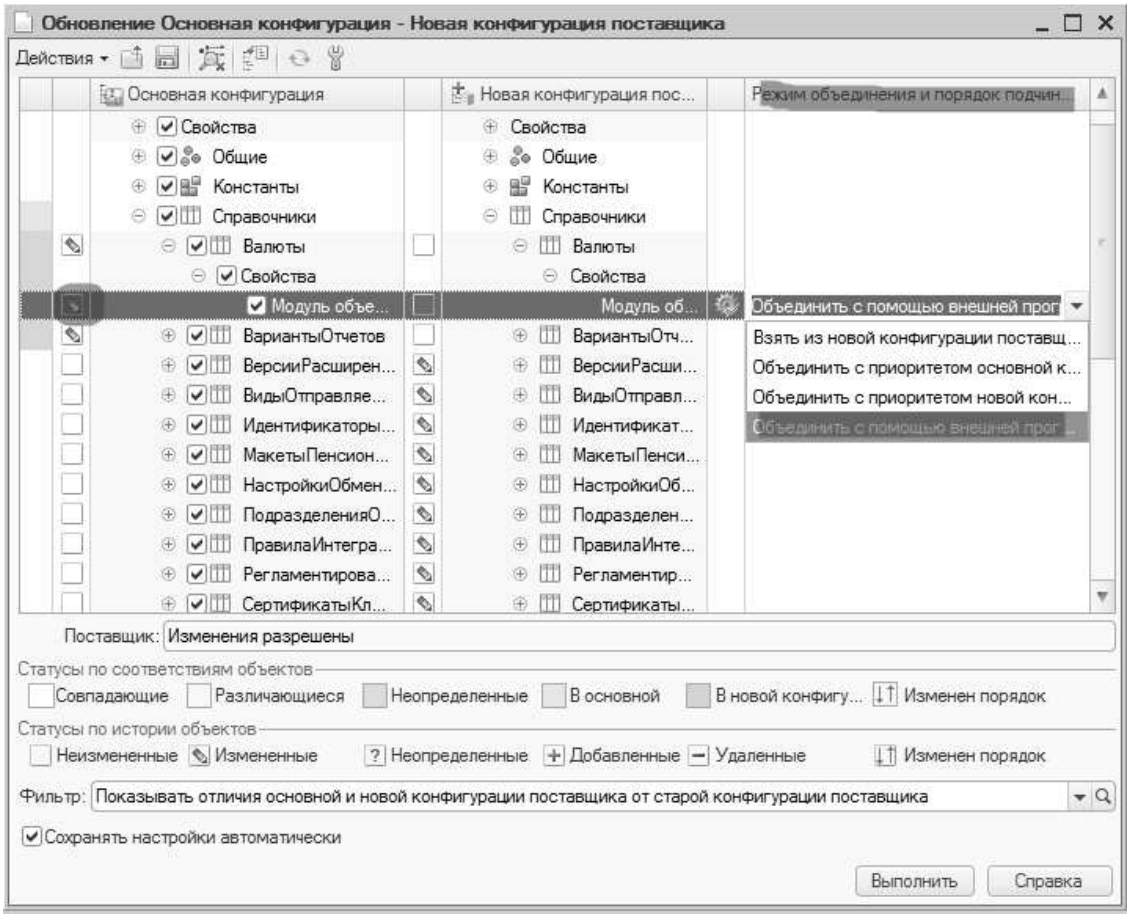

Рисунок 9 - Дерево сравнения конфигураций

После обновления с помощью кнопки «Посмотреть» можно увидеть изменения, которые не удалось внести.

Чтобы закончить сравнение и объединение нужно выйти из программы, нажав крестик в верхнем правом углу.

2.4 Тестирование

Тестирование разработанного приложения проводилось на версии платформы: «1С:Предприятие 8.3». В качестве тестовых были использованы следующие конфигурации: «Зарплата и управление персоналом»; «Бухгалтерия предприятия»; «Документооборот».

Результаты тестирования считаются удовлетворительными, программа отрабатывает заложенные в нее функции. Перенос изменений происходит корректно и в полном объеме.

Однако в процессе тестирования выявился один небольшой недостаток. В текущей версии приходится запускать программу отдельно для каждого объекта, что не совсем удобно. Гораздо удобнее будет, если программа сама получит все измененные модули объектов одновременно и автоматически их обработает. В этом случае программисту нужно будет нажать несколько кнопок и подождать окончания обновления. Поэтому дальнейшее развитие приложения будет вестись в направлении большей автоматизации.

ЗАКЛЮЧЕНИЕ

В результате работы были изучены встроенные в «1С:Предприятие» возможности по упрощению и автоматизации перевода конфигурации на новую версию. Были рассмотрены имеющиеся на данный момент внешние решения, автоматизирующие процесс обновления конфигураций. После анализа имеющихся решений было разработано собственное приложение для автоматического сравнения и объединения модулей конфигураций на языке программирования C++/CLI, а также реализовано взаимодействие приложения и 1С для получения необходимых для объединения данных.
Приложение было протестировано на реальных конфигурациях в системе «1С:Предприятие 8.3».

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1 Глоссарий разработчика [Электронный ресурс] : Информационно-технологическое сопровождение пользователей 1С:Предприятия. – Режим доступа: https://its.1c.ru/db/v8devgloss.

2 Сравнение и объединение конфигураций, механизм [Электронный ресурс] : 1С:Предприятие 8, система программ. – Режим доступа: http://v8.1c.ru/overview/Term_000000291.htm.

3 Автоматизированное обновление конфигураций 1С [Электронный ресурс] : - Режим доступа: https://solutions.1c.ru/catalog/auto-update/features.

4 Радченко, М. Г. 1С Предприятие 8.2 Практическое пособие разработчика. Примеры и типовые приемы / М. Г Радченко, Е. Ю Хрусталева.– М.:ООО «1С-Паблишинг», 2009. – 874 с.

5 Байдаков, В. 1С-Предприятие 8.3. Руководство разработчика / В. Байдаков, В. Дранищев, Е. Королькова, А. Краюшкин, И. Кузнецов, М. Лавров, А. Моничев, А. Плякин, М. Радченко.–М.: ООО «1С», 2012. – 346 с.

6 Расширения [Электронный ресурс] : 1С:Предприятие 8, система программ. - Режим доступа: http://v8.1c.ru/o7/201410ext/index.htm

7 Хрусталева, Е. Ю. 1С:Предприятие 8.1. Конфигурирование и администрирование - 3-е изд.перераб.и доп.-Спб.: «1С Паблишинг».2010. – 485с.

8 Хортон А. Visual C++ 2010: полный курс./ А. Хортон. – Пер.с англ. – М.: ООО «И.Д. Вильямс». 2011. - 1216 с.

9 Chowdhury K. Mastering Visual Studio 2017./ K. Chowdhury - Packt Publishing. 2017. - 466 с.

10 Templeman J. Microsoft Visual C++/CLI Step by Step./J. Templeman – 2013. 82 с.

11 Пахомов Б. И. C/C++ и MS Visual C++ 2010 для начинающих./ Б. И Пахомов – СПБ.: БХВ-Петербург. 2011. – 736 с.

12 Рихтер Дж. Windows для профессионалов: создание эффективных Win32-приложений с учетом специфики 64-разрядной версии Windows / Дж. Рихтер - Пер. с англ. — 4-е изд. — Спб.: Питер; М.: Издательство «Русская Редакция». 2008. - 720 с.

13 Hogenson G. Foundations of C++/CLI: The Visual C++ Language for .NET 3.5./ G. Hogenson - 2008. - 497 с.

ПРИЛОЖЕНИЕ А

Исходный код файла MyForm.cpp

```
int main(cli::array<String^>^ args)
{
	setlocale(LC_ALL, "");
	Application::EnableVisualStyles();
	Application::SetCompatibleTextRenderingDefault(false);
	Project1::MyForm form, param;
	if (args->Length > 0)
	{
		param.WriteParam(args);
		Application::Run(%form);
			return 0;
	}
	else
	{
		MessageBox::Show("Не удалось загрузить модули. Проверьте правильность указанных параметров командной строки.","Ошибка",
MessageBoxButtons::OK,
MessageBoxIcon::Error);
			return 0;
	}

}
```

ПРИЛОЖЕНИЕ Б

Исходный код файла FuncComparation.h

```
const int max_len = 500;
const char *OldMoudle, *NewMoudle, *stream_comb, *stream_not_ins;
string ArrParam[4], not_ins, NameMoudle, AdrSave;
int count_change, k, count_not_ins, MasNumNotIns[500] = { 0 }, j = 0;
FILE *New_Moudle_Provider, *Old_Moudle_Modified;
long ptr1, ptr2, ptr3;
char Str_buff[max_len];
string begin_change = "//правка_начало", end_change = "//правка_конец", procedure = "процедура",
function = "функция", end_procedure = "конецпроцедуры", end_function = "конецфункции";
string name_proc, name_func, next_str, previous_str, Changes, Converted_Str, str_lower;
bool found_proc, found_func;
void Print_Not_Inserted(string stream_not_ins);
void Print_Combined_Moudle(const char *stream_comb, string stream_not_ins, bool flag);
string Delete_Spaces(string str);
string to_lower(string str);
string Converter_to_1251(string const & str_utf8);
string Converter_UT(string const & str_utf8);
void Compare_And_Search_Changes(const char *stream_comb, string stream_not_ins)
{
	count_change = 0; k = 0; ptr1 = 0; ptr2 = 0;  ptr3 = 0; found_proc = 0; found_func = 0;
	count_not_ins = 0; previous_str = ""; next_str = ""; str_lower = "";
	string Str_Of_Old_Moudle, buff;
	bool  found_end_change = 0, found_change = 0, several_changes = 0, in_change = 0;
	fseek(Old_Moudle_Modified, 0, SEEK_SET);
	while (1)
	{
		fgets(Str_buff, max_len, Old_Moudle_Modified);
		if (feof(Old_Moudle_Modified)) break;
		Converted_Str = Converter_to_1251(Str_buff);
		Str_Of_Old_Moudle = Delete_Spaces(Converted_Str);
		if (!Str_Of_Old_Moudle.empty())
		{
			next_str = Str_Of_Old_Moudle;
			str_lower = to_lower(Str_Of_Old_Moudle);
			if (several_changes && str_lower.substr(0, 15) != begin_change)
			{
```

```
                found_change = 0;
                found_end_change = 1;
                in_change = 0;
                several_changes = 0;
        }
        if (str_lower.substr(0, 9) == procedure && !in_change)
        {
                name_proc = Converted_Str;
                found_proc = 1;
        }
        else if (str_lower.substr(0, 14) == end_procedure)
                found_proc = 0;
        if (str_lower.substr(0, 7) == function && !in_change)
        {
                name_func = Converted_Str;
                found_func = 1;
        }
        else if (str_lower.substr(0, 12) == end_function)
                found_func = 0;
        if (!found_change)
                buff = to_lower(next_str);
        if (found_end_change)
        {
                found_end_change = 0;
                string str_buff;
                k = 0;
                ptr2 = ftell(Old_Moudle_Modified);
                fseek(Old_Moudle_Modified, 0, SEEK_SET);
                while (1)
                {
                        fgets(Str_buff, max_len, Old_Moudle_Modified);

if(feof(Old_Moudle_Modified)||ptr2==ftell(Old_Moudle_Modified))
                                break;
                        Converted_Str = Converter_to_1251(Str_buff);
                        Converted_Str = Delete_Spaces(Converted_Str);
                        if (previous_str == Converted_Str)
                                k++;
                }
                fseek(Old_Moudle_Modified, ptr2, SEEK_SET);
                Print_Combined_Moudle(stream_comb, stream_not_ins, 1);
```

```
                        }
                        if (buff.substr(0, 15) == begin_change)
                        {
                                found_change = 1;
                                in_change = 1;
                                if (str_lower.substr(0, 14) != end_change)
                                {
                                        Changes += Str_buff;
                                        several_changes = 0;
                                }
                                else
                                {
                                        count_change++;
                                        Changes += Str_buff;
                                        several_changes = 1;
                                }
                        }
                        if (!found_change)
                                previous_str = Str_Of_Old_Moudle;
                }
        }
        Print_Combined_Moudle(stream_comb, stream_not_ins, 0);
}
void Print_Not_Inserted(string stream_not_ins)
{
        bool contained = 0;
        ofstream Out_Not_Ins(stream_not_ins, ios_base::app);
        count_not_ins++;
        Out_Not_Ins << "Изменение № " << count_change << endl;
        Out_Not_Ins << Converter_to_1251(Changes);
        Changes = "";
        MasNumNotIns[j] = count_change; j++;
        if (!previous_str.empty())
                Out_Not_Ins << "\n Предыдущая строка: " << previous_str;
        str_lower = to_lower(next_str);
        if (str_lower.substr(0, 14) != end_change)
                Out_Not_Ins << "\n Следующая строка: " << next_str;
        str_lower = to_lower(previous_str);
        if (found_proc && str_lower.substr(0, 14) != end_procedure)
        {
                Out_Not_Ins << "\n Это изменение содержится в процедуре: " << name_proc;
```

```
                contained = 1;
        }
        else if (!contained)
                Out_Not_Ins << "\n Это изменение содержится вне процедуры. \n";
        if (found_func && str_lower.substr(0, 12) != end_function)
        {
                Out_Not_Ins << "\n Это изменение содержится в функции: " << name_func;
                contained = 1;
        }
        else if (!contained)
                Out_Not_Ins << " Это изменение содержится вне функции. \n";
        Out_Not_Ins << "\n--------------------------------------------------------------------------" << endl;
        Out_Not_Ins.close();
}
void Print_Combined_Moudle(const char *stream_comb, string stream_not_ins, bool flag)
{
        ofstream Out_Combined(stream_comb, ios_base::app);
        bool out_change = 0, found_prev_str = 0;
        if (flag)
        {
                string Str_Of_New_Moudle, Buffer;
                fseek(New_Moudle_Provider, ptr3, SEEK_SET);
                while (!out_change)
                {
                        fgets(Str_buff, max_len, New_Moudle_Provider);
                        Converted_Str = Converter_to_1251(Str_buff);
                        Str_Of_New_Moudle = Delete_Spaces(Converted_Str);
                        if (previous_str.empty())
                        {
                                Out_Combined << Delete_Spaces(Changes) << endl;
                                Changes = "";
                                found_prev_str = 0;
                                out_change = 1;
                        }
                        if (previous_str == Str_Of_New_Moudle && !found_prev_str)
                        {
                                found_prev_str = 1;
                                        ptr1 = ftell(New_Moudle_Provider);
                                fseek(New_Moudle_Provider, 0, SEEK_SET);
                                while (1)
                                {
```

```
                                        fgets(Str_buff, max_len, New_Moudle_Provider);
                                        Converted_Str = Converter_to_1251(Str_buff);
                                        Str_Of_New_Moudle = Delete_Spaces(Converted_Str);
                                        if(ptr1 <= ftell(New_Moudle_Provider))
                                        Buffer += Str_buff;
                                        if (previous_str== Str_Of_New_Moudle)
                                                k--;
                                        if (k == 0) {
                                                ptr3 = ftell(New_Moudle_Provider);
                                                break;
                                        }
                                        if (feof(New_Moudle_Provider)) break;
                                }
                                fseek(New_Moudle_Provider, ptr3, SEEK_SET);
                        }
                        if (next_str == Str_Of_New_Moudle && found_prev_str)
                        {
                                found_prev_str = 0;
                                out_change = 1;
                        }
                        if (!out_change    && previous_str != Str_Of_New_Moudle)
                                Buffer += Str_buff;
                        if (!out_change && found_prev_str == 0 && feof(New_Moudle_Provider) &&
!Changes.empty())
                                Print_Not_Inserted(stream_not_ins);
                        if (feof(New_Moudle_Provider))
                                break;
                }
                if (!feof(New_Moudle_Provider))
                {
                        if (Buffer[0] == '?')
                                Buffer.erase(0, 1);
                        Out_Combined << Buffer;
                        Out_Combined << Changes;
                }
                Buffer = "";
                Changes = "";
        }
        else
        {
                string s;
```

```
                        fseek(New_Moudle_Provider, ptr1, SEEK_SET);
                        while (!feof(New_Moudle_Provider))
                        {
                                fgets(Str_buff, max_len, New_Moudle_Provider);
                                if (feof(New_Moudle_Provider) && s == Str_buff)
                                        break;
                                s = Str_buff;
                                Out_Combined << Str_buff;
                        }
                        if (!Changes.empty())
                                Out_Combined << endl << Changes;
                }
                Out_Combined.close();
        }
        string Delete_Spaces(string str)
        {
                while ((str[0] == ' ' || str[0] == '?' || str[0] == '\t' || str[0] == '\n') && str.length() > 0)
                        str.erase(0, 1);
                if (str.length() != 0)
                        while ((str[str.length() - 1] == ' ' || str[str.length() - 1] == '\t' || str[str.length() - 1] == '\n') &&
str.length() > 0)
                                str.erase(str.length() - 1, 1);
                return str;
        }
        string Converter_to_1251(string const & str_utf8)
        {
                if (!str_utf8.empty())
                {
                        int wchar_len = MultiByteToWideChar(CP_UTF8, 0, str_utf8.c_str(), str_utf8.length(),
NULL, 0);
                        if (wchar_len > 0 && wchar_len != 0xFFFD)
                        {
                                vector<wchar_t> wbuf(wchar_len);
                                MultiByteToWideChar(CP_UTF8, 0, str_utf8.c_str(), str_utf8.length(), &wbuf[0],
wchar_len);
                                vector<char> buf(wchar_len);
                                WideCharToMultiByte(1251, 0, &wbuf[0], wchar_len, &buf[0], wchar_len, 0, 0);
                                return string(&buf[0], wchar_len);
                        }
                }
                return string();
```

```
	}
	string Converter_to_UTF8(const wchar_t* buffer, int len)
	{
		int nChars = ::WideCharToMultiByte(CP_UTF8,0,buffer,len,NULL,0,NULL,NULL);
		if (nChars == 0) return "";
		string newbuffer;
		newbuffer.resize(nChars);
		::WideCharToMultiByte(CP_UTF8,0,buffer,len,const_cast< char*
>(newbuffer.c_str()),nChars,NULL,NULL);
		return newbuffer;
	}
	string Converter_to_UTF8(const std::wstring& str)
	{
		return Converter_to_UTF8(str.c_str(), (int)str.size());
	}
	string to_lower(string str)
	{
		transform(str.begin(), str.end(), str.begin(), ::tolower);
		return str;
	}
```

ПРИЛОЖЕНИЕ В

Исходный код файла MyForm.h (Главная форма)

```
private: System::Void MyForm_Load(System::Object^ sender, System::EventArgs^ e) {

        bool paint = 0;
            OldMoudle = ArrParam[0].c_str();
            NewMoudle = ArrParam[1].c_str();
            stream_not_ins = ArrParam[2].c_str();          stream_comb = ArrParam[3].c_str();
                Old_Moudle_Modified = fopen(OldMoudle, "a+");
                New_Moudle_rtb->LoadFile(msclr::interop::marshal_as<String^>(NewMoudle),
RichTextBoxStreamType::PlainText);
            fseek(Old_Moudle_Modified, 0, SEEK_END);
                fputs("\n", Old_Moudle_Modified);
            fseek(Old_Moudle_Modified, 0, SEEK_SET);
            while (1)
            {
                fgets(Str_buff, max_len, Old_Moudle_Modified);
                    if (feof(Old_Moudle_Modified))
                        break;
                Converted_Str = Converter_to_1251(Str_buff);
                str_lower = to_lower(Delete_Spaces(Converted_Str));
                if (Converted_Str[0] == '?')
                    Converted_Str.erase(0, 1);
                if (str_lower.substr(0, 15) == begin_change)
                    paint = 1;
                if (paint)
                    Old_Moudle_rtb->SelectionColor = dlgFont->Color.Green;
                if (str_lower.substr(0, 14) == end_change)
                    paint = 0;
                strcpy(Str_buff, Converted_Str.c_str());
                Old_Moudle_rtb->AppendText(gcnew String(Str_buff));
            }
            fclose(Old_Moudle_Modified);
            textBox2->Text = "//правка_начало";
            textBox3->Text = "//правка_конец";
        }
        private: System::Void Combine_Click(System::Object^ sender, System::EventArgs^ e) {
```

```
char* pstr;
FILE *CombMoudle, *SaveNotIns;
	bool paint = 0, inserted = 1;
	int i = 0, num_not_ins = 0;
	TCHAR path[MAX_PATH];
	GetModuleFileName(NULL, path, MAX_PATH);
	USES_CONVERSION;
	pstr = W2A(path);
if(textBox2->Text != "" && textBox2->Text != "")
{
	begin_change = msclr::interop::marshal_as<std::string>(textBox2->Text);
	end_change = msclr::interop::marshal_as<std::string>(textBox3->Text);
	to_lower(begin_change); to_lower(end_change);
}
Old_Moudle_Modified = fopen(OldMoudle, "r");
New_Moudle_Provider = fopen(NewMoudle, "r");
not_ins = pstr;
if (not_ins.length() != 0)
	while ((not_ins[not_ins.length() - 1] != '\\') && not_ins.length() > 0)
		not_ins.erase(not_ins.length() - 1, 1);
not_ins += "Невнесенные";
CreateDirectory(A2W(not_ins.c_str()), NULL);
Old_Moudle_rtb->Clear();
Comb_Moudle_rtb->Clear();
NameMoudle.clear();
for (int i = 0; stream_not_ins[i] != '.'; i++)
{
	NameMoudle += stream_not_ins[i];
		if (stream_not_ins[i] == '\0')
			break;
}
not_ins += "\\"+NameMoudle + ".txt";
ofstream stream1(stream_comb, ios_base::trunc);
ofstream stream2(not_ins.c_str(), ios_base::trunc);
CombMoudle = fopen(stream_comb, "r");
if (textBox1->Text != "")
	AdrSave = msclr::interop::marshal_as<std::string>(textBox1->Text);
Compare_And_Search_Changes(stream_comb, not_ins);
	CountChange->Text = Convert::ToString(count_change);
	CountNotIns->Text = Convert::ToString(count_not_ins);
fseek(Old_Moudle_Modified, 0, SEEK_SET);
```

```
while (1)
{
	fgets(Str_buff, max_len, Old_Moudle_Modified);
		if (feof(Old_Moudle_Modified))
			break;
	Converted_Str = Converter_to_1251(Str_buff);
	str_lower = to_lower(Delete_Spaces(Converted_Str));
	if (Converted_Str[0] == '?')
		Converted_Str.erase(0, 1);
	if (str_lower.substr(0, 15) == begin_change)
	{
		paint = 1;
		num_not_ins++;
	}
	if (paint && inserted)
		Old_Moudle_rtb->SelectionColor = dlgFont->Color.Green;
	if (MasNumNotIns[i] == num_not_ins && MasNumNotIns[i] != 0)
	{
		i++;
		inserted = 0;
	}
	if (!inserted)
		Old_Moudle_rtb->SelectionColor = dlgFont->Color.Red;
	if (str_lower.substr(0, 14) == end_change)
	{
		paint = 0;
		inserted = 1;
	}
	strcpy(Str_buff, Converted_Str.c_str());
		Old_Moudle_rtb->AppendText(gcnew String(Str_buff));
}
fseek(CombMoudle, 0, SEEK_SET);
while (1)
{
	fgets(Str_buff, max_len, CombMoudle);
		if (feof(CombMoudle))
			break;
	Converted_Str = Converter_to_1251(Str_buff);
	str_lower = to_lower(Delete_Spaces(Converted_Str));
	if (Converted_Str[0] == '?')
		Converted_Str.erase(0, 1);
```

```
                        if (str_lower.substr(0, 15) == begin_change)
                                paint = 1;
                        if (paint)
                                Comb_Moudle_rtb->SelectionColor = dlgFont->Color.Green;
                        if (str_lower.substr(0, 14) == end_change)
                                paint = 0;
                        strcpy(Str_buff, Converted_Str.c_str());
                                Comb_Moudle_rtb->AppendText(gcnew String(Str_buff));
                }
                fclose(New_Moudle_Provider);
        fclose(Old_Moudle_Modified);
fclose(CombMoudle);
                if (AdrSave != "")
                        {
                ofstream stream5(AdrSave + "\\" + NameMoudle + ".txt");
                        SaveNotIns = fopen(not_ins.c_str(), "r");
                while (1)
                        {
                                fgets(Str_buff, max_len, SaveNotIns);
                                        if (feof(SaveNotIns))
                                                break;
                                stream5 << Str_buff;
                        }
                fclose(SaveNotIns);
        stream5.close();
                }
        }
        private: System::Void SaveChange_Click(System::Object^ sender, System::EventArgs^ e) {
                 string sfr; wstring wstr;
                 ofstream save(OldMoudle, std::ios::out | std::ios::binary);
                        wstr = msclr::interop::marshal_as<std::wstring>(Old_Moudle_rtb->Text);
                        sfr = Converter_to_UTF8(wstr);
                                save << sfr;
        }
private: System::Void OpenNotIns_Click(System::Object^ sender, System::EventArgs^ e) {
        if(not_ins!="")
                {
                        FormNotInsChange^ f2 = gcnew FormNotInsChange(not_ins);
                                f2->Show(this);
                }
}
```

```
        private: System::Void openFileDialog1_FileOk(System::Object^  sender,
System::ComponentModel::CancelEventArgs^  e) {
        }
        private: System::Void saveFileDialog1_FileOk(System::Object^  sender,
System::ComponentModel::CancelEventArgs^  e) {
        }
        private: System::Void MyForm_Closing(System::Object^  sender,
System::Windows::Forms::FormClosingEventArgs^  e) {
                if (AdrSave != "")
                {
                        remove(not_ins.c_str());
                }
        }
        };
        }
```

ПРИЛОЖЕНИЕ Г

Исходный код файла FormNotInsChange.h
(Форма невнесенных изменений)

```
#pragma once
#include <msclr/marshal_cppstd.h>
namespace Project1 {

	using namespace System;
	using namespace System::ComponentModel;
	using namespace System::Collections;
	using namespace System::Windows::Forms;
	using namespace System::Data;
	using namespace System::Drawing;
	public ref class FormNotInsChange : public System::Windows::Forms::Form
	{
	public:

		FormNotInsChange(std::string text)
		{
			InitializeComponent();
			richTextBox1->LoadFile(msclr::interop::marshal_as<String^>(text),
RichTextBoxStreamType::PlainText);
		}


	protected:
		~FormNotInsChange()
		{
			if (components)
			{
				delete components;
			}
		}
	private: System::Windows::Forms::RichTextBox^  richTextBox1;
	private: System::Windows::Forms::Label^  label1;
	private: System::Windows::Forms::Label^  label2;
	protected:
```

```
private:
        /// <summary>
        /// Обязательная переменная конструктора.
        /// </summary>
        System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code
        /// <summary>
        /// Требуемый метод для поддержки конструктора — не изменяйте
        /// содержимое этого метода с помощью редактора кода.
        /// </summary>
        void InitializeComponent(void)
        {
            System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(FormNotInsChange::typeid));
            this->richTextBox1 = (gcnew System::Windows::Forms::RichTextBox());
            this->label1 = (gcnew System::Windows::Forms::Label());
            this->label2 = (gcnew System::Windows::Forms::Label());
            this->SuspendLayout();
            //
            // richTextBox1
            //
            this->richTextBox1->Font = (gcnew System::Drawing::Font(L"Times New Roman", 12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(204)));
            this->richTextBox1->Location = System::Drawing::Point(12, 54);
            this->richTextBox1->Name = L"richTextBox1";
            this->richTextBox1->ReadOnly = true;
            this->richTextBox1->Size = System::Drawing::Size(429, 505);
            this->richTextBox1->TabIndex = 0;
            this->richTextBox1->Text = L"";
            //
            // label1
            //
            this->label1->AutoSize = true;
            this->label1->Location = System::Drawing::Point(309, 562);
            this->label1->Name = L"label1";
            this->label1->Size = System::Drawing::Size(131, 13);
            this->label1->TabIndex = 1;
            this->label1->Text = L"(c) Михаил Гладких 2018";
```

```
                                //
                                // label2
                                //
                                this->label2->AutoSize = true;
                                this->label2->Font = (gcnew System::Drawing::Font(L"Times New Roman",
11.25F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
                                        static_cast<System::Byte>(204)));
                                this->label2->Location = System::Drawing::Point(12, 19);
                                this->label2->Name = L"label2";
                                this->label2->Size = System::Drawing::Size(413, 17);
                                this->label2->TabIndex = 2;
                                this->label2->Text = L"Изменения, которые не удалось внести в итоговый
файл.";
                                //
                                // FormNotInsChange
                                //
                                this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);
                                this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;
                                this->BackColor = System::Drawing::SystemColors::Control;
                                this->ClientSize = System::Drawing::Size(452, 592);
                                this->Controls->Add(this->label2);
                                this->Controls->Add(this->label1);
                                this->Controls->Add(this->richTextBox1);
                                this->FormBorderStyle                                                =
System::Windows::Forms::FormBorderStyle::FixedDialog;
                                this->Icon          =          (cli::safe_cast<System::Drawing::Icon^>(resources-
>GetObject(L"$this.Icon")));
                                this->MaximizeBox = false;
                                this->Name = L"FormNotInsChange";
                                this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;
                                this->Text = L"Не внесенные изменения";
                                this->ResumeLayout(false);
                                this->PerformLayout();

                        }
        #pragma endregion

                };
        }
```

Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт математики и фундаментальной информатики
Базовая кафедра вычислительных и информационных технологий

УТВЕРЖДАЮ
Заведующий кафедрой
_____ / В.В. Шайдуров
«_2_» _июня_ 2018 г.

БАКАЛАВРСКАЯ РАБОТА

Направление 02.03.01 Математика и компьютерные науки

АВТОМАТИЗИРОВАННОЕ ОБНОВЛЕНИЕ ИЗМЕНЕННЫХ КОНФИГУРАЦИЙ 1С

Научный руководитель
кандидат физико-математических наук,
доцент _____ / Д.А. Цыганок
08.06.2018

Выпускник _____ / М.В. Гладких
08.06.2018

Красноярск 2018