ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Кубанский государственный аграрный университет Имени

И.Т. Трубилина»

Факультет прикладной информатики

Кафедра системного анализа и обработки информации

**КУРСОВАЯ РАБОТА**

по дисциплине: Системный анализ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

на тему: Разработка подсистемы для оценки организационных структур организации, основанной на информационном подходе

выполнила студентка группы БИ1401

Семёнова Екатерина Константиновна

Допущен к защите

Руководитель проекта Барановская Т.П.\_\_\_\_\_\_\_\_\_\_\_\_\_

Нормоконтролер\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, расшифровка подписи)

Защищен\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

Члены комиссии\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата, расшифровка подписи)

Краснодар

2017

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Кубанский государственный аграрный университет Имени

И.Т. Трубилина»

Факультет прикладной информатики

Кафедра системного анализа и обработки информации

УТВЕРЖДАЮ:

Зав.кафедрой\_\_

Барановская Т.П. \_\_

**ЗАДАНИЕ**

**на курсовую работу**

Студенту:\_\_\_\_\_БИ1401\_\_\_\_\_\_\_\_\_\_группы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3\_курса

Факультета Прикладной информатики\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Специальности 38.03.05 Бизнес-информатика\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Семёновой Екатерине Константиновне\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Ф.И.О, шифр)

Тема курсовой Разработка подсистемы для оценки организационных структур организации, основанной на информационном подходе

Содержание задания\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, расшифровка подписи)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Объем работы:

а) пояснительная записка к проекту\_\_\_\_\_\_\_\_\_\_66\_\_листов формата А4

Срок выполнения проекта с: «»\_\_\_\_\_\_\_\_\_\_по «» \_\_\_\_\_\_\_\_\_ 2017г.

Срок защиты: « » \_\_\_\_\_\_\_\_\_\_\_\_ 2017г.

Дата выдачи задания: « » \_\_\_\_\_\_\_\_\_\_\_\_ 2017г.

Дата сдачи проекта на кафедру: « » \_\_\_\_\_\_\_\_\_\_\_\_ 2017г.

Руководитель проекта\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, Ф.И.О., звание, степень)

Задание принял студент\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата)

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc486223873)

[1 ХАРАКТЕРИСТИКА ОБЪЕКТА ИССЛЕДОВАНИЯ И ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc486223874)

[1.1 Понятие организационных структур 5](#_Toc486223875)

[1.2 Сущность и назначение оценки организационных структур 8](#_Toc486223876)

[1.3 Информационный подход к оценке организационных структур 11](#_Toc486223877)

[2 РАЗРАБОТКА ТРЕБОВАНИЙ К ПО ПРОЦЕДУРЫ ОЦЕНКИ ОРГСТРУКТУР 19](#_Toc486223878)

[2.1 Разработка функциональной модели 19](#_Toc486223879)

[2.2 Выбор средств реализации 20](#_Toc486223880)

[3 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ПО ОЦЕНКЕ ОРГСТРУКТУР 22](#_Toc486223881)

[3.1 Описание процесса разработки 22](#_Toc486223882)

[3.2 Инструкция пользователя 31](#_Toc486223883)

[ЗАКЛЮЧЕНИЕ 36](#_Toc486223884)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 37](#_Toc486223885)

[ПРИЛОЖЕНИЕ А 39](#_Toc486223886)

[ПРИЛОЖЕНИЕ Б 42](#_Toc486223887)

# ВВЕДЕНИЕ

С каждым днем появляются все новые предприятия, организации, становится жестче законодательство, на фоне чего конкурентная борьба приобретает все более жесткие формы. Затрудняется разработка и реализация стратегических планов организаций, которые, в свою очередь, требуют повышения ее гибкости, способности оперативно реагировать на постоянные меняющиеся условия деятельности, что напрямую зависит от функционирующей структуры управления предприятием. Поэтому одним из основополагающих принципов рыночной экономики является отбор наиболее эффективных организационных систем.

Актуальность курсового проекта обусловлена тем, что оптимизация организационных структур управления является составной частью комплекса мер по повышению эффективности работы любого предприятия. Руководители должны выбрать ту структуру, которая лучше всего отвечает поставленным целям и задачам предприятия, своевременно и адекватно реагирует на действия факторов внутренней и внешней среды, целенаправленно распределяет и координирует усилия сотрудников и таким образом повышает собственную конкурентоспособность.

Целью данного проекта является разработка подсистемы для оценки организационных структур организации, основанной на информационном подходе.

Задачи курсового проекта:

* исследование предметной области;
* анализ объекта автоматизации;
* обоснованность выбора средств разработки;
* описание структуры базы данных;
* реализация бизнес-приложения;
* тестирование бизнес-приложения;
* составление руководство пользователя.

# 1 ХАРАКТЕРИСТИКА ОБЪЕКТА ИССЛЕДОВАНИЯ И ПРЕДМЕТНОЙ ОБЛАСТИ

## Понятие организационных структур

**Организационная структура управления** — это упорядоченная совокупность взаимосвязанных элементов, находящихся между собой в устойчивых отношениях, обеспечивающих их развитие и функционирование как единого целого.

Организационная структура управления направлена на установление четких взаимосвязей между отдельными подразделениями организации, распределение между ними прав и ответственности.

Связи между элементами структуры управления бывают:

1) вертикальными, когда происходит взаимодействие между руководителем и подчиненным (например, связь между директором фирмы и управляющим структурным подразделением);

2) горизонтальными, когда происходит взаимодействие равноправных элементов (например, связи между управляющими структурными подразделениями одного уровня)

Виды отношений внутри организации аналогичны типу построения структуры ее управления и делятся на:

1) линейные отношения — это отношения между руководителем и его подчиненными;

2) функциональные отношения — это отношения специалиста, который уполномочен выполнять ту или иную функцию в рамках всей организации, с другими членами организации;

3) отношения управленческого аппарата, данный тип отношений имеет место в случае представления чьих-либо прав и полномочий. Должностные обязанности при этом состоят в предоставлении рекомендации, советов.

Структура управления оказывает огромное влияние на все стороны управления, так как связана с ключевыми понятиями менеджмента — целями, функциями, процессом, механизмом, функционирования, полномочиями людей. Поэтому менеджеры всех уровней уделяют огромное внимание принципам и методам формирования структур, выбору типа или комбинаций типов структур, изучению тенденций их построения, оценке их соответствия решаемым целям и задачам.

Элементами структуры управления являются: работник управления — человек, выполняющий определенную функцию управления; орган управления — группа работников, связанных определенными отношениями, состоящая из первичных групп.

Первичная группа — коллектив работников управления, у которого есть общий руководитель, но нет подчиненных.

Структура управления должна отражать цели и задачи фирмы, быть подчиненной производству и меняться вместе с ним. Она должна отражать функциональное разделение труда и объем полномочий работников управления; последние определяются политикой, процедурами, правилами и должностными инструкциями и расширяются, как правило, в направлении более высоких уровней управления. Полномочия руководителей ограничиваются факторами внешней среды, уровнем культуры и ценностными ориентациями, принятыми традициями и нормами. Структура управления должна удовлетворять множеству требований, которые отражают ее значение для менеджмента. Эти принципы учитываются в принципах проектирования организационной структуры управления. Принципы проектирования организационной структуры:

1. структура должна отражать цели и задачи организации, быть подчиненной производству и его потребностям;

2. структура должна предусматривать оптимальное разделение труда между органами управления и отдельными работниками, обеспечивающее творческий характер работы и нормальную нагрузку, а также надлежащую специализацию;

3. формирование структуры должно быть неразрывно с определением полномочий и ответственности каждого работника и органа управления, с установлением системы вертикальных и горизонтальных связей между ними;

4. структура должна поддерживать соответствие между функциями, обязанностями, полномочиями и ответственностью, так как нарушение его приводит к дисбалансу системы управления в целом;

5. структура управления должна быть адекватной социально-культурной среде организации, оказывать существенное влияние на решения относительно уровня централизации и детализации, распределения полномочий и ответственности, степени самостоятельности и масштабов контроля руководителей и менеджеров.

Важнейшими требованиями, которым должны удовлетворять организационные структуры управления являются следующие.

1. Оптимальность. Структура управления признается оптимальной, если между звеньями и ступенями управления на всех уровнях устанавливаются рациональные связи при наименьшем числе ступеней управления.

2. Оперативность. Суть данного требования состоит в том, чтобы за время от принятия решения до его исполнения в управляемой системе не успели произойти необратимые отрицательные изменения, делающие ненужной реализацию принятых решений.

3. Надежность. Структура аппарата управления должна гарантировать достоверность передачи информации, не допускать искажений управляющих команд и других передаваемых данных, обеспечивать бесперебойность связи в системе управления.

4. Экономичность. Задача состоит в том, чтобы нужный эффект от управления достигался при минимальных затратах на управленческий аппарат. Критерием этого может служить соотношение между затратами ресурсов и полезным результатом.

5. Гибкость. Способность изменяться в соответствии с изменениями внешней среды.

6. Устойчивость структуры управления. Неизменность ее основных свойств при различных внешних воздействиях, целостность функционирования системы управления и ее элементов.

## Сущность и назначение оценки организационных структур

Оценка эффективности является важным элементом разработки проектных и плановых решений, позволяющих определить уровень прогрессивности действующей структуры, и проводится с целью выбора наиболее рационального варианта структуры или способа её совершенствования. Эффективность организационной структуры должна оцениваться на стадии проектирования, при анализе структур управления действующих организаций для планирования и осуществления мероприятий по совершенствованию управления.

Принципиальное значение для оценки эффективности системы управления имеет выбор базы для сравнения или определения уровня эффективности, который принимается за нормативный. Один из подходов дифференцирования сводится к сравнению с показателями, характеризующими эффективность организационной структуры эталонного варианта систем управления. Эталонный вариант может быть разработан и спроектирован с использованием всех имеющихся методов и средств проектирования систем управления. Характеристики такого варианта принимаются в качестве нормативных. Может применяться также сравнение с показателями эффективности и характеристиками системы управления, выбранной в качестве эталона, определяющего допустимый или достаточный уровень эффективности организационной структуры.

Критерием эффективности при сравнении различных вариантов организационной структуры служит возможность наиболее полного и устойчивого достижения конечных целей системы управления при относительно меньших затратах на её функционирование.

Показатели, используемые при оценках эффективности аппарата управления и его организационной структуры, могут быть разбиты на следующие три взаимосвязанные группы.

1. Группа показателей, характеризующих эффективность системы управления, выражающихся через конечные результаты деятельности организации, и затраты на управления.

2. Группа показателей, характеризующих содержание и организацию процесса управления, в том числе непосредственные результаты и затраты управленческого труда.

3. Группа показателей, характеризующих рациональность организационной структуры и её техническо-организационной уровень. К структурам относится звенность системы управления, уровень централизации функций управления, принятые нормы управляемости, сбалансированность распределения прав и ответственности.

Для оценки эффективности управления важное значение имеет определение соответствия системы управления и ее организационной структуры объекту управления. Это находит выражение в сбалансированности состава функций и целей управления, соответствии численности состава работников объёму и сложности работ, полноте обеспечения требуемой информацией, обеспеченности процессов управления технологическими средствами с учетом их номенклатуры.

Оценивают, насколько структура организации отвечает задачам, целям и факторам, от которых зависит успешность её функционирования. Для этого необходимо тщательно изучить существующую формальную структуру, определить все её слабые и сильные стороны, выяснить, какие недостатки в формальной структуре компенсируют неформальные элементы и насколько эффективно.

Эффективность построения организационной структуры не может быть оценена каким-либо одним показателем. С одной стороны, здесь следует учитывать, насколько структура обеспечивает достижения организацией результатов, соответствующих поставленным перед ней производственно-хозяйственным целям, с другой – насколько её внутреннее построение и процессы функционирования адекватны объективным требованиям к их содержанию, организации и свойствам.

Конечным критерием эффективности при сравнении различных вариантов организационной структуры является наиболее полное и устойчивое достижение целей. Однако довести этот критерий до практически применимых простых показателей, как правило, чрезвычайно трудно. Поэтому целесообразно использовать набор нормативных характеристик аппарата управления: его производительность при переработке информации; оперативность принятия управленческих решений; надёжность аппарата управления; адаптивность и гибкость.

Эффективная организационная структура дает возможность как согласованного взаимодействия всех отделов, служб и должностных лиц в соответствии с принятыми планами, так и возможность самореализации всех работников, которые обеспечивают развитие, прогресс организации.

К структуре управления предъявляется множество требований, отражающих ее ключевое для менеджмента значение. Они учитываются в принципах формирования организационной структуры управления.

Главные из этих принципов могут быть сформулированы следующим образом:

1. Организационная структура управления должна, прежде всего, отражать цели и задачи организации, а, следовательно, быть подчиненной производству и его потребностям.

2. Следует предусматривать оптимальное разделение труда между органами управления и отдельными работниками, обеспечивающее творческий характер работы и нормальную нагрузку.

3. Формирование структуры управления надлежит связывать с определением полномочий и ответственности каждого работника и органа управления, с установлением системы вертикальных и горизонтальных связей между ними.

4. Между функциями и обязанностями, с одной стороны, и полномочиями, и ответственностью с другой, необходимо поддерживать соответствие, нарушение которого приводит к дисфункции системы управления в целом.

5. Организационная структура управления призвана быть адекватной социально-культурной среде организации, оказывающей существенное влияние на решения относительно уровня централизации и детализации, распределения полномочий и ответственности, степени самостоятельности и масштабов контроля руководителей и менеджеров. Практически это означает, что попытки слепо копировать структуры управления, успешно функционирующие в других социально-культурных условиях, не гарантируют желаемого результата. Реализация этих принципов означает необходимость учета при формировании (или перестройке) структуры управления множества различных факторов воздействия на организационную структуру управления.

Таким образом, эффективность организационной структуры управления определяют: оптимальные взаимосвязи, возникающие в процессе управления; политика руководства организации, стиль управления и методы, которые влияют на человеческое поведение; система функций и полномочий предприятия на различных уровнях управления.

## 1.3 Информационный подход к оценке организационных структур

Информационный подход к оценке управленческих структур первоначально был ориентирован на отображение и анализ пространственно распределенных систем и опирался на аппарат математической теории поля, поэтому он получил название теории информационного поля. В дальнейшем на основе этой теории был получен вариант информационного описания объектов с сосредоточенными параметрами (то есть с выделением дискретных элементов), что более удобно для исследования экономических объектов и процессов.

Можно выделить три этапа отражения действительности, из которых два пассивных — этап чувственного и этап логического отражения, а один активный — этап прагматического отражения. Поскольку единственным продуктом всякого отражения является информация, то продуктами вышеназванных этапов отражения являются чувственная, логическая и прагматическая информации.

Чувственная информация J вводится как мера отраженной в нашем сознании элементной базы системы в формуле

J = , (1)

где А - общее количество каких-либо знаков, воспринимаемых измерительными приборами или нашими органами чувств; ΔА - «квант», с точностью до которого нас интересует воспринимаемая информация, или разрешающая способность прибора.

Логическая информация (сущность) Н, в отличие от J, всегда относящейся к конкретным объектам или свойствам, характеризует целый класс однородных в определенном отношениии объектов или свойств, являясь семантическим синтезом законов логики, правил функционирования системы и ее элементов, образующих функционал ее существования.

Согласно основному закону классической логики Аристотеля собственная сущность (суть) системы обратно пропорциональна объему понятия о ней, т. е.

. (2)

Логическую информацию H можно определить не только через параметры синтезирующей ее системы (человека, автоматизированной информационной системы). Если учесть, что H характеризует не единичный объект, а класс однородных в определенном смысле объектов или свойств, то это значение можно определить через интеграл плотности вероятности f(Ji), полагая, что J имеет значение Ji

. (3)

В частном случае вместо плотности вероятности можно ввести класс однородных объектов с вероятностью qi представив Ji в логарифмической форме; тогда получим

. (4)

Значения qi и pi могут быть не равны, но возможны ситуации (как, например, в случае передачи сигналов по каналам связи, когда qi = pi), что имеет место в формуле Шеннона

. (5)

Прагматическая информация Hц описывается аналогичной моделью, только под J понимается информация о средствах достижения цели, а под n - количество бит информации о средствах на бит информации о цели (результате).

Семантическая информация Н так же, как и прагматическая Hц, может иметь статистическую трактовку, т. е. Нц можно определять аналогично, только в этом случае для практических приложений часто удобнее заменить вероятность недостижения цели pi на сопряженную (’):

 (6)

где  - вероятность достижения цели; qi - вероятность того, что оцениваемая компонента будет использована для достижения цели.

Наряду с рассмотренными основными понятиями информационного подхода, в вводится еще характеристика сложности содержания, — в случае семантической информации, или смысла — в случае прагматической информации.

Сложность (содержание, смысл) С определяется пересечением (в частных случаях — произведением) J и H

. (7)

Соотношение получено в теории информационного поля, и доказательство его требует введения ряда понятий этой теории. Поэтому ограничимся таким пояснением: H характеризует содержание (суть) только единицы чувственной информации, а для того чтобы охарактеризовать сложность всей информации, нужно, естественно, умножить H на количество чувственной информации J. Для случая прагматической информации сложность Сц (смысл информации для достижения поставленной цели) должна определяться с учетом прагматической чувственной информации, влияющей на достижение цели, и прагматической сути Hц .

Для систем организационного управления интерпретация С зависит от конкретных условий его применения. В частности, в зависимости от того, применительно к характеристике всей системы или ее элементов используется С, можно говорить о системной, собственной и взаимной сложности.

Если учесть чувственную информацию I, то получим соотношение, определяющее взаимосвязь системной СС, собственной СО и взаимной СВ сложности системы, т. е.

СС =Со+СВ. (8)

Собственная сложность Со характеризует суммарную сложность (содержание) элементов системы вне связи их между собой (в случае прагматической информации — суммарная сложность элементов, влияющих на достижение цели). Системная сложность СС отражает содержание системы как целого (например, сложность ее использования). Наконец, СВ характеризует степень взаимосвязи элементов в системе (т. е. сложность ее устройства, схемы, структуры).

Обратим внимание на тот факт, что суммарная собственная сложность элементов в устойчивых системах больше, чем системная, т. е.   
Со > СС. Большим, нежели СС, может быть и СВ. Например, сложность телевизора СС для пользователя меньше сложности его конструкции СВ и суммарной сложности СС (возможностей) элементов, из которых собран телевизор. Так что иногда бытующее выражение «целое больше своих частей» не следует понимать буквально. Количественно содержание целого может быть меньше, но качественно его свойства принципиально новы по сравнению со свойствами составляющих его частей.

В изображениях иерархической структуры системы способ вычленения элементов не определен, и ее «читать» можно неодинаково.

Так, элементами можно считать каждую ветвь иерархической структуры, полагая, что она имеет два возможных состояния («участвует» или «не участвует» в принятии решения), т. е. оценивается минимальной единицей чувственной информации J = 1 бит. А можно разделить структуры на элементы так, как показано на рис. 2.2 а, б, которые являются основой для структур на рисунке 2.2 в, г, д, е, и тогда каждый элемент по отношению к системе может оцениваться так же, как участвующий или нет в принятии решения (т. е. его J = 1 бит), но с разными «способностями», которые оцениваются числом его возможных состояний, отражаемых в оценке его Н: для элементов с двумя состояниями (примем для упрощения состояния равновероятными) H=1 бит; для элементов с четырьмя равновероятными состояниями Н = 2 бита и т. д.

. (9)

Оценки J, Н, а, соответственно и С, можно не переводить в привычные для измерения информации единицы — биты, т. е. не прибегать к логарифмической шкале представления оценок, как это показано, например, в работе. При этом получаются более громоздкие вычисления и несколько иная степень различия Н и С (логарифм сжимает шкалу оценок), но характер выводов сохраняется.

Наряду с рассмотренными оценками, характеризующими структурные особенности систем, взаимоотношения частей и целого, часто бывает полезно оценить систему и ее структуру с точки зрения затрат труда на принятие решения в процессе функционирования системы. В частности, большинство управленческих решений связано с выбором исполнителя или адресата из числа сотрудников, подчиненных той или иной вершине оргструктуры. Так, определяя, до сведения каких из подразделений или отдельных сотрудников нужно довести соответствующую директивную или отчетную информацию, ЛПР затрачивает труд на прочтение хотя бы заголовков распределяемых документов, на сопоставление их с наименованиями подчиненных подразделений (или с темами, выполняемыми подчиненными ему сотрудниками), т. е. на переработку определенной информации (которую можно оценить в буквах, словах, абзацах).

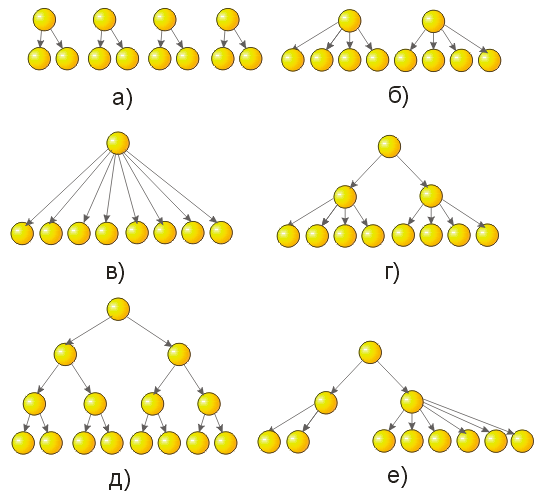


Рисунок 1 – Иерархические структуры систем

Таким образом, реальные затраты управленческого труда, то есть фактический смысл задачи Сф, превосходят ее структурный смысл во столько раз, во сколько раз фактически перерабатываемая для принятия решения информация Jф превосходит структурную J

Сф=, (10)

где Нi -cущность (потенциал) i-гo элемента структуры; Jф- информация того же элемента (которая определяется с учетом числа состояний элемента, то есть числа подчиненных ему составляющих).

Суммарная оценка затрат труда на принятие решений (выбор) при прохождении по структуре сверху вниз важна не только при оценке оргструктур, но и при определении структур банков данных, алгоритмов поиска информации в них.

Для оценки систем можно также использовать характеристику полезной производительности (информационной мощности) N. Можно ввести коэффициент полезного действия γ структуры, определяя его с учетом полных возможностей структуры C∑., и используемых ее возможностей С с точки зрения данной цели :

γ = C/C∑. (11)

Информационный подход к анализу систем имеет широкий спектр приложений. Наряду с рассмотренными оценками структур и процессов принятия решений, что имеет особо важное значение для экономических объектов, подход позволяет получать объединение разнородных критериев при решении многокритериальных задач, проводить сравнительный анализ влияния разнородных нововведений на реализацию целей систем управления, оценивать переходные процессы принятия решений, тенденции развития систем различной физической природы, принципиальные возможности реализации системы как самоорганизующейся, развивающейся, переход ее на новый качественный уровень.

Рассмотренные выше методики и средства системного анализа особенно необходимы в том случае, когда предстоит кардинальная переработка существующей системы управления, или, когда нужно создавать заново структуру системы управления в связи с полностью изменившимися целевыми установками.

# РАЗРАБОТКА ТРЕБОВАНИЙ К ПО ПРОЦЕДУРЫ ОЦЕНКИ ОРГСТРУКТУР

## 2.1 Разработка функциональной модели

В ходе разработки были разработаны следующие модели требований к ПО для оценки оргструктуры предприятия. Основными требованиями стали гибкость, возможность критериальной оценки и экспорта/импорта построенных структур.

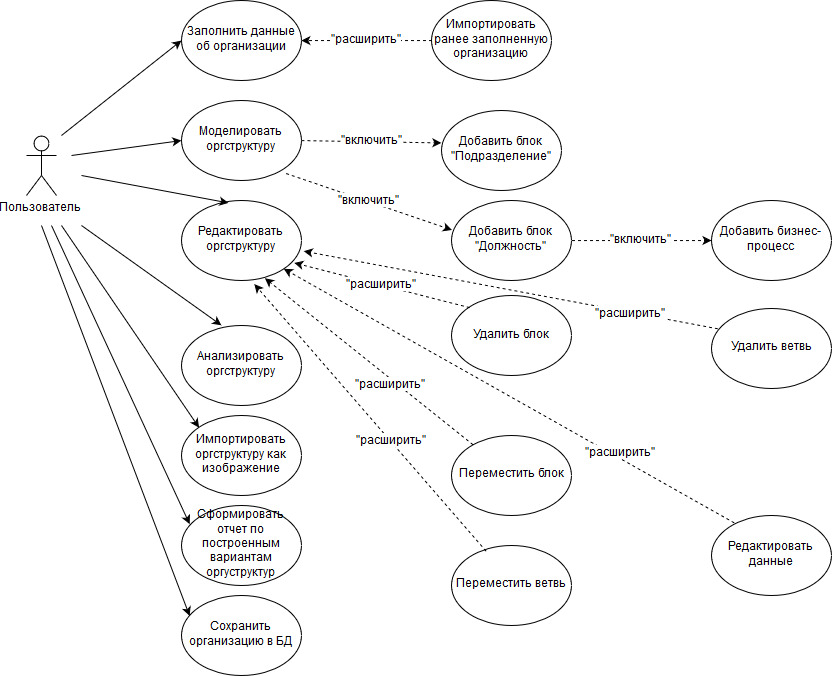


Рисунок 2 – Диаграмма прецедентов

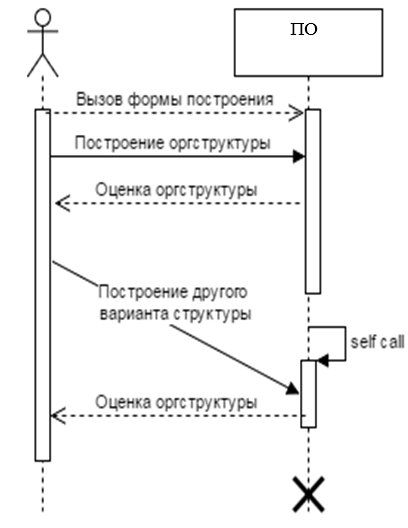


Рисунок 3 — Диаграмма последовательностей

## 2.2 Выбор средств реализации

В качестве инструмента реализации был выбран объектно-ориентированный язык программирования С#, часто используемый на сегодняшний день для быстрого конструирования различных компонентов — от высокоуровневой бизнес логики до системных приложений, использующих низкоуровневый код. В качестве среды разработки была использована передовая среда разработки на языке C# – Visual Studio 2015 – предназначенная для создания интерактивных приложений с пользовательским интерфейсом для настольных компьютеров, рабочих станций, сенсорных дисплеев, информационных терминалов и Интернета. На сегодняшний день это единственная действительно двусторонняя визуальная среда с широкими функциональными возможностями для быстрой разработки на языке C# приложений с развитым пользовательским интерфейсом и приложений баз данных. Приложения, созданные в среде Visual Studio, отличаются высочайшим быстродействием и компактностью, а также обладают полнофункциональным пользовательским интерфейсом и способностью подключаться практически к любой базе данных или другому источнику данных, не требуя дополнительной доработки.

Visual Studio включает в себя мощный интегрированный набор инструментов визуальную двустороннюю среду для создания интерактивных приложений с пользовательским интерфейсом и расширяемую платформу визуальных компонентов, содержащую свыше 250 стандартных классов и компонентов. Многочисленные сторонние поставщики и сообщество Visual Studio, насчитывающее в своих рядах более 1,5 миллиона разработчиков, предлагают тысячи дополнительных компонентов Visual Studio практически для каждого конкретного приложения, технологии, функции или задачи.

# РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ПО ОЦЕНКЕ ОРГСТРУКТУР

## Описание процесса разработки

Ключевым решением и отправной точкой разработки стало использование пользовательского класса Block, которой является основной единицей интерфейса – элементом оргструктуры. В полях класса Block реализовано хранение данных о типе элемента оргструктуры, его уровне, родителе, детях и всей ветке. Таким образом, вся информация, необходимая для работы с ним, будет легко доступна и использована для различных манипуляций с объектом. Поля класса представлены ниже:

public List<Block> MyChilds = new List<Block>();

public List<Block> Branch = new List<Block>();

public int lvl = 0;

public int number = 0;

public int margin = 30;

public int newloc = 0;

public Block first;

public Block myParent;

public string TypeOfBlock;

Интерфейс объектов класса Block имеет следующий вид (рис.4):

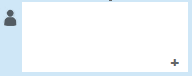
 

Рисунок 4 – Интерфейс объектов класса Block

Блок представляет собой прозрачную панель с текстовым блоком, также имеется индикатор типа блока (подразделение или должность). Кнопки навигации:

1. Добавление блока в ветку(«+»).

Кнопка добавления вызывает контекстное меню для выбора типа блока. Далее происходит вызов метода Add класса Block, который выполняет создание объекта класса заданного типа, а также отрисовку его в корректном положении относительно родительского блока, и в соответствие с этим передвижение всех соседствующих блоков. Ниже указана логика вычисления положения блока:

// первый блок распологается непосредственно под родительским

if (papa.MyChilds.Count == 0)

{

newBlock.Location = new Point(papa.Location.X, papa.Location.Y + papa.Height + margin);

papa.MyChilds.Add(newBlock);

pra = newBlock;

do

{

pra.Branch.Add(newBlock);

pra = pra.myParent;

}

while (pra != null);

}

//все последующие - добавляются справа за последним элементом данного уровня

else

{

newBlock.Location = new Point(papa.MyChilds[papa.MyChilds.Count - 1].Location.X + papa.Width + margin, papa.MyChilds[papa.MyChilds.Count - 1].Location.Y);

papa.MyChilds.Add(newBlock);

pra = newBlock;

do

{

pra.Branch.Add(newBlock);

pra = pra.myParent;

}

while (pra != null);

//если добавление ОТ 1 уровня (на второй)

if (papa.lvl == 1)

{

right = papa.MyChilds[papa.MyChilds.Count - 2];

while (right.MyChilds.Count > 0)

{

right = right.MyChilds[right.MyChilds.Count - 1];

}

newBlock.Location = new Point(right.Location.X + right.Width + margin, papa.MyChilds[papa.MyChilds.Count - 2].Location.Y);

}

//на последующие уровни - необходимо перемещать всех родителей и их родителей, находящихся справа...

else

{

//находим индекс родителя

pra = papa;

while (pra.myParent != null)

{

pra.Location = new Point(pra.MyChilds[0].Location.X + (pra.MyChilds[pra.MyChilds.Count - 1].Location.X - pra.MyChilds[0].Location.X) / 2, pra.Location.Y);

//берем всех правых братьев родителя

for (int idx = pra.number ; idx < pra.myParent.MyChilds.Count; idx++)

{

//и двигаем ВСЮ ИХ ВЕТКУ вправа

foreach (Block ch in pra.myParent.MyChilds[idx].Branch)

{

ch.Location = new Point(ch.Location.X + pra.Width + margin, ch.Location.Y);

}

}

//index = 0;

pra = pra.myParent;

}

}

}

//вычисляем координаты для блока1 (середину)

newBlock.first.Location = new Point(newBlock.first.MyChilds[0].Location.X + (newBlock.first.MyChilds[newBlock.first.MyChilds.Count - 1].Location.X - newBlock.first.MyChilds[0].Location.X) / 2, newBlock.first.Location.Y);

tc.SelectedTab.Controls.Add(newBlock);

//вызов метода отрисовки линий связи

drawLines();

В конце метода происходит вызов отрисовки связей между блоками.

Также блоку было добавлено контекстное меню, содержащее в себе следующие пункты:

* копировать (блок, ветвь);
* вставить (блок ниже, ветвь ниже, блок с заменой текущего, ветвь с заменой текущего);
* удалить;
* изменить тип блока.

1. Просмотр и редактирование процессов (подроцессов, функций, операций), осуществляющихся подразделением («▼»).

Для блоков типа «Подразделение» доступна возможность редактирования и просмотра процессов в отдельном окне. Интерфейс данного окна представляет собой четырехуровневый элемент TreeView.

В соответствие с функциональными требованиями к приложению, была разработана возможность экспорта/импорта оргструктур. Сохранение организации происходит в базу данных MS Access посредством классов OleDbCommand и OleDbDataReader, предоставляемых библиотекой System.Data.OleDb. Для хранения данных была спроектирована база (рис.5), состоящая из пяти связанных таблиц: Организации, Структуры, Блоки, Функции, Характеристики.

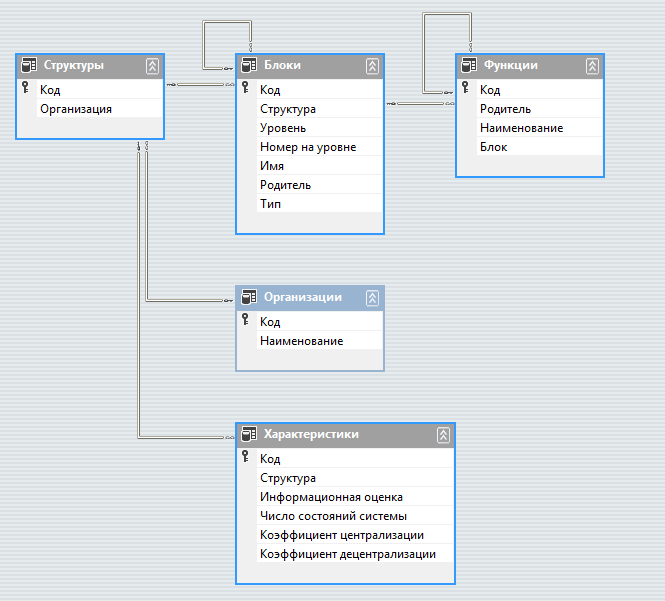


Рисунок 5 – Схема данных БД для хранения оргуструктур

В качестве примера рассмотрим процедуру частично, а именно сохранение организации и связанной с ней структурой:

command = new OleDbCommand();

OleDbDataReader reader;

oleDbConnection1.Open();

command.Connection = oleDbConnection1;

string query = "Select @@Identity";

command.CommandText = "SELECT (Код) FROM Организации WHERE (Наименование='" + org\_name.Text + "')";

if (command.ExecuteScalar() != null)

{

reader = command.ExecuteReader();

reader.Read();

id\_org = (int)reader["Код"];

reader.Close();

}

else

{

command.CommandText = "insert into Организации (наименование) values ('" + org\_name.Text + "')";

command.ExecuteNonQuery();

command.CommandText = query;

id\_org = (int)command.ExecuteScalar();

}

foreach (TabPage tp in tabControl.TabPages)

{

if (tp.Text == "+")

break;

else

{

command.CommandText = "insert into Структуры (Организация) values (" + id\_org + ")";

command.ExecuteNonQuery();

command.CommandText = query;

id\_str = (int)command.ExecuteScalar();

}

}

oleDbConnection1.Close();

Аналогичным образом происходит загрузка организации, выбранной пользователем через диалоговое окно. Пример запроса на выборку функций для блока подразделения:

command\_func.CommandText = "SELECT \* FROM (Функции AS t1 LEFT JOIN Функции AS t2 ON t1.Родитель=t2.Код) WHERE (t1.Блок=" + id\_block + ") ORDER BY t1.Родитель";

Следующая возможность, реализованная в приложении – сохранение структуры как изображения. Для этого процедура вычисляет самый левый, правый и нижний блоки, по которым и вычисляются границы и размеры изображения. Затем происходит автоскроллинг к началу изображения и последовательное копирование графики с области видимости на битмап методом CopyFromScreen(). Это происходит до тех пор, пока не будет достигнут край изображения, после чего изображение сохраняется по выбранному пользователем через диалоговое окно пути.

Расчет оценок оргструктур производится по формулам, представленным в таблице 1.

Таблица 1. Показатели оценки организационной структуры управления

|  |  |  |
| --- | --- | --- |
| Коэффициент | Формула | Реализация |
| Коэффициент централизации управления | Количество работников центрального аппарата управления  /  Общее количество работников управления | Math.Round(((double)lvls[2] / first\_here.Branch.Count),3) |
| Коэффициент децентрализации управления | Количество уровней аппарата управления  /  Общее количество работников управления | Math.Round(((double)lvls.Length / first\_here.Branch.Count), 3) |
| Информационная оценка | Log(Общее количество работников управления) | Math.Round(Math.Log10(first\_here.Branch.Count), 3) |
| Количество состояний системы | Log2(Общее количество работников управления) | Math.Round(Math.Log(first\_here.Branch.Count, 2), 3) |

Предусмотрен вывод отчета в документ MS Word посредством компонента Microsoft.Office.Interop.Word. Отчет представляет пользователю изображения всех построенных в системе структур с их оценкой.

private void ImgStr\_Click(object sender, EventArgs e)

{

Bitmap largeBmp = CreateImg(tabControl.SelectedTab);

SaveFileDialog saveFileDialog = new SaveFileDialog();

saveFileDialog.Filter = "Изображения .bmp (\*.bmp)|\*.bmp";

saveFileDialog.RestoreDirectory = true;

saveFileDialog.CreatePrompt = false;

if (saveFileDialog.ShowDialog() == DialogResult.OK)

{

FileStream fstream = new FileStream(saveFileDialog.FileName, FileMode.Create);

largeBmp.Save(fstream, System.Drawing.Imaging.ImageFormat.Bmp);

fstream.Close();

}

}

private void CreateReport\_Click(object sender, EventArgs e)

{

Microsoft.Office.Interop.Word.Application wordApp;

Microsoft.Office.Interop.Word.Document wordDoc;

wordApp = new Microsoft.Office.Interop.Word.Application();

wordDoc = wordApp.Documents.Add();

wordDoc.Characters.Last.Select();

wordApp.Selection.InsertAfter("Варианты организационных структур для " + this.org\_name.Text);

wordDoc.Characters.Last.Select();

wordApp.Selection.InsertAfter("\r");

foreach (TabPage tp in tabControl.TabPages)

{

if (tp.Text != "+")

{

tabControl.SelectedTab = tp;

Clipboard.SetImage(CreateImg(tp));

wordDoc.Characters.Last.Select();

wordApp.Selection.InsertParagraphAfter();

wordApp.Selection.Paste();

wordDoc.Characters.Last.Select();

wordApp.Selection.InsertAfter("\r");

wordDoc.Tables.Add(wordDoc.Characters.Last, 2, 4);

Microsoft.Office.Interop.Word.Table tbl = wordDoc.Tables[wordDoc.Tables.Count];

tbl.AllowAutoFit = true;

CreateResult();

result.Show();

tbl.Cell(1, 1).Range.Text = result.dataGridView.Rows[0].Cells[0].Value.ToString();

tbl.Cell(1, 2).Range.Text = result.dataGridView.Rows[1].Cells[0].Value.ToString();

tbl.Cell(1, 3).Range.Text = result.dataGridView.Rows[2].Cells[0].Value.ToString();

tbl.Cell(1, 4).Range.Text = result.dataGridView.Rows[3].Cells[0].Value.ToString();

tbl.Cell(2, 1).Range.Text = result.dataGridView.Rows[0].Cells[tp.Text].Value.ToString();

tbl.Cell(2, 2).Range.Text = result.dataGridView.Rows[1].Cells[tp.Text].Value.ToString();

tbl.Cell(2, 3).Range.Text = result.dataGridView.Rows[2].Cells[tp.Text].Value.ToString();

tbl.Cell(2, 4).Range.Text = result.dataGridView.Rows[3].Cells[tp.Text].Value.ToString();

tbl.Borders.InsideLineStyle = Microsoft.Office.Interop.Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.InsideLineWidth = Microsoft.Office.Interop.Word.WdLineWidth.wdLineWidth075pt;

tbl.Borders.OutsideLineStyle = Microsoft.Office.Interop.Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.OutsideLineWidth = Microsoft.Office.Interop.Word.WdLineWidth.wdLineWidth075pt;

wordDoc.Characters.Last.Select();

wordApp.Selection.InsertParagraphAfter();

}

}

wordDoc.Content.Select();

wordApp.Selection.ParagraphFormat.Alignment = Microsoft.Office.Interop.Word.WdParagraphAlignment.wdAlignParagraphCenter;

wordApp.Selection.Font.Size = 14;

wordApp.Selection.Font.Name = "Times new Roman";

wordApp.Visible = true;

}

Для ведения построения нескольких вариантов структур в разных вкладках используется компонент TabControl, добавление структур на который происходит фиктивной вкладкой «+» и обработкой события её выбора:

public void addPage\_Selected(object sender, EventArgs e)

{

if (tabControl.SelectedTab.Text == "+")

{

tabControl.SelectedTab.AutoScrollMinSize = new System.Drawing.Size(3000, 3000);

currentTab = getNumber(tabControl.TabPages[tabControl.TabPages.Count - 2].Text) + 1;

Block nb = new Block();

nb.lvl = 1;

nb.ContextMenu = menu;

nb.BlockData.ContextMenu = menu;

nb.Extend.Visible = false;

nb.Name = "block" + currentTab + "\_1";

nb.first = nb;

nb.Location = new Point(1500, 20);

nb.Branch.Add(nb);

TabPage new\_tp = new TabPage("+");

tabControl.TabPages[tabControl.TabPages.Count - 1].Text = "Оргструктура" + currentTab;

tabControl.TabPages[tabControl.TabPages.Count - 1].Name = "tabPage" + currentTab;

new\_tp.Name = "addPage";

new\_tp.Text = "+";

new\_tp.Paint += Page\_DrawLines;

new\_tp.AutoScroll = true;

new\_tp.SetAutoScrollMargin((tabControl.Width - block1\_1.Width) / 2, 0);

new\_tp.BackColor = Color.FromArgb(207, 231, 247);

tabControl.TabPages.Add(new\_tp);

tabControl.SelectedTab.Controls.Add(nb);

nb.drawLines();

tabControl.SelectedTab.ScrollControlIntoView(nb);

}

else

{

currentTab = getNumber(tabControl.SelectedTab.Text);

Block nb = ((Block)((TabControl)sender).SelectedTab.Controls.Find("block" + currentTab + "\_1", false).FirstOrDefault());

tabControl.SelectedTab.ScrollControlIntoView(nb);

nb.drawLines();

}

}

Также для перерисовки линий связи была добавлена обработка события Paint для TabControl’а:

public void Page\_DrawLines(object sender, EventArgs e)

{

try

{

currentTab = getNumber(tabControl.SelectedTab.Text);

((Block)((TabPage)sender).Controls.Find("block" + currentTab + "\_1", false).FirstOrDefault()).drawLines();

}

catch { }

}

При разработке макета приложения были учтены основные принципы популярного стиля Google Material Design. В связи с этим максимально упрощены стандартные стили и 3D-эффекты элементов, добавлена пользовательская панель навигации, и в этой связи описана обработка события перемещения форм:

private void Form\_MouseDown(object sender, MouseEventArgs e)

{

isDragging = true;

lastCursor = Cursor.Position;

lastForm = this.Location;

}

private void Form\_MouseMove(object sender, MouseEventArgs e)

{

if (isDragging)

{

this.Location = Point.Add(lastForm, new Size(Point.Subtract(Cursor.Position, new Size(lastCursor))));

}

}

private void Form\_MouseUp(object sender, MouseEventArgs e)

{

isDragging = false;

}

## 3.2 Инструкция пользователя

В начале работы в среде построения организационных структур необходимо ввести название вашей организации в поле ввода вверху экрана(рис.6):

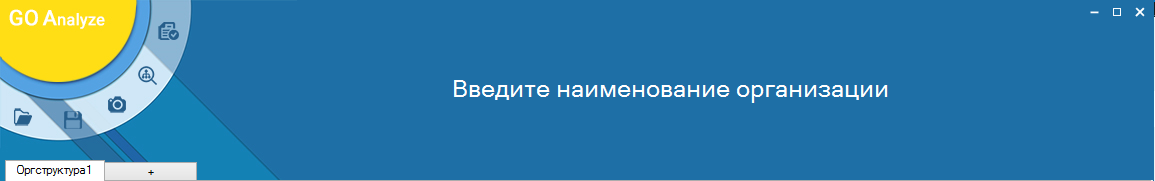


Рисунок 6 – Ввод названия организации

Для добавления структурной единицы нажмите на кнопку «+» центрального блока и выберете необходимый тип в контекстном меню (рис. 7).

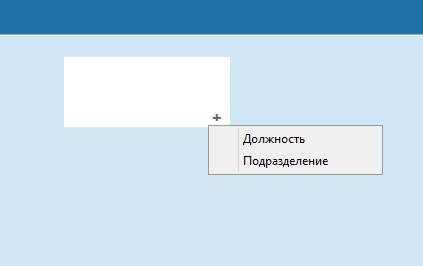


Рисунок 7 – Добавление структурной единицы

После добавления блока следует ввести его наименование. Для добавления процессов подразделения следует нажать на кнопку «▼», после чего откроется форма, представленная на рисунке 8. Необходимо вызвать контекстное меню через наименование блока и выбрать пункт «Добавить ниже». Добавление подпроцессов, функций и операций, происходит аналогичным образом. Также в контекстном меню предусмотрена возможность удаления элемента. Переименование происходит после двоеного клика на элементе. После закрытия формы все введенные данные сохраняются и могут быть открыты для просмотра и редактирования позже.

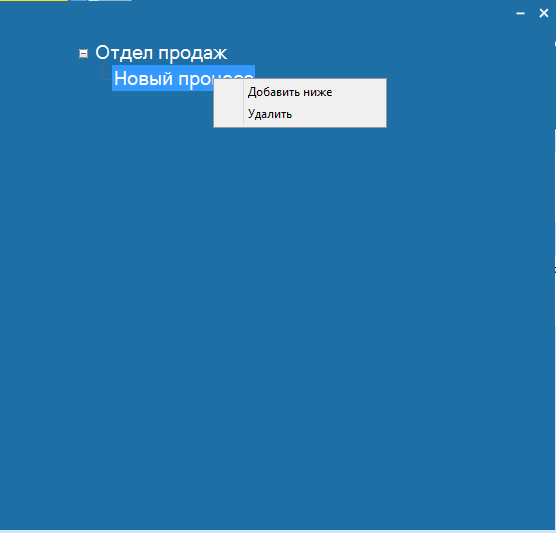


Рисунок 8 – Добавление процессов для подразделения

Для добавления нового варианта структуры необходимо выбрать вкладку «+» на панели в верхней части рабочей области. Удаление вкладки можно произвести через соответствующе ей контекстное меню (рис. 9).

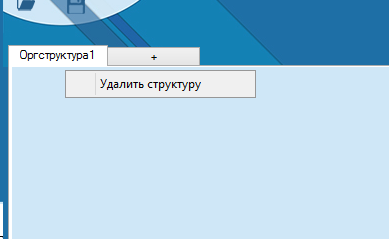


Рисунок 9 – Панель вкладок

Копирование, удаление, вставка блоков и ветвей осуществляется через контекстное меню блока. Например, для копирования ветви из одного варианта в другой с заменой, следует вызвать меню блока, начинающего ветвь, которую необходимо скопировать. Далее выберете пункт Копировать – Ветвь, перейдите к организационной структуре, куда необходимо осуществить вставку. Вызовите контекстное меню блока, который хотите заменить и нажмите Вставить – Ветвь с заменой текущей.

Для оценки построенных структур необходимо нажать на иконку  в верхнем левом углу приложения, после чего будет открыта форма с рассчитанными показателями, представленными в табличном виде (рис. 10).

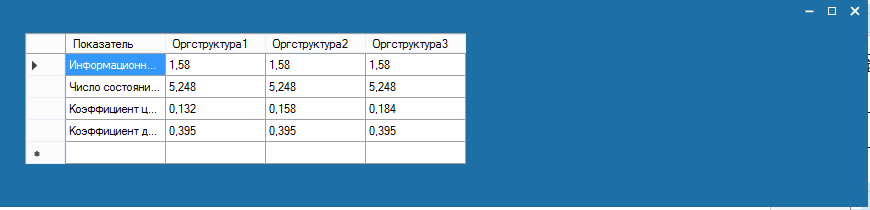


Рисунок 10 – Форма представления оценок оргструктур

Формирование отчета происходит после нажатия на иконку . В отчете будут представлены изображения всех построенных вариантов структур с их оценками (рис. 11).

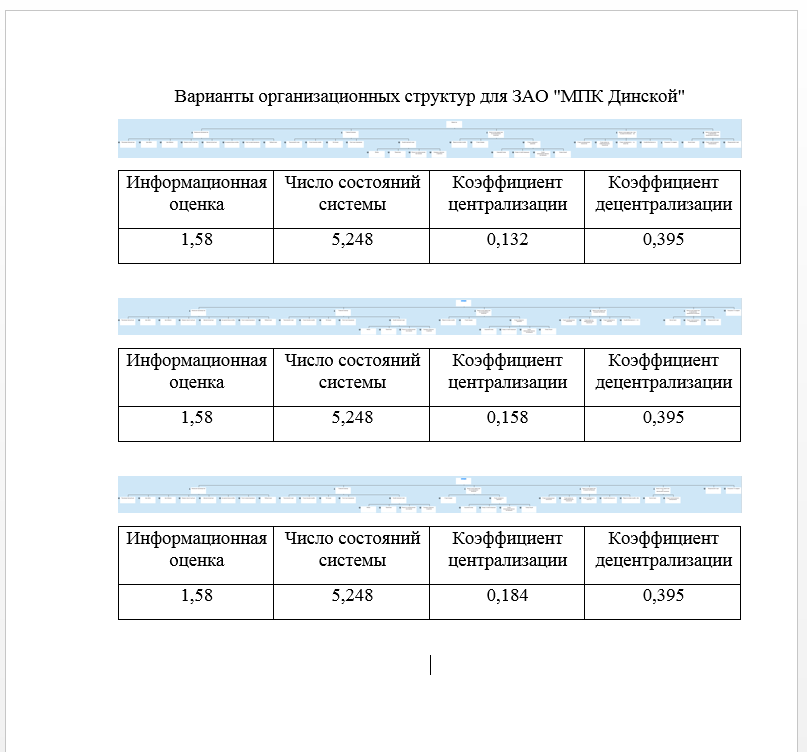


Рисунок 11 – Пример отчета

По завершению работы данные следует сохранить в базу данных – для этого нажмите на кнопку . Чтобы сохранить выбранную структуру как изображение, нажмите на иконку  и выберите путь сохранения файла. При повторной работе с приложением сохраненные данные можно открыть, нажав на кнопку . Далее будет представлена форма для выбора из списка всех сохраненных ранее организаций (рис. 12).



Рисунок 12 – Форма открытия ранее сохраненной организации

Примеры построенных структур представлены в приложении А, а весь программный код в приложении Б.

# ЗАКЛЮЧЕНИЕ

Работая в условиях рыночной экономики и конкуренции, организации должны стремиться к обеспечению высокой степени удовлетворенности клиентов, к высокой эффективности внутренних бизнес-процессов и рентабельности. Помимо этого, в процессе развития организации возникают экономические, технические и технологические проблемы, требующие постоянного совершенствования ее организационной структуры и поиска новых прогрессивных форм управления.

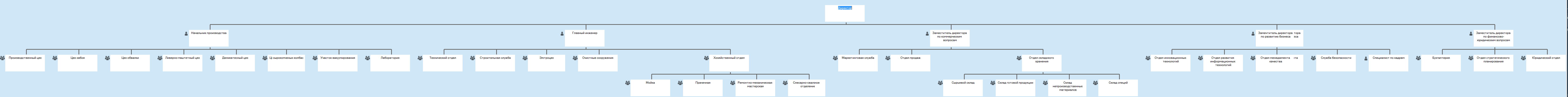
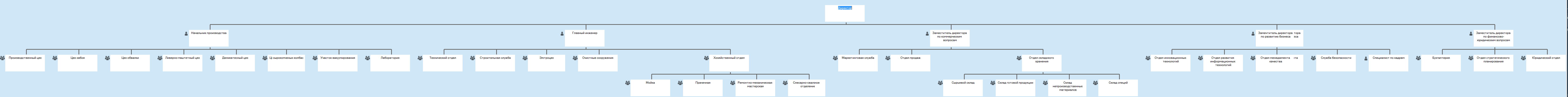
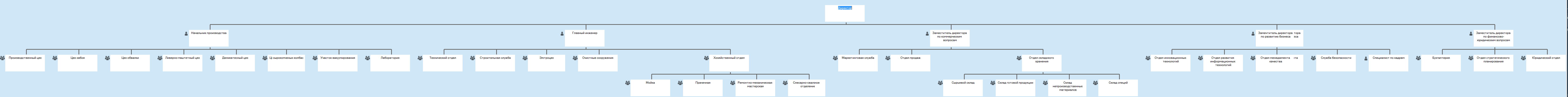
В ходе курсового проекта была разработана подсистема для оценки организационных структур организации, основанная на информационном подходе. Приложение позволяет строить оргструктуры любого типа и оценивать по большинству существующих критериев, обладает понятным и многофункциональным интерфесом, имеет весь необходимый инструментарий для разработки множества вариантов структур. При проведении тестирования работы приложения ошибок не выявленно.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

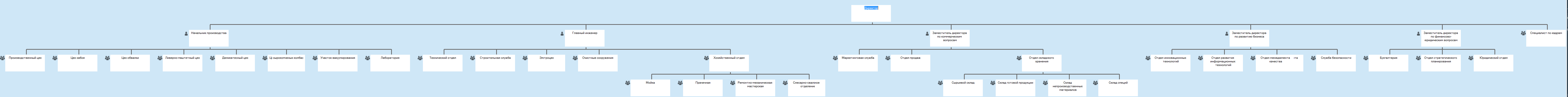
1. Гриценко Ю.Б. Архитектура предприятия [Электронный ресурс]: учебное пособие/ Гриценко Ю.Б.— Электрон. текстовые данные. — Томск: Томский государственный университет систем управления и радиоэлектроники, — 264 c.— Режим доступа: http://www.iprbookshop.ru/14005.— ЭБС «IPRbooks»
2. Абызова Е. В. Методика формирования организационной структуры системы управления Краснодарским краевым союзом потребительских обществ и ее когнитивное моделирование // Научный журнал Куб ГАУ [Электронный ресурс]. – Краснодар: Куб ГАУ, 2015. - № .- Режим доступа: http: //eg. Kubagro. ru/ 2015/ .- 0,9 п. л.
3. Абызова Е. В. Дерево целей и функций системы управления региональной потребительской кооперацией / Абызова Е.В., Першакова Т.В., Дейнека А.В., // Научный журнал Куб ГАУ [Электронный ресурс]. – Краснодар: Куб ГАУ, 2005. - № .- Режим доступа: http: //eg. Kubagro. ru/ 2015/.- 0,7 п. л.(авт. 0,5)
4. Абызова Е. В. Информационный анализ организационных структур систем управления региональной потребительской кооперацией / Абызова Е.В., Першакова Т.В., Дейнека А.В., // Научный журнал Куб ГАУ [Электронный ресурс]. – Краснодар: Куб ГАУ, 2005. - № .- Режим доступа: http: //eg. Kubagro. ru/ 2015/.- 0,8 п. л. (авт. 0,6).
5. Руководство по программированию на C#. [Электронный ресурс]. .- Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/
6. Бабешко Л.О. Коллокационные модели прогнозирования в финансовой сфере.- М.: «Экзамен», 2014г. – 288с.
7. Барановская Т.П. Системный анализ организационной структуры управления автодорожной отраслью региона /  Т.П. Барановская, О.К. Безродный, В.И. Лойко // Политематический сетевой электронный научный журнал Кубанского государственного аграрного университета (Научный журнал КубГАУ) [Электронный ресурс]. – Краснодар: КубГАУ, 2004. – №06(008). С. 168 – 201. – IDA [article ID]: 0080406018. – Режим доступа: http://ej.kubagro.ru/2004/06/pdf/18.pdf, 2,125 у.п.л.
8. Александр Бабич. Введение в UML: Информация [Электронный ресурс] - Режим доступа: <http://www.intuit.ru/studies/courses/1007/229/info>
9. C# 4.0 и платформа .NET 4 для профессионалов. Кристиан Нейгел, Билл Ивьен, Джей Глинн, Карли Уотсон, Морган Скиннер.. –М: «Диалектика», 2011. -1440с.
10. Организационная структура управления. [Электронный ресурс] - Режим доступа: http://www.grandars.ru/college/ekonomika-firmy/organizacionnaya-struktura.html

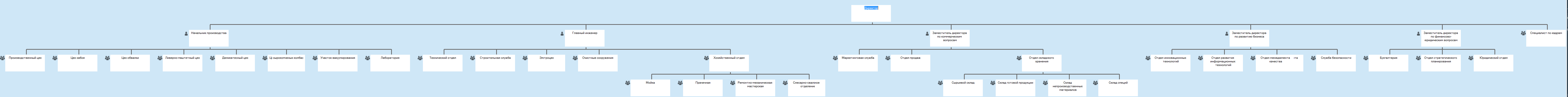
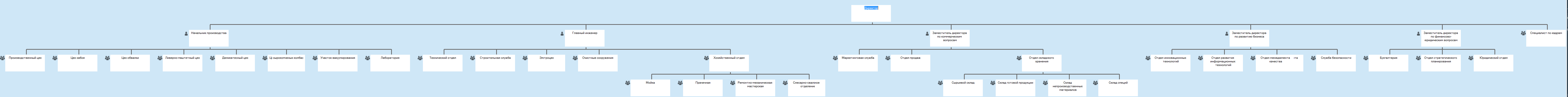
# ПРИЛОЖЕНИЕ А

Первый вариант организационной структуры ЗАО «МПК Динской»

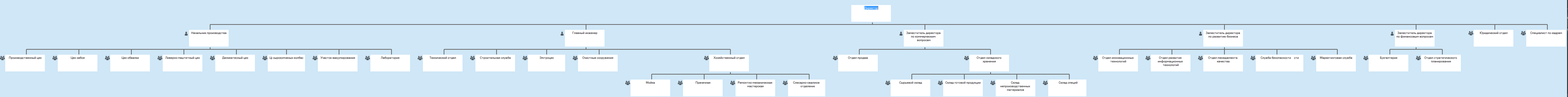
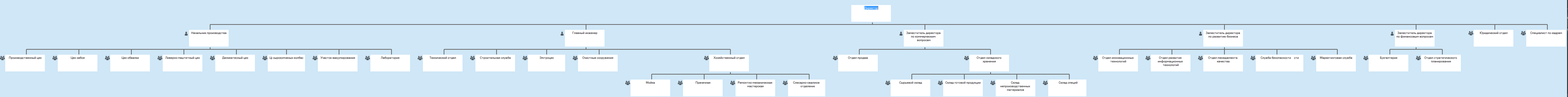
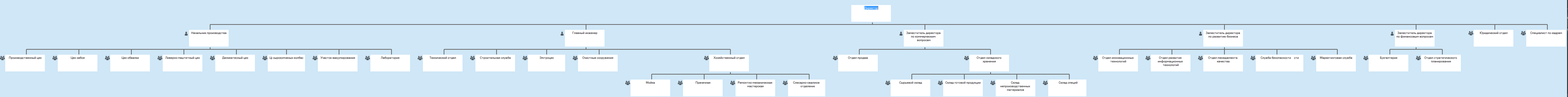


Второй вариант организационной структуры ЗАО «МПК Динской»





Третий вариант организационной структуры ЗАО «МПК Динской»



# ПРИЛОЖЕНИЕ Б

**Листинг файла Form1.cs**

using System;

using System.Data.OleDb;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text.RegularExpressions;

using System.Windows.Forms;

namespace GOA

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

Block copyBlock;

Block currentBlock;

OleDbCommand command;

ContextMenu menu = new ContextMenu();

ContextMenu delPage = new ContextMenu();

Results result = new Results();

MenuItem mi\_copy, mi\_copy\_block, mi\_copy\_branch, mi\_paste, mi\_paste\_block\_down, mi\_paste\_branch\_down, mi\_paste\_block\_excange, mi\_paste\_branch\_excange, mi\_delete, typeItem, tp\_delete;

bool isCopyBlock = false, isCopyBranch = false;

int id\_org, id\_str, id\_par, id\_block, id\_node\_par, currentTab;

int pos\_x, pos\_y;

private bool isDragging = false;

private Point lastCursor;

private Point lastForm;

private void Form1\_Load(object sender, EventArgs e)

{

CreateMenu();

block1\_1.lvl = 1;

block1\_1.first = block1\_1;

block1\_1.Branch.Add(block1\_1);

block1\_1.ContextMenu = menu;

block1\_1.BlockData.ContextMenu = menu;

block1\_1.Extend.Visible = false;

tabPage1.AutoScroll = true;

tabPage1.HorizontalScroll.Visible = true;

tabPage1.AutoScrollMinSize = new System.Drawing.Size(3000, 3000);

block1\_1.Location = new Point(1500, 20);

tabPage1.AutoScrollPosition = new Point(1500 - tabControl.Width / 2 + block1\_1.Width / 2, 0);

tabPage1.SetAutoScrollMargin((tabControl.Width - block1\_1.Width) / 2, 0);

tabControl.Selected += addPage\_Selected;

tabControl.TabPages[0].Paint += Page\_DrawLines;

tabControl.TabPages[1].Paint += Page\_DrawLines;

tabControl.TabPages[0].BackColor = Color.FromArgb(207, 231, 247);

tabControl.TabPages[1].BackColor = Color.FromArgb(207, 231, 247);

tabControl.TabPages[0].AutoScrollPosition = new Point(1500 - tabControl.Width / 2 + block1\_1.Width / 2, 0);

tabControl.TabPages[1].AutoScrollPosition = new Point(1500 - tabControl.Width / 2 + block1\_1.Width / 2, 0);

result.MaximizeBox = false;

result.dataGridView.Columns[0].HeaderText = "Показатель";

result.dataGridView.Rows.Add(4);

result.dataGridView.Rows[0].Cells[0].Value = "Информационная оценка";

result.dataGridView.Rows[1].Cells[0].Value = "Число состояний системы";

result.dataGridView.Rows[2].Cells[0].Value = "Коэффициент централизации";

result.dataGridView.Rows[3].Cells[0].Value = "Коэффициент децентрализации";

this.BackColor = Color.FromArgb(30, 111, 166);

org\_name.BackColor = Color.FromArgb(30, 111, 166);

this.MouseDown += Form\_MouseDown;

this.MouseMove += Form\_MouseMove;

this.MouseUp += Form\_MouseUp;

OpenStruc.FlatAppearance.BorderColor = Color.FromArgb(206, 230, 247);

SaveStruc.FlatAppearance.BorderColor = Color.FromArgb(103, 150, 189);

ImgStr.FlatAppearance.BorderColor = Color.FromArgb(206, 230, 247);

AnalyzeStr.FlatAppearance.BorderColor = Color.FromArgb(206, 230, 247);

CreateReport.FlatAppearance.BorderColor = Color.FromArgb(138, 179, 211);

}

private void Form\_MouseDown(object sender, MouseEventArgs e)

{

isDragging = true;

lastCursor = Cursor.Position;

lastForm = this.Location;

}

private void Form\_MouseMove(object sender, MouseEventArgs e)

{

if (isDragging)

{

this.Location = Point.Add(lastForm, new Size(Point.Subtract(Cursor.Position, new Size(lastCursor))));

}

}

private void Form\_MouseUp(object sender, MouseEventArgs e)

{

isDragging = false;

}

public void addPage\_Selected(object sender, EventArgs e)

{

if (tabControl.SelectedTab.Text == "+")

{

tabControl.SelectedTab.AutoScrollMinSize = new System.Drawing.Size(3000, 3000);

currentTab = getNumber(tabControl.TabPages[tabControl.TabPages.Count - 2].Text) + 1;

Block nb = new Block();

nb.lvl = 1;

nb.ContextMenu = menu;

nb.BlockData.ContextMenu = menu;

nb.Extend.Visible = false;

nb.Name = "block" + currentTab + "\_1";

nb.first = nb;

nb.Location = new Point(1500, 20);

nb.Branch.Add(nb);

TabPage new\_tp = new TabPage("+");

tabControl.TabPages[tabControl.TabPages.Count - 1].Text = "Оргструктура" + currentTab;

tabControl.TabPages[tabControl.TabPages.Count - 1].Name = "tabPage" + currentTab;

new\_tp.Name = "addPage";

new\_tp.Text = "+";

new\_tp.Paint += Page\_DrawLines;

new\_tp.AutoScroll = true;

new\_tp.SetAutoScrollMargin((tabControl.Width - block1\_1.Width) / 2, 0);

new\_tp.BackColor = Color.FromArgb(207, 231, 247);

tabControl.TabPages.Add(new\_tp);

tabControl.SelectedTab.Controls.Add(nb);

nb.drawLines();

tabControl.SelectedTab.ScrollControlIntoView(nb);

}

else

{

currentTab = getNumber(tabControl.SelectedTab.Text);

Block nb = ((Block)((TabControl)sender).SelectedTab.Controls.Find("block" + currentTab + "\_1", false).FirstOrDefault());

tabControl.SelectedTab.ScrollControlIntoView(nb);

nb.drawLines();

}

}

public void Page\_DrawLines(object sender, EventArgs e)

{

try

{

currentTab = getNumber(tabControl.SelectedTab.Text);

((Block)((TabPage)sender).Controls.Find("block" + currentTab + "\_1", false).FirstOrDefault()).drawLines();

}

catch { }

}

public void CreateMenu()

{

mi\_copy = new MenuItem { Name = "Копировать", Text = "Копировать" };

mi\_copy\_block = new MenuItem { Name = "Блок", Text = "Блок" };

mi\_copy\_branch = new MenuItem { Name = "Ветвь", Text = "Ветвь" };

mi\_copy\_block.Click += CopyMenuItem\_Click;

mi\_copy\_branch.Click += CopyMenuItem\_Click;

mi\_paste = new MenuItem { Name = "Вставить", Text = "Вставить" };

mi\_paste\_block\_down = new MenuItem { Name = "Блок ниже", Text = "Блок ниже" };

mi\_paste\_block\_down.Click += menuPasteBlockDown\_Click;

mi\_paste\_branch\_down = new MenuItem { Name = "Ветвь ниже", Text = "Ветвь ниже" };

mi\_paste\_branch\_down.Click += menuPasteBranchDown\_Click;

mi\_paste\_block\_excange = new MenuItem { Name = "Блок с заменой", Text = "Блок с заменой текущего" };

mi\_paste\_block\_excange.Click += menuPasteBlockExcange\_Click;

mi\_paste\_branch\_excange = new MenuItem { Name = "Ветвь с заменой", Text = "Ветвь с заменой текущей" };

mi\_paste\_branch\_excange.Click += menuPasteBranchExcange\_Click;

mi\_delete = new MenuItem { Name = "Удалить", Text = "Удалить" };

mi\_delete.Click += menuDelete\_Click;

typeItem = new MenuItem { Name = "Изменить тип блока", Text = "Изменить тип блока" };

typeItem.Click += ChangeType\_Click;

mi\_paste.MenuItems.AddRange(new[] { mi\_paste\_block\_down, mi\_paste\_branch\_down, mi\_paste\_block\_excange, mi\_paste\_branch\_excange });

mi\_paste.Select += menuVisible;

mi\_copy.MenuItems.AddRange(new[] { mi\_copy\_block, mi\_copy\_branch });

menu.MenuItems.AddRange(new[] { mi\_copy, mi\_paste, mi\_delete, typeItem });

menu.Popup += HideChangeType;

tp\_delete = new MenuItem { Name = "Удалить структуру", Text = "Удалить структуру" };

tp\_delete.Click += (sen, ee) => { tabControl.TabPages.Remove(tabControl.TabPages[(int)delPage.Tag]); };

delPage.MenuItems.Add(tp\_delete);

}

private void tabControl\_MouseUp(object sender, MouseEventArgs e)

{

// проверяем что нажата была правая кнопка

if (e.Button == MouseButtons.Right)

{

// проходим циклом по всем табам для поиска на котором был клик

for (int i = 0; i < tabControl.TabCount - 1; i++)

{

// получаем область таба и проверяем входит ли курсор в него или нет

Rectangle r = tabControl.GetTabRect(i);

if (r.Contains(e.Location))

{

// показываем контекстое меню и сохраняем номер таба

delPage.Tag = i; // сохраняем номер таба

tabControl.TabPages[i].Show();

delPage.Show((sender as TabControl).TabPages[i], e.Location);

}

}

}

}

private void Form1\_SizeChanged(object sender, EventArgs e)

{

tabControl.SelectedTab.AutoScrollPosition = new Point(1500 - tabControl.SelectedTab.HorizontalScroll.Value - tabControl.Width / 2 + block1\_1.Width / 2, 0);

Refresh();

}

private Bitmap CreateImg(TabPage currentPage)

{

Block first = (Block)currentPage.Controls.Find("block" + getNumber(currentPage.Text) + "\_1", false).FirstOrDefault(), left = first, right = first, last = first;

foreach (Block b in currentPage.Controls)

{

b.Extend.Visible = false;

b.AddChild.Visible = false;

}

while (left.MyChilds.Count > 0)

left = left.MyChilds[0];

while (right.MyChilds.Count > 0)

right = right.MyChilds[right.MyChilds.Count - 1];

foreach (Block b in first.Branch)

{

if (b.lvl > last.lvl)

last = b;

}

currentPage.SetAutoScrollMargin(0, 0);

currentPage.AutoScroll = false;

currentPage.AutoScrollPosition = new Point(left.Location.X, 0);

Refresh();

int max = Convert.ToInt32(Math.Ceiling((double)(right.Location.X - left.Location.X + 300) / tabControl.Width)), x = 0, count = 1,

max\_v = Convert.ToInt32(Math.Ceiling((double)(last.Location.Y + 100) / tabControl.Height)), y = 0, count\_v = 1;

//if (max < 1) max = 1;

//if (max\_v < 1) max\_v = 1;

Graphics from\_tab = currentPage.CreateGraphics();

Bitmap largeBmp;

if (max == 1 && max\_v == 1)

{

largeBmp = new Bitmap(currentPage.Width, currentPage.Height, from\_tab);

Graphics g = Graphics.FromImage(largeBmp);

g.CopyFromScreen(currentPage.PointToScreen(Point.Empty), new Point(x, y), tabControl.Size);

}

else

{

largeBmp = new Bitmap(right.Location.X - left.Location.X + left.Width, last.Location.Y + last.Height, from\_tab);

Graphics g = Graphics.FromImage(largeBmp);

while (count\_v <= max\_v)

{

x = 0;

count = 1;

while (count <= max)

{

g = Graphics.FromImage(largeBmp);

Bitmap smallBmp = new Bitmap(tabControl.Width, tabControl.Height, from\_tab);

Refresh();

if (count == max && max > 1)

{

int small\_x = right.Location.X - left.Location.X + left.Width - x;

if (count\_v == max\_v && max\_v > 1)

{

int small\_y = last.Location.Y + last.Height - y;

g.CopyFromScreen(currentPage.PointToScreen(Point.Empty), new Point(x - (tabControl.Width - small\_x), y - (tabControl.Height - small\_y)), tabControl.Size);

}

else

g.CopyFromScreen(currentPage.PointToScreen(Point.Empty), new Point(x - (tabControl.Width - small\_x), y), tabControl.Size);

}

else if (count\_v == max\_v && max\_v > 1)

{

int small\_y = last.Location.Y + last.Height - y;

g.CopyFromScreen(currentPage.PointToScreen(Point.Empty), new Point(x, y - (tabControl.Height - small\_y)), tabControl.Size);

}

else

{

g.CopyFromScreen(currentPage.PointToScreen(Point.Empty), new Point(x, y), tabControl.Size);

}

currentPage.AutoScrollPosition = new Point(-currentPage.AutoScrollPosition.X + tabControl.Width, -currentPage.AutoScrollPosition.Y);

count++;

x += tabControl.Width-10;

}

currentPage.AutoScrollPosition = new Point(left.Location.X, -currentPage.AutoScrollPosition.Y + tabControl.Height);

count\_v++;

y += tabControl.Height;

}

}

currentPage.AutoScroll = true;

foreach (Block b in currentPage.Controls)

{

b.Extend.Visible = true;

b.AddChild.Visible = true;

}

currentPage.AutoScrollPosition = new Point(0, 0);

currentPage.SetAutoScrollMargin(200, 0);

return largeBmp;

}

private void ImgStr\_Click(object sender, EventArgs e)

{

Bitmap largeBmp = CreateImg(tabControl.SelectedTab);

SaveFileDialog saveFileDialog = new SaveFileDialog();

saveFileDialog.Filter = "Изображения .bmp (\*.bmp)|\*.bmp";

saveFileDialog.RestoreDirectory = true;

saveFileDialog.CreatePrompt = false;

if (saveFileDialog.ShowDialog() == DialogResult.OK)

{

FileStream fstream = new FileStream(saveFileDialog.FileName, FileMode.Create);

largeBmp.Save(fstream, System.Drawing.Imaging.ImageFormat.Bmp);

fstream.Close();

}

}

private void CreateReport\_Click(object sender, EventArgs e)

{

Microsoft.Office.Interop.Word.Application wordApp;

Microsoft.Office.Interop.Word.Document wordDoc;

wordApp = new Microsoft.Office.Interop.Word.Application();

wordDoc = wordApp.Documents.Add();

wordDoc.Characters.Last.Select();

wordApp.Selection.InsertAfter("Варианты организационных структур для " + this.org\_name.Text);

wordDoc.Characters.Last.Select();

wordApp.Selection.InsertAfter("\r");

foreach (TabPage tp in tabControl.TabPages)

{

if (tp.Text != "+")

{

tabControl.SelectedTab = tp;

Clipboard.SetImage(CreateImg(tp));

wordDoc.Characters.Last.Select();

wordApp.Selection.InsertParagraphAfter();

wordApp.Selection.Paste();

wordDoc.Characters.Last.Select();

wordApp.Selection.InsertAfter("\r");

wordDoc.Tables.Add(wordDoc.Characters.Last, 2, 4);

Microsoft.Office.Interop.Word.Table tbl = wordDoc.Tables[wordDoc.Tables.Count];

tbl.AllowAutoFit = true;

CreateResult();

result.Show();

tbl.Cell(1, 1).Range.Text = result.dataGridView.Rows[0].Cells[0].Value.ToString();

tbl.Cell(1, 2).Range.Text = result.dataGridView.Rows[1].Cells[0].Value.ToString();

tbl.Cell(1, 3).Range.Text = result.dataGridView.Rows[2].Cells[0].Value.ToString();

tbl.Cell(1, 4).Range.Text = result.dataGridView.Rows[3].Cells[0].Value.ToString();

tbl.Cell(2, 1).Range.Text = result.dataGridView.Rows[0].Cells[tp.Text].Value.ToString();

tbl.Cell(2, 2).Range.Text = result.dataGridView.Rows[1].Cells[tp.Text].Value.ToString();

tbl.Cell(2, 3).Range.Text = result.dataGridView.Rows[2].Cells[tp.Text].Value.ToString();

tbl.Cell(2, 4).Range.Text = result.dataGridView.Rows[3].Cells[tp.Text].Value.ToString();

tbl.Borders.InsideLineStyle = Microsoft.Office.Interop.Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.InsideLineWidth = Microsoft.Office.Interop.Word.WdLineWidth.wdLineWidth075pt;

tbl.Borders.OutsideLineStyle = Microsoft.Office.Interop.Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.OutsideLineWidth = Microsoft.Office.Interop.Word.WdLineWidth.wdLineWidth075pt;

wordDoc.Characters.Last.Select();

wordApp.Selection.InsertParagraphAfter();

}

}

wordDoc.Content.Select();

wordApp.Selection.ParagraphFormat.Alignment = Microsoft.Office.Interop.Word.WdParagraphAlignment.wdAlignParagraphCenter;

wordApp.Selection.Font.Size = 14;

wordApp.Selection.Font.Name = "Times new Roman";

wordApp.Visible = true;

}

int getNumber(string s)

{

Regex regex = new Regex(@"\D");

s = regex.Replace(s, "");

return Convert.ToInt16(s);

}

private void CrossButton\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void RollButton\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

private void SizeButton\_Click(object sender, EventArgs e)

{

if (Size.Width < 1500)

{

pos\_x = Location.X;

pos\_y = Location.Y;

Width = SystemInformation.VirtualScreen.Width;

Height = SystemInformation.VirtualScreen.Height;

Location = new Point(0, 0);

SizeButton.Image = Properties.Resources.size\_min;

}

else

{

Size = new Size(1154, 760);

Location = new Point(pos\_x, pos\_y);

SizeButton.Image = Properties.Resources.size\_max;

}

}

public void CreateResult()

{

int[] lvls = new int[15];

int cur\_i = -1;

currentTab = getNumber(tabControl.SelectedTab.Text);

Block first\_here = (Block)tabControl.SelectedTab.Controls.Find("block" + currentTab + "\_1", false).FirstOrDefault();

foreach (Block b in first\_here.Branch)

{

lvls[b.lvl]++;

}

for (int i = 0; i < result.dataGridView.Columns.Count; i++)

{

if (result.dataGridView.Columns[i].Name == tabControl.SelectedTab.Text)

{

cur\_i = i;

break;

}

}

if (cur\_i == -1)

{

result.dataGridView.Columns.Add(tabControl.SelectedTab.Text, tabControl.SelectedTab.Text);

cur\_i = result.dataGridView.Columns.Count - 1;

}

result.dataGridView.ColumnHeadersVisible = true;

result.dataGridView.Rows[0].Cells[cur\_i].Value = Math.Round(Math.Log10(first\_here.Branch.Count), 3).ToString();

result.dataGridView.Rows[1].Cells[cur\_i].Value = Math.Round(Math.Log(first\_here.Branch.Count, 2), 3).ToString();

result.dataGridView.Rows[2].Cells[cur\_i].Value = Math.Round(((double)lvls[2] / first\_here.Branch.Count), 3).ToString();

result.dataGridView.Rows[3].Cells[cur\_i].Value = Math.Round(((double)lvls.Length / first\_here.Branch.Count), 3).ToString();

}

private void AnalyzeStr\_Click(object sender, EventArgs e)

{

foreach (TabPage tp in tabControl.TabPages)

{

tabControl.SelectedTab = tp;

CreateResult();

}

result.Show();

}

private void OpenStruc\_Click(object sender, EventArgs e)

{

Organizations orgDialog = new Organizations();

orgDialog.OrgList.Items.Clear();

Block first\_here = new Block();

OleDbCommand command\_org = new OleDbCommand();

command\_org.Connection = oleDbConnection1;

command\_org.CommandText = "SELECT DISTINCT (Наименование) FROM Организации";

oleDbConnection1.Open();

OleDbDataReader reader\_org = command\_org.ExecuteReader();

if (reader\_org.HasRows)

{

orgDialog.OrgList.Enabled = true;

while (reader\_org.Read())

{

orgDialog.OrgList.Items.Add(reader\_org["Наименование"]);

}

reader\_org.Close();

}

else

{

orgDialog.OrgList.Text = "Ранее добавленные организации отсутсвуют";

orgDialog.OrgList.Enabled = false;

}

orgDialog.ShowDialog();

if (orgDialog.DialogResult == DialogResult.OK)

{

orgDialog.Hide();

tabControl.Selected -= addPage\_Selected;

this.org\_name.Text = orgDialog.OrgList.SelectedItem.ToString();

foreach (TabPage tp in tabControl.TabPages)

{

tabControl.TabPages.Remove(tp);

tabControl.TabPages.Clear();

}

command\_org.CommandText = "SELECT (Код) FROM Организации WHERE (Наименование='" + orgDialog.OrgList.SelectedItem + "')";

reader\_org = command\_org.ExecuteReader();

reader\_org.Read();

id\_org = (int)reader\_org["Код"];

reader\_org.Close();

OleDbCommand command\_str = new OleDbCommand();

command\_str.Connection = oleDbConnection1;

OleDbDataReader reader\_str;

command\_str.CommandText = "SELECT \* FROM Структуры WHERE (Организация=" + id\_org + ")";

reader\_str = command\_str.ExecuteReader();

if (reader\_str.HasRows)

{

while (reader\_str.Read())

{

id\_str = (int)reader\_str["Код"];

if (tabControl.TabPages.Count > 0)

currentTab = getNumber(tabControl.TabPages[tabControl.TabPages.Count - 1].Text) + 1;

else currentTab = 1;

TabPage new\_tp = new TabPage("tabPage" + currentTab);

new\_tp.Text = "Оргструктура" + currentTab;

new\_tp.Paint += Page\_DrawLines;

new\_tp.AutoScroll = true;

new\_tp.SetAutoScrollMargin((tabControl.Width - block1\_1.Width) / 2, 0);

new\_tp.BackColor = Color.FromArgb(207, 231, 247);

tabControl.TabPages.Add(new\_tp);

tabControl.SelectedTab = new\_tp;

OleDbCommand command\_block = new OleDbCommand();

command\_block.Connection = oleDbConnection1;

OleDbDataReader reader\_block;

command\_block.CommandText = "SELECT \* FROM Блоки WHERE (Структура=" + id\_str + ") AND (Уровень=1)";

reader\_block = command\_block.ExecuteReader();

if (reader\_block.HasRows)

{

reader\_block.Read();

id\_block = (int)reader\_block["Код"];

currentTab = getNumber(tabControl.TabPages[tabControl.TabPages.Count - 1].Text);

first\_here = new Block();

first\_here.lvl = (int)reader\_block["Уровень"];

first\_here.number = (int)reader\_block["Номер на уровне"];

first\_here.BlockData.Text = (string)reader\_block["Имя"];

first\_here.TypeOfBlock = "position";

first\_here.ContextMenu = menu;

first\_here.BlockData.ContextMenu = menu;

first\_here.Extend.Visible = false;

first\_here.Name = "block" + currentTab + "\_1";

first\_here.first = first\_here;

first\_here.Branch.Add(first\_here);

tabControl.SelectedTab.AutoScrollMinSize = new Size(3000, 3000);

first\_here.Location = new Point(1500, 20);

new\_tp.Controls.Add(first\_here);

reader\_block.Close();

command\_block.CommandText = "SELECT \* FROM (Блоки AS b1 LEFT JOIN Блоки AS b2 ON b1.Родитель=b2.Код) LEFT JOIN Блоки AS b3 ON b2.Родитель=b3.Код WHERE (b1.Структура=" + id\_str + ") AND (b1.Родитель IS NOT NULL) ORDER BY b1.Родитель, b1.[Номер на уровне]";

reader\_block = command\_block.ExecuteReader();

if (reader\_block.HasRows)

{

while (reader\_block.Read())

{

Block nb = new Block();

nb.lvl = (int)reader\_block["b1.Уровень"];

nb.number = (int)reader\_block["b1.Номер на уровне"];

nb.BlockData.Text = (string)reader\_block["b1.Имя"];

id\_block = (int)reader\_block["b1.Код"];

if ((string)reader\_block["b1.Тип"] == "department")

{

nb.MyTreeView.treeView.Nodes.Clear();

OleDbCommand command\_func = new OleDbCommand();

command\_func.Connection = oleDbConnection1;

OleDbDataReader reader\_func;

command\_func.CommandText = "SELECT \* FROM (Функции AS t1 LEFT JOIN Функции AS t2 ON t1.Родитель=t2.Код) WHERE (t1.Блок=" + id\_block + ") ORDER BY t1.Родитель";

reader\_func = command\_func.ExecuteReader();

if (reader\_func.HasRows)

{

while (reader\_func.Read())

{

if (reader\_func["t1.Родитель"] == DBNull.Value)

{

TreeNode tn = new TreeNode((string)reader\_func["t1.Наименование"]);

tn.Name = tn.Text;

nb.MyTreeView.treeView.Nodes.Add(tn);

}

else

{

TreeNode[] tns = nb.MyTreeView.treeView.Nodes.Find((string)reader\_func["t2.Наименование"], true);

TreeNode tn = new TreeNode((string)reader\_func["t1.Наименование"]);

tn.Name = tn.Text;

tns[0].Nodes.Add(tn);

}

}

reader\_func.Close();

}

}

Block par;

currentTab = getNumber(tabControl.SelectedTab.Text);

if (reader\_block["b2.Родитель"] == DBNull.Value)

par = first\_here;

else

par = (Block)tabControl.SelectedTab.Controls.Find("block" + currentTab + "\_" + ((int)reader\_block["b2.Уровень"]).ToString() + "\_" + ((int)reader\_block["b3.Номер на уровне"]).ToString() + "(" + ((int)reader\_block["b2.Номер на уровне"]).ToString() + ")", true).FirstOrDefault();

try

{ par.Add(par, 1, nb, (string)reader\_block["b1.Тип"]); }

catch { MessageBox.Show("block" + currentTab + "\_" + (reader\_block["b2.Уровень"]).ToString() + "\_" + (reader\_block["b3.Номер на уровне"]).ToString() + "(" + (reader\_block["b2.Номер на уровне"]).ToString() + ")"); }

}

reader\_block.Close();

}

}

}

reader\_str.Close();

TabPage add\_tp = new TabPage("+");

add\_tp.Name = "addPage";

add\_tp.Text = "+";

add\_tp.Paint += Page\_DrawLines;

add\_tp.AutoScroll = true;

add\_tp.SetAutoScrollMargin((tabControl.Width - block1\_1.Width) / 2, 0);

add\_tp.BackColor = Color.FromArgb(207, 231, 247);

tabControl.TabPages.Add(add\_tp);

tabControl.Selected += addPage\_Selected;

}

oleDbConnection1.Close();

}

tabControl.SelectedTab.ScrollControlIntoView(first\_here);

}

private void TabControl\_Selected(object sender, TabControlEventArgs e)

{

throw new NotImplementedException();

}

private void SaveStruc\_Click(object sender, EventArgs e)

{

if (org\_name.Text == "Введите наименование организации" || org\_name.Text == "")

MessageBox.Show("Введите наименование организации!");

else

{

//проверка заполненности блоков

//foreach (TabPage tp in tabControl.TabPages)

//{

// currentTab = getNumber(tp.Text);

// Block first\_here = (Block)tp.Controls.Find("block" + currentTab + "\_1", false).FirstOrDefault();

// foreach (Block b in first\_here.Branch)

// {

// if (b.BlockData.Text == "")

// {

// MessageBox.Show("Не все блоки заполнены! Проверьте уровень " + b.lvl + " структуры" + tp.Text);

// return;

// }

// }

//}

command = new OleDbCommand();

OleDbDataReader reader;

oleDbConnection1.Open();

command.Connection = oleDbConnection1;

string query = "Select @@Identity";

command.CommandText = "SELECT (Код) FROM Организации WHERE (Наименование='" + org\_name.Text + "')";

if (command.ExecuteScalar() != null)

{

reader = command.ExecuteReader();

reader.Read();

id\_org = (int)reader["Код"];

reader.Close();

command.CommandText = "DELETE \* FROM Организации WHERE (Код=" + id\_org + ")";

command.ExecuteNonQuery();

}

command.CommandText = "insert into Организации (наименование) values ('" + org\_name.Text + "')";

command.ExecuteNonQuery();

command.CommandText = query;

id\_org = (int)command.ExecuteScalar();

foreach (TabPage tp in tabControl.TabPages)

{

if (tp.Text == "+")

break;

else

{

command.CommandText = "insert into Структуры (Организация) values (" + id\_org + ")";

command.ExecuteNonQuery();

command.CommandText = query;

id\_str = (int)command.ExecuteScalar();

currentTab = getNumber(tp.Text);

Block first\_here = (Block)tp.Controls.Find("block" + currentTab + "\_1", false).FirstOrDefault();

foreach (Block b in first\_here.Branch)

{

if (b.myParent != null)

{

command.CommandText = "SELECT (Код) FROM Блоки WHERE (Имя='" + b.myParent.BlockData.Text + "') AND (Структура=" + id\_str + ")";

command.ExecuteNonQuery();

reader = command.ExecuteReader();

reader.Read();

id\_par = reader.GetInt32(0);

reader.Close();

command.CommandText = "insert into Блоки (Структура, Уровень, [Номер на уровне], Имя, Родитель, Тип) values (" + id\_str + "," + b.lvl + "," + b.number + ",'" + b.BlockData.Text + "'," + id\_par + ",'" + b.TypeOfBlock + "')";

command.ExecuteNonQuery();

command.CommandText = query;

id\_block = (int)command.ExecuteScalar();

}

else

{

command.CommandText = "insert into Блоки (Структура, Уровень, [Номер на уровне], Имя, Тип) values (" + id\_str + "," + b.lvl + "," + b.number + ",'" + b.BlockData.Text + "','" + b.TypeOfBlock + "')";

command.ExecuteNonQuery();

command.CommandText = query;

id\_block = (int)command.ExecuteScalar();

}

if (b.TypeOfBlock == "department")

{

foreach (TreeNode tn in b.MyTreeView.treeView.Nodes)

{

command.CommandText = "insert into Функции ( Наименование, Блок) values ('" + tn.Text + "'," + id\_block + ")";

command.ExecuteNonQuery();

command.CommandText = query;

id\_node\_par = (int)command.ExecuteScalar();

SaveNodes(tn, id\_node\_par);

}

}

}

try

{

command.CommandText = "insert into Характеристики (Структура, [Информационная оценка], [Число состояний системы], [Коэффициент централизации], [Коэффициент децентрализации]) values (" +

id\_str + ",'" + Convert.ToSingle(result.dataGridView.Rows[0].Cells[tp.Text].Value) + "','" + Convert.ToSingle(result.dataGridView.Rows[1].Cells[tp.Text].Value) + "','" +

Convert.ToSingle(result.dataGridView.Rows[2].Cells[tp.Text].Value) + "','" + Convert.ToSingle(result.dataGridView.Rows[3].Cells[tp.Text].Value) + "')";

command.ExecuteNonQuery(); ;

}

catch { }

}

}

oleDbConnection1.Close();

MessageBox.Show("Данные успешно сохранены");

}

}

public void SaveNodes(TreeNode tn, int id)

{

string query = "Select @@Identity";

int func = id;

foreach (TreeNode tnch in tn.Nodes)

{

command.CommandText = "insert into Функции (Родитель, Наименование, Блок) values (" + func + ",'" + tnch.Text + "'," + id\_block + ")";

command.ExecuteNonQuery();

command.CommandText = query;

SaveNodes(tnch, (int)command.ExecuteScalar());

}

}

public void HideChangeType(object sender, EventArgs e)

{

if (((ContextMenu)sender).SourceControl.Parent.GetType() == typeof(Block))

currentBlock = (Block)((ContextMenu)sender).SourceControl.Parent;

else if (((ContextMenu)sender).SourceControl.Parent.GetType() == typeof(TabPage))

currentBlock = (Block)((ContextMenu)sender).SourceControl;

if (currentBlock.myParent == null)

{

currentBlock.ContextMenu.MenuItems["Изменить тип блока"].Enabled = false;

currentBlock.BlockData.ContextMenu.MenuItems["Изменить тип блока"].Enabled = false;

}

else

{

currentBlock.ContextMenu.MenuItems["Изменить тип блока"].Enabled = true;

currentBlock.BlockData.ContextMenu.MenuItems["Изменить тип блока"].Enabled = true;

}

}

public void ChangeType\_Click(object sender, EventArgs e)

{

if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Block))

currentBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent;

else if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Panel))

currentBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl;

if (currentBlock.TypeOfBlock == "department")

{

currentBlock.TypeOfBlock = "position";

currentBlock.AddType(currentBlock);

}

else if (currentBlock.TypeOfBlock == "position")

{

currentBlock.TypeOfBlock = "department";

currentBlock.AddType(currentBlock);

}

}

private void menuVisible(object sender, EventArgs e)

{

if (isCopyBlock == false)

{

mi\_paste\_block\_down.Enabled = false;

mi\_paste\_block\_excange.Enabled = false;

}

else

{

mi\_paste\_block\_down.Enabled = true;

mi\_paste\_block\_excange.Enabled = true;

}

if (isCopyBranch == false)

{

mi\_paste\_branch\_down.Enabled = false;

mi\_paste\_branch\_excange.Enabled = false;

}

else

{

mi\_paste\_branch\_down.Enabled = true;

mi\_paste\_branch\_excange.Enabled = true;

}

}

private void menuDelete\_Click(object sender, EventArgs e)

{

if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Block))

currentBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent;

else if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Panel))

currentBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl;

// дети удаляются из всех ветвей

for (int i = currentBlock.Branch.Count - 1; i > -1; i--)

{

Block par = currentBlock;

while (par.myParent != null)

{

par = par.myParent;

par.Branch.Remove(currentBlock.Branch[i]);

}

tabControl.TabPages[tabControl.SelectedIndex].Controls.Remove(currentBlock.Branch[i]);

}

if (currentBlock.myParent.MyChilds.Count > currentBlock.number)

{

//находим самого левого ребенка в удаляемой ветви, до его позиции будем сдвигать братьев удаляемого блока

Block left\_del = currentBlock;

while (left\_del.MyChilds.Count > 0)

left\_del = left\_del.MyChilds[0];

Block left\_next = currentBlock.myParent.MyChilds[currentBlock.number];

while (left\_next.MyChilds.Count > 0)

left\_next = left\_next.MyChilds[0];

int newpos = left\_next.Location.X - left\_del.Location.X;

for (int i = currentBlock.number; i < currentBlock.myParent.MyChilds.Count; i++)

{

foreach (Block b in currentBlock.myParent.MyChilds[i].Branch)

{

b.Location = new Point(b.Location.X - newpos, b.Location.Y);

b.number--;

}

currentBlock.myParent.MyChilds[i].Location = new Point(currentBlock.myParent.MyChilds[i].Location.X, currentBlock.myParent.MyChilds[i].Location.Y);

}

Block par = currentBlock;

while (par.myParent.myParent != null)

{

par = par.myParent;

for (int i = par.number; i < par.myParent.MyChilds.Count; i++)

{

foreach (Block b in par.myParent.MyChilds[i].Branch)

{

b.Location = new Point(b.Location.X - newpos, b.Location.Y);

}

}

}

}

tabControl.TabPages[tabControl.SelectedIndex].Controls.Remove(currentBlock);

currentBlock.myParent.MyChilds.Remove(currentBlock);

currentBlock.myParent.Location = new Point(currentBlock.myParent.MyChilds[0].Location.X + (currentBlock.myParent.MyChilds[currentBlock.myParent.MyChilds.Count - 1].Location.X - currentBlock.myParent.MyChilds[0].Location.X) / 2, currentBlock.myParent.Location.Y);

currentBlock.first.Location = new Point(currentBlock.first.MyChilds[0].Location.X + (currentBlock.first.MyChilds[currentBlock.first.MyChilds.Count - 1].Location.X - currentBlock.first.MyChilds[0].Location.X) / 2, currentBlock.first.Location.Y);

currentBlock.Dispose();

currentBlock.first.drawLines();

}

private void menuPasteBranchExcange\_Click(object sender, EventArgs e)

{

if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Block))

currentBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent;

else if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Panel))

currentBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl;

currentBlock.copyNodes(copyBlock.MyTreeView, currentBlock.MyTreeView);

currentBlock.BlockData.Text = copyBlock.BlockData.Text;

currentBlock.TypeOfBlock = copyBlock.TypeOfBlock;

currentBlock.AddType(currentBlock);

if (currentBlock.TypeOfBlock == "department")

currentBlock.MyTreeView = copyBlock.MyTreeView;

for (int i = currentBlock.Branch.Count - 1; i > -1; i--)

{

if (currentBlock.Branch[i].Name != currentBlock.Name)

{

Block par = currentBlock;

while (par.myParent != null)

{

par = par.myParent;

par.Branch.Remove(currentBlock.Branch[i]);

}

tabControl.TabPages[tabControl.SelectedIndex].Controls.Remove(currentBlock.Branch[i]);

currentBlock.Branch.Remove(currentBlock.Branch[i]);

}

}

currentBlock.MyChilds.Clear();

pasteChilds(currentBlock, copyBlock);

isCopyBranch = false;

}

private void menuPasteBranchDown\_Click(object sender, EventArgs e)

{

if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Block))

currentBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent;

else if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Panel))

currentBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl;

foreach (Block c in copyBlock.Branch)

{

if (currentBlock.Name == c.Name)

{

MessageBox.Show("Ошибка: Нельзя копировать ветвь саму в себя!");

return;

}

}

currentBlock.Add(currentBlock, 1, copyBlock, copyBlock.TypeOfBlock);

currentBlock = currentBlock.MyChilds[currentBlock.MyChilds.Count - 1];

pasteChilds(currentBlock, copyBlock);

isCopyBranch = false;

}

private void pasteChilds(Block parentBlock, Block copyBlock)

{

foreach (Block child in copyBlock.MyChilds)

{

currentBlock = parentBlock;

currentBlock.Add(currentBlock, 1, child, child.TypeOfBlock);

if (child.MyChilds.Count > 0)

{

currentBlock = currentBlock.MyChilds[currentBlock.MyChilds.Count - 1];

pasteChilds(currentBlock, child);

}

}

}

private void CopyMenuItem\_Click(object sender, EventArgs e)

{

if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Block))

copyBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent;

else if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Panel))

copyBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl;

if (((MenuItem)sender).Text == "Блок")

isCopyBlock = true;

else if (((MenuItem)sender).Text == "Ветвь")

isCopyBranch = true;

}

private void menuPasteBlockExcange\_Click(object sender, EventArgs e)

{

if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Block))

currentBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent;

else if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Panel))

currentBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl;

currentBlock.BlockData.Text = copyBlock.BlockData.Text;

if (currentBlock.TypeOfBlock == "department")

currentBlock.MyTreeView = copyBlock.MyTreeView;

isCopyBlock = false;

}

private void menuPasteBlockDown\_Click(object sender, EventArgs e)

{

if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Block))

currentBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent;

else if (((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.GetType() == typeof(Panel))

currentBlock = (Block)((sender as MenuItem).GetContextMenu() as ContextMenu).SourceControl;

currentBlock.Add(currentBlock, 1, copyBlock, copyBlock.TypeOfBlock);

isCopyBlock = false;

}

}

}

**Листинг файла Block.cs**

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Drawing.Drawing2D;

using System.Linq;

using System.Windows.Forms;

namespace GOA

{

public partial class Block : UserControl

{

public List<Block> MyChilds = new List<Block>();

public List<Block> Branch = new List<Block>();

public int lvl = 0;

public int number = 0;

public int margin = 30;

public int newloc = 0;

public Block first;

public Block myParent;

public string TypeOfBlock;

ContextMenu add\_menu = new ContextMenu();

ContextMenu node\_menu = new ContextMenu();

ContextMenu node\_menu\_add = new ContextMenu();

ContextMenu node\_menu\_del = new ContextMenu();

public Tree MyTreeView = new Tree();

TabControl tc;

public Graphics formGraphics;

public Block()

{

InitializeComponent();

CreateAddMenu();

AddChild.ForeColor = Color.FromArgb(102, 102, 102);

Extend.ForeColor = Color.FromArgb(102, 102, 102);

}

public void CreateAddMenu()

{

MenuItem position = new MenuItem { Name = "Должность", Text = "Должность" };

MenuItem department = new MenuItem { Name = "Подразделение", Text = "Подразделение" };

position.Click += AddPosition\_Click;

department.Click += AddDepartment\_Click;

add\_menu.MenuItems.AddRange(new[] { position, department });

}

public void Extend\_Click(object sender, EventArgs e)

{

((Block)(((Button)sender).Parent.Parent)).MyTreeView.treeView.Nodes[0].Text = ((Block)(((Button)sender).Parent.Parent)).BlockData.Text;

((Block)(((Button)sender).Parent.Parent)).MyTreeView.Text = ((Block)(((Button)sender).Parent.Parent)).BlockData.Text;

((Block)(((Button)sender).Parent.Parent)).MyTreeView.treeView.ExpandAll();

((Block)(((Button)sender).Parent.Parent)).MyTreeView.Show();

}

public void copyNodes(Tree source, Tree target)

{

TreeNode newTn = new TreeNode();

MenuItem add\_node1 = new MenuItem { Name = "Добавить ниже", Text = "Добавить ниже" };

MenuItem del\_node1 = new MenuItem { Name = "Удалить", Text = "Удалить" };

add\_node1.Click += target.AddNode\_Click;

del\_node1.Click += (s, ee) => { source.treeView.SelectedNode.Remove(); };

MenuItem add\_node2 = new MenuItem { Name = "Добавить ниже", Text = "Добавить ниже" };

MenuItem del\_node2 = new MenuItem { Name = "Удалить", Text = "Удалить" };

add\_node2.Click += target.AddNode\_Click;

del\_node2.Click += (s, ee) => { source.treeView.SelectedNode.Remove(); };

node\_menu.MenuItems.AddRange(new[] { add\_node1, del\_node1 });

node\_menu\_add.MenuItems.Add(add\_node2);

node\_menu\_del.MenuItems.Add(del\_node2);

target.treeView.Nodes.Clear();

foreach (TreeNode tn in source.treeView.Nodes)

{

copyChildNodes(tn, newTn);

target.treeView.Nodes.Add(newTn);

}

target.treeView.NodeMouseClick += MyTreeView.treeView\_NodeMouseClick;

}

private void copyChildNodes(TreeNode tn, TreeNode newTn)

{

newTn.Text = tn.Text;

if (tn.Level == 0)

newTn.ContextMenu = node\_menu\_add;

else if (tn.Level == 1)

newTn.ContextMenu = node\_menu;

else if (tn.Level == 2)

newTn.ContextMenu = node\_menu;

else if (tn.Level == 3)

newTn.ContextMenu = node\_menu\_del;

foreach (TreeNode tn\_child in tn.Nodes)

{

newTn.Text = tn.Text;

//newTn.Name = tn.Text;

TreeNode newTn\_child = new TreeNode();

newTn\_child.Text = tn\_child.Text;

copyChildNodes(tn\_child, newTn\_child);

newTn.Nodes.Add(newTn\_child);

}

}

public void AddPosition\_Click(object sender, EventArgs e)

{

Add(sender, 0, null, "position");

}

public void AddDepartment\_Click(object sender, EventArgs e)

{

Add(sender, 0, null, "department");

}

public void AddChild\_Click(object sender, EventArgs e)

{

add\_menu.Show((Button)sender, new Point(0, ((Button)sender).Height));

}

public void AddType(object obj)

{

if ((obj as Block).TypeOfBlock == "position")

{

((Block)obj).typePicture.BackgroundImage = new Bitmap(Properties.Resources.position\_ico);

((Block)obj).Extend.Visible = false;

}

else if ((obj as Block).TypeOfBlock == "department")

{

((Block)obj).typePicture.BackgroundImage = new Bitmap(Properties.Resources.department\_ico);

((Block)obj).Extend.Visible = true;

((Block)obj).Extend.Click += Extend\_Click;

}

}

public void Add(object par, int k, Block copy, string type)

{

Block newBlock = new Block();

Block papa;

Block deda;

Block pra;

tc = ((TabControl)Form.ActiveForm.Controls.Find("tabControl", false).FirstOrDefault());

newBlock.first = (Block)tc.SelectedTab.Controls.Find("block" + (tc.SelectedIndex + 1) + "\_1", false).FirstOrDefault();

Block left = newBlock.first;

Block right = newBlock.first;

newBlock.MyTreeView.Hide();

newBlock.TypeOfBlock = type;

AddType(newBlock);

if (k == 0) //добавление вызвано кнопкой Добавить

papa = ((Block)((par as MenuItem).GetContextMenu() as ContextMenu).SourceControl.Parent.Parent);

else //добавление вызвано кнопкой Вставить или Открыть

{

papa = par as Block;

copyNodes(copy.MyTreeView, newBlock.MyTreeView);

newBlock.BlockData.Text = copy.BlockData.Text;

}

if (papa.myParent != null)

deda = papa.myParent;

newBlock.myParent = papa;

newBlock.lvl = papa.lvl + 1;

newBlock.ContextMenu = Form.ActiveForm.Controls.Find("block1\_1", true).FirstOrDefault().ContextMenu;

newBlock.ContextMenu.MenuItems["Изменить тип блока"].Enabled = true;

newBlock.BlockData.ContextMenu = Form.ActiveForm.Controls.Find("block1\_1", true).FirstOrDefault().ContextMenu;

newBlock.number = papa.MyChilds.Count+1;

newBlock.Name = "block" + (tc.SelectedIndex + 1) + "\_" + newBlock.lvl + "\_" + papa.number + "(" + newBlock.number + ")";

// первый блок распологается непосредственно под родительским

if (papa.MyChilds.Count == 0)

{

newBlock.Location = new Point(papa.Location.X, papa.Location.Y + papa.Height + margin);

papa.MyChilds.Add(newBlock);

pra = newBlock;

do

{

pra.Branch.Add(newBlock);

pra = pra.myParent;

}

while (pra != null);

}

//все последующие - добавляются справа за последним элементом данного уровня

else

{

newBlock.Location = new Point(papa.MyChilds[papa.MyChilds.Count - 1].Location.X + papa.Width + margin, papa.MyChilds[papa.MyChilds.Count - 1].Location.Y);

papa.MyChilds.Add(newBlock);

pra = newBlock;

do

{

pra.Branch.Add(newBlock);

pra = pra.myParent;

}

while (pra != null);

//если добавление ОТ 1 уровня (на второй )

if (papa.lvl == 1)

{

////элементы размещаются по середине новое положение = середина\_формы/2 - (колво\_эл\_2ур\*ширина + отступ\*(колво\_эл\_2ур - 2 (по одному отступу у первого и последнего)))/2

//newloc = Form.ActiveForm.Size.Width / 2 - ((papa.MyChilds.Count) \* (papa.Width) + margin \* (papa.MyChilds.Count - 1)) / 2;

//foreach (Block ch in papa.MyChilds)

//{

// ch.Location = new Point(newloc, ch.Location.Y);

// newloc += papa.Width + margin;

//}

right = papa.MyChilds[papa.MyChilds.Count - 2];

while (right.MyChilds.Count > 0)

{

right = right.MyChilds[right.MyChilds.Count - 1];

}

newBlock.Location = new Point(right.Location.X + right.Width + margin, papa.MyChilds[papa.MyChilds.Count - 2].Location.Y);

}

//на последующие уровни - необходимо перемещать всех родителей и их родителей, находящихся справа...

else

{

//находим индекс родителя

pra = papa;

while (pra.myParent != null)

{

pra.Location = new Point(pra.MyChilds[0].Location.X + (pra.MyChilds[pra.MyChilds.Count - 1].Location.X - pra.MyChilds[0].Location.X) / 2, pra.Location.Y);

//берем всех правых братьев родителя

for (int idx = pra.number ; idx < pra.myParent.MyChilds.Count; idx++)

{

//и двигаем ВСЮ ИХ ВЕТКУ вправа

foreach (Block ch in pra.myParent.MyChilds[idx].Branch)

{

ch.Location = new Point(ch.Location.X + pra.Width + margin, ch.Location.Y);

}

}

//index = 0;

pra = pra.myParent;

}

}

}

//вычисляем координаты для блока1 (середину)

newBlock.first.Location = new Point(newBlock.first.MyChilds[0].Location.X + (newBlock.first.MyChilds[newBlock.first.MyChilds.Count - 1].Location.X - newBlock.first.MyChilds[0].Location.X) / 2, newBlock.first.Location.Y);

tc.SelectedTab.Controls.Add(newBlock);

//вызов метода отрисовки линий связи

drawLines();

}

//отрисовка линий связи

public void drawLines()

{

if (tc == null)

tc = ((TabControl)Form.ActiveForm.Controls.Find("tabControl", false).FirstOrDefault());

Pen pen = new Pen(Color.FromArgb(102,102,102), 3);

if (formGraphics != null)

formGraphics.Clear(Color.FromArgb(207, 231, 247));

formGraphics = tc.SelectedTab.CreateGraphics();

formGraphics.SmoothingMode = SmoothingMode.HighQuality;

int x1, y1, x2, y2, x3, y3, x4, y4;

foreach (Block b in this.first.Branch)

{

if (b.myParent != null)

{

x1 = 15 + b.myParent.Location.X + b.Width / 2;

y1 = b.myParent.Location.Y - 2 + b.Height;

x4 = 15 + b.Location.X + b.Width / 2;

y4 = b.Location.Y + 2;

if (b.Location.X == b.myParent.Location.X)

{

formGraphics.DrawLine(pen, x1, y1, x4, y4);

}

else

{

x2 = x1;

y2 = b.myParent.Location.Y - 2 + b.Height + 10;

x3 = 15 + b.Location.X + b.Width / 2;

y3 = y2;

formGraphics.DrawLine(pen, x1, y1, x2, y2);

formGraphics.DrawLine(pen, x2, y2, x3, y3);

formGraphics.DrawLine(pen, x3, y3, x4, y4);

}

}

}

//pen.Dispose();

//formGraphics.Dispose();

}

}

}

**Листинг файла Organizations.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace GOA

{

public partial class Organizations : Form

{

public Organizations()

{

InitializeComponent();

BackColor = Color.FromArgb(30, 111, 166);

this.MouseDown += Form\_MouseDown;

this.MouseMove += Form\_MouseMove;

this.MouseUp += Form\_MouseUp;

}

private bool isDragging = false;

private Point lastCursor;

private Point lastForm;

int pos\_x, pos\_y;

private void Form\_MouseDown(object sender, MouseEventArgs e)

{

isDragging = true;

lastCursor = Cursor.Position;

lastForm = this.Location;

}

private void Form\_MouseMove(object sender, MouseEventArgs e)

{

if (isDragging)

{

this.Location = Point.Add(lastForm, new Size(Point.Subtract(Cursor.Position, new Size(lastCursor))));

}

}

private void Form\_MouseUp(object sender, MouseEventArgs e)

{

isDragging = false;

}

private void CrossButton\_Click(object sender, EventArgs e)

{

this.Hide();

}

}

}

**Листинг файла Results.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace GOA

{

public partial class Results : Form

{

public Results()

{

InitializeComponent();

BackColor = Color.FromArgb(30, 111, 166);

dataGridView.BackgroundColor = Color.FromArgb(30, 111, 166);

this.MouseDown += Form\_MouseDown;

this.MouseMove += Form\_MouseMove;

this.MouseUp += Form\_MouseUp;

}

private bool isDragging = false;

private Point lastCursor;

private Point lastForm;

int pos\_x, pos\_y;

private void Results\_FormClosing(object sender, FormClosingEventArgs e)

{

e.Cancel = true;

this.Hide();

}

private void Form\_MouseDown(object sender, MouseEventArgs e)

{

isDragging = true;

lastCursor = Cursor.Position;

lastForm = this.Location;

}

private void Form\_MouseMove(object sender, MouseEventArgs e)

{

if (isDragging)

{

this.Location = Point.Add(lastForm, new Size(Point.Subtract(Cursor.Position, new Size(lastCursor))));

}

}

private void Form\_MouseUp(object sender, MouseEventArgs e)

{

isDragging = false;

}

private void CrossButton\_Click(object sender, EventArgs e)

{

this.Hide();

}

private void RollButton\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

private void SizeButton\_Click(object sender, EventArgs e)

{

if (Size.Width < 1000)

{

pos\_x = Location.X;

pos\_y = Location.Y;

Width = SystemInformation.VirtualScreen.Width;

Height = SystemInformation.VirtualScreen.Height;

Location = new Point(0, 0);

SizeButton.Image = Properties.Resources.size\_min;

}

else

{

Size = new Size(869, 207);

Location = new Point(pos\_x, pos\_y);

SizeButton.Image = Properties.Resources.size\_max;

}

}

}

}

**Листинг файла Tree.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace GOA

{

public partial class Tree : Form

{

public Tree()

{

InitializeComponent();

BackColor = Color.FromArgb(30, 111, 166);

treeView.BackColor = Color.FromArgb(30, 111, 166);

this.MouseDown += Form\_MouseDown;

this.MouseMove += Form\_MouseMove;

this.MouseUp += Form\_MouseUp;

}

ContextMenu node\_menu = new ContextMenu();

ContextMenu node\_menu\_add = new ContextMenu();

ContextMenu node\_menu\_del = new ContextMenu();

private bool isDragging = false;

private Point lastCursor;

private Point lastForm;

public void Tree\_Load(object sender, EventArgs e)

{

CreateNodeMenu(node\_menu, node\_menu\_add, node\_menu\_del);

treeView.Nodes[0].ContextMenu = node\_menu\_add;

treeView.NodeMouseClick += treeView\_NodeMouseClick;

}

private void Form\_MouseDown(object sender, MouseEventArgs e)

{

isDragging = true;

lastCursor = Cursor.Position;

lastForm = this.Location;

}

private void Form\_MouseMove(object sender, MouseEventArgs e)

{

if (isDragging)

{

this.Location = Point.Add(lastForm, new Size(Point.Subtract(Cursor.Position, new Size(lastCursor))));

}

}

private void Form\_MouseUp(object sender, MouseEventArgs e)

{

isDragging = false;

}

public void CreateNodeMenu(ContextMenu node\_menu, ContextMenu node\_menu\_add, ContextMenu node\_menu\_del)

{

MenuItem add\_node1 = new MenuItem { Name = "Добавить ниже", Text = "Добавить ниже" };

MenuItem del\_node1 = new MenuItem { Name = "Удалить", Text = "Удалить" };

add\_node1.Click += AddNode\_Click;

del\_node1.Click += (s, ee) => { treeView.SelectedNode.Remove(); };

MenuItem add\_node2 = new MenuItem { Name = "Добавить ниже", Text = "Добавить ниже" };

MenuItem del\_node2 = new MenuItem { Name = "Удалить", Text = "Удалить" };

add\_node2.Click += AddNode\_Click;

del\_node2.Click += (s, ee) => { treeView.SelectedNode.Remove(); };

node\_menu.MenuItems.AddRange(new[] { add\_node1, del\_node1 });

node\_menu\_add.MenuItems.Add(add\_node2);

node\_menu\_del.MenuItems.Add(del\_node2);

}

public void AddNode\_Click(object sender, EventArgs e)

{

TreeNode new\_node = new TreeNode();

if (treeView.SelectedNode.Level == 0)

{

new\_node.Text = "Новый процесс";

new\_node.ContextMenu = node\_menu;

treeView.SelectedNode.Nodes.Add(new\_node);

}

else if (treeView.SelectedNode.Level == 1)

{

new\_node.Text = "Новый подпроцесс";

new\_node.ContextMenu = node\_menu;

treeView.SelectedNode.Nodes.Add(new\_node);

}

else if (treeView.SelectedNode.Level == 2)

{

new\_node.Text = "Новая операция";

new\_node.ContextMenu = node\_menu\_del;

treeView.SelectedNode.Nodes.Add(new\_node);

}

treeView.SelectedNode.Expand();

}

public void treeView\_NodeMouseClick(object sender, TreeNodeMouseClickEventArgs e)

{

if (e.Button == MouseButtons.Right)

treeView.SelectedNode = e.Node;

}

public void Tree\_FormClosing(object sender, FormClosingEventArgs e)

{

e.Cancel = true;

this.Hide();

}

private void CrossButton\_Click(object sender, EventArgs e)

{

this.Hide();

}

private void RollButton\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

}

}