МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

КАФЕДРА: Экономической информатики

ФАКУЛЬТЕТ: Инженерно-экономический

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе по дисциплине «Объектно-ориентированное программирование и проектирование»

НА ТЕМУ:

«Автоматизированная система анализа использования оборудования»

ИСПОЛНИТЕЛЬ: (Wasja)
(ст.гр.)

РУКОВОДИТЕЛЬ: (Т.А. Огневая)

МИНСК 2009

Содержание

Введение

Широкое распространение компьютерной техники значительно упростило возможность проведения разнообразных бухгалтерских и экономических расчетов, что, в свою очередь, способствует более детальной и качественной проработке параметров хозяйственной деятельности, как отдельных предприятий, так и производственных отраслей экономики в целом. В то же время большое количество используемых бухгалтерами и экономистами методик расчетов обосновывает необходимость разработки конкретного программного обеспечения под каждую из задач.

Целью данной курсовой работы является разработка автоматизированной системы для проведения анализа использования оборудования. Поставленная задача является типичной задачей анализа хозяйственной деятельности.

Данный анализ позволяет оценить эффективность использования оборудования (как в пределах всего предприятия, так и по отдельным цехам), обозначить возможность улучшения экономических показателей предприятия, а также выявить резервы роста производительности. Это подтверждает актуальность решаемой задачи.

В то же время, актуальность разработки специализированного программного средства для решения поставленной задачи обосновывается сложностью проведения первоначального этапа анализа – сбора исходных данных, поскольку на данном этапе необходимо проанализировать информацию о работе всего оборудования предприятия (цеха) за достаточно продолжительный период времени (неделя, месяц, год), что весьма проблематично сделать вручную. При использовании же программного обеспечения этот процесс будет осуществляться автоматически.

Необходимость хранения больших объемов анализируемой информации в свою очередь обосновывает необходимость использования баз данных. Поскольку хранимая информация может быть использована в разнообразных расчетах, то для доступа к ней целесообразно организовать сетевой доступ, для чего в составе системы необходимо выделить сетевую и клиентскую часть.

Таким образом, для достижения цели курсовой работы необходимо решить следующие задачи:

- проанализировать предметную область и определить расчеты, которые должна выполнять система;
- разработать модели, позволяющие детализировать работу системы и уточнить требования к дальнейшей разработке;
- разработать базу данных для хранения анализируемой информации;
- разработать алгоритм работы системы в целом, после чего выделить в нем серверную и клиентскую части;
- разработать программную реализацию системы;
- осуществить проверку работоспособности системы.

1 Обзор предметной области

Для анализа работы оборудования применяется система показателей, характеризующих использование его численности, времени работы и мощности.

1) Анализ использования численности оборудования

Различают оборудование наличное и установленное (сданное в эксплуатацию), фактически используемое в производстве, находящееся в ремонте и на модернизации и резервное. К наличному оборудованию относится все имеющееся оборудование независимо от того, где оно находится (в цехах, на складе) и в каком состоянии. К установленному относится смонтированное и подготовленное к работе оборудование, находящееся в цехах, причем часть установленного оборудования может находиться в резерве, на консервации, в плановом ремонте, модернизации. К действующему оборудованию относится все фактически работающее в отчетном периоде (независимо от времени его работы).

Оптимальной считается ситуация, при которой наличное, установленное и действующее оборудование по величине примерно одинаково. В этом случае оборудование используется с высокой степенью загрузки, а производственная программа соответствует производственной мощности.

Для оценки использования имеющееся в наличии оборудование сопоставляют с установленным, а установленное – с действующим, так как не все имеющееся оборудование установлено и не все установленное оборудование эксплуатируется. Кроме того, для характеристики степени привлечения оборудования в производство рассчитывают следующие показатели:

– коэффициент использования парка наличного оборудования:

$$K_{ДН} = \frac{\text{Количество действующего оборудования}}{\text{Количество наличного оборудования}}; \quad (1.1)$$

– коэффициент использования парка установленного оборудования:

$$K_{ДУ} = \frac{\text{Количество действующего оборудования}}{\text{Количество установленного оборудования}}; \quad (1.2)$$

– коэффициент использования оборудования сданного в эксплуатацию:

$$K_{УН} = \frac{\text{Количество установленного оборудования}}{\text{Количество наличного оборудования}} \quad (1.3)$$

Задача наиболее полного использования оборудования заключается в том, чтобы свести к минимуму количество неустановленного бездействующего оборудования. С этой целью необходимо сближение величин, характеризующих количество наличного, установленного и работающего оборудования.

Повышение эффективности использования действующего оборудования обеспечивается двумя путями: экстенсивным (по времени) и интенсивным (по производительности).

2) Анализ времени работы оборудования

Анализ экстенсивного использования оборудования начинается с анализа баланса времени его работы. Он включает:

- календарный фонд времени – максимально возможное время работы оборудования (количество календарных дней в отчетном периоде умножается на 24 часа и на количество единиц установленного оборудования);
- режимный фонд времени (количество единиц установленного оборудования умножается на количество рабочих дней отчетного периода и на количество часов ежедневной работы с учетом коэффициента сменности);
- эффективный (возможный, плановый) фонд времени (разница между режимным фондом и временем планового ремонта и модернизации);
- фактический фонд отработанного времени (по данным учета, отличается от планового фонда на время внеплановых простоев).

Сравнение фактического и планового календарных фондов времени позволяет установить степень выполнения плана по вводу оборудования в эксплуатацию по количеству и срокам; календарного и режимного – по возможности лучшего использования оборудования за счет повышения коэффициента сменности, а режимного и планового – резервы времени за счет сокращения затрат времени на ремонт. Анализ осуществляется по предприятию, цехам, участкам, конкретным видам машин.

Показателем, характеризующим использование оборудование во времени, служат коэффициенты экстенсивности, которые определяют как отношение фактически отработанного времени к фондам времени:

- коэффициент использования календарного фонда времени:

$$K_{КФ} = \frac{\text{Фактический фонд времени}}{\text{Календарный фонд времени}}; \quad (1.4)$$

- коэффициент использования режимного фонда времени:

$$K_{РФ} = \frac{\text{Фактический фонд времени}}{\text{Режимный фонд времени}}; \quad (1.5)$$

- коэффициент использования планового фонда времени:

$$K_{ПФ} = \frac{\text{Фактический фонд времени}}{\text{Эффективный фонд времени}} \quad (1.6)$$

Общий коэффициент экстенсивности может быть рассчитан как среднеарифметическое всех коэффициентов использования оборудования.

Снижение показателей фондов времени работы оборудования и увеличение удельного веса простоев свидетельствует о том, что имеются резервы роста производства (наличие упущенных возможностей). Необходимо усилить контроль за работой оборудования, выяснить причины снижения указанных показателей

(неисправности и внеплановый ремонт; отклонение электроэнергии; наладка и переналадка оборудования в связи с изменением ассортимента).

3) Анализ производственной мощности оборудования

Интенсивное использование основных фондов характеризуется как показателями выпуска за один станко-час, так и натуральными и условно-натуральными показателями, принятыми в той или иной отрасли.

Показатели интенсивного использования оборудования определяются по плану, по факту определяется абсолютное отклонение. Абсолютное отклонение является объектом анализа. Определяются причины изменения показателей, выявляются резервы роста объема выпуска. Снижение фактических показателей по сравнению с расчетными плановыми, исходя из рациональной загрузки оборудования, свидетельствует о нерациональном использовании производственной мощности, об упущении реальных возможностей увеличения выпуска продукции.

Показателем интенсивности работы оборудования является коэффициент интенсивной загрузки:

$$K_{инт} = \frac{\text{Средняя часовая выработка единицы оборудования фактическая}}{\text{Средняя часовая выработка единицы оборудования плановая}} \qquad (1.7)$$

Коэффициент интенсивной нагрузки оборудования может быть больше единицы, равным единице и меньшим единицы.

Обобщающим показателем, характеризующим использование оборудования по времени и по производительности, является коэффициент интегральной нагрузки:

$$K_{интегр} = K_{экст} \cdot K_{инт} \qquad (1.8)$$

Далее в процессе анализа изучается динамика рассчитанной системы показателей, выполнение плана, определяются причины изменений и резервы упущенных возможностей. Для этого по группам однородного оборудования рассчитывается изменение объема выпуска продукции за счет его количества, экстенсивности и интенсивности использования. При этом объем производства рассчитывается как произведение количества работающего оборудования на время работы единицы оборудования и на среднечасовую выработку единицы оборудования. Расчет влияния этих факторов (факторный анализ) производится способами цепной подстановки или абсолютных и относительных разниц.

Следующим этапом является определение резервов увеличение выпуска продукции:

- Резервы увеличения выпуска продукции за счет ввода в действие нового оборудования определяются умножением его дополнительного количества на фактическую величину средней выработки в отчетном периоде или на фактическую величину всех факторов, которые формируют ее уровень.
- Сокращение целодневных простоев оборудования приводит к увеличению среднего количества отработанных дней каждой его единицей за период. Этот прирост необходимо умножить на возможное (прогнозируемое) ко-

личество единиц оборудования и фактическую среднедневную выработку единицы в отчетном периоде.

- Чтобы подсчитать резерв увеличения выпуска продукции за счет повышения коэффициента сменности в результате лучшей организации производства, необходимо планируемый прирост последнего умножить на возможное количество дней работы всего парка оборудования и на фактическую сменную выработку.
- За счет сокращения внутрисменных простоев увеличивается средняя продолжительность смены, а, следовательно, и выпуск продукции. Для определения величины этого резерва следует планируемый прирост средней продолжительности смены умножить на фактический уровень среднечасовой выработки оборудования и на возможное количество машино-смен всем его парком (произведение возможного количества оборудования, возможного количества отработанных дней единицей оборудования и возможного коэффициента сменности).
- Для определения резерва увеличения выпуска продукции за счет повышения среднечасовой выработки оборудования необходимо сначала выявить возможности роста последней за счет его модернизации, более интенсивного использования, внедрения мероприятий НТП и т.д. Затем выявленный резерв повышения среднечасовой выработки надо умножить на планируемое количество часов работы оборудования (произведение возможного количества единиц, количества дней работы, коэффициента сменности, продолжительности смены).

Следовательно, разрабатываемая автоматизированная система должна:

- осуществлять доступ к данным о подразделениях предприятия (цехах), о результатах работы предприятия, а также об использовании оборудования, хранимым в БД на сервере;
- обеспечивать возможность задания условий для выполнения анализа, основными из которых являются даты анализируемых периодов;
- осуществлять автоматическое вычисление параметров для анализа по обоим из периодов, основываясь на данных об использовании оборудования и датах;
- осуществлять выдачу результатов в виде отчета в доступном для восприятия виде.

2 Постановка задачи

В соответствии с заданием на курсовую работу:

- необходимо разработать автоматизированную систему анализа использования оборудования;
- для разработки использовать систему Visual Studio, язык C++ и библиотеку MFC;
- для реализации графического интерфейса использовать элементы управления CComboBox, CStatic, CButton, ClistCtrl и Cedit;
- в основе системы должна лежать клиент-серверная архитектура;
- соединения между клиентами и сервером должно осуществляться через протокол TCP/IP;
- сервер программы должен базироваться на алгоритме псевдопараллельной обработки запросов;
- для хранения данных необходимо использовать СУБД Microsoft Access;
- доступ к данным БД должен осуществляться с использование интерфейса OLE DB;
- БД должна генерироваться SQL-скриптом под пользователем;
- необходимо выполнить моделирование процессов предметной области в соответствии со стандартом IDEF0 и информационное моделирование в соответствии со стандартом IDEF1x;
- результаты каждого конкретного проведённого анализа должны быть представлены пользователю в виде DOC-документа.

Основываясь на вышеприведенном обзоре предметной области и задании на проектирование можно уточнить задачи, которые необходимо решить в ходе выполнения курсовой работы:

- провести обзор информационных источников по используемым технологиям, средствам и системам, а именно: языку C++, системе Visual Studio, библиотеке MFC, протоколу TPC/IP и его реализации в семействе операционных систем Windows, клиент-серверным архитектурам, СУБД Microsoft Access, стандартам IDEF0 и IDEF1x;
- разработать методы и модели представления предметной области, на основе которых разработать алгоритмы работы системы;
- разработать информационную модель системы, но основе которой разработать базу данных для хранения анализируемых данных об использовании оборудования;
- выделить в алгоритме работы системы серверную и клиентскую части, после чего разработать программные реализации соответствующих частей системы;
- заполнить БД необходимыми данными об использовании оборудования и осуществить проверку работоспособности системы;
- разработать руководство пользователя автоматизированной системы;
- оформить пояснительную записку.

3 Разработка методов и моделей предметной области

Цель построения модели предметной области – специфицирование операций и действий, выполняемых в данной области, и взаимосвязей между ними. При адекватном построении такая модель обеспечивает полное представление о функционировании исследуемого процесса и обо всех потоках информации и материалов, имеющихся в нем. Одной из наиболее широко используемых методологий моделирования систем является стандарт IDEF0.

В основе IDEF0-моделирования лежит понятие блока, который реализует некую конкретную функцию, т.е. выполняет управляемое действие над входными данными, результатом которого являются выходные данные, при этом используется некий механизм. Четыре стороны блока имеют разное назначение: слева отображаются входные данные; справа – выходные данные; сверху – управление; снизу – механизм. Взаимодействие между функциями отображается в виде стрелок.

IDEF0 основана на трех принципах моделирования:

1) принцип функциональной декомпозиции – любая функция может быть разбита на более простые функции;

2) принцип ограничения сложности – количество блоков на диаграмме должно быть не менее трех и не более шести;

3) принцип контекста – моделирование делового процесса начинается с построения контекстной диаграммы, состоящей из одного блока (главная функция моделируемой системы), определяющего границы системы.

Для разработки моделей предметной области была использована система PBwin фирмы Computer Associates. Основной функцией BPwin является рисование диаграмм, представляющих собой визуальное представление отдельных компонентов моделируемой предметной области различных уровней детализации, а также проверка целостности и согласованности иерархической модели.

Как было сказано выше, процесс моделирования начинается с определения контекста, для чего необходимо определить субъект моделирования, цели и точки зрения на модель. В данной курсовой работе субъектом моделирования является процесс выполнения анализа использования оборудования, целью – моделирование этого процесса, точка зрения на процесс моделирования – экономист предприятия. Контекстная диаграмма будет иметь:

- Два входа: «Данные об использовании оборудования» и «Время». На первый вход поступает исходная информация для проведения анализа, а присутствие второго входа (туннельный со скрытым приемником) объясняется наличием затрат времени на проведение анализа.
- Две линии управления: «Параметры проведения анализа» и «Порядок проведения анализа». Первая линия представляет собой задание на анализ, которое исходит от руководства предприятия, а вторая (туннельная со скрытым приемником) – формализованные принципы проведения анализа.
- Два механизма: «Экономист» и «Автоматизированная система». Первый механизм – это сотрудник предприятия, непосредственно выполняющий анализ, а второй – система, разработка которой, является целью данной курсовой работы.

– Один выход: «Отчет о проведенном анализе». На выход системы поступают численные характеристики, полученные в результате проведения анализа, на основе которых экономист может сделать выводы об эффективности использования оборудования на предприятии.

Контекстная диаграмма модели приведена на рисунке А.1 в приложении А. Функцию, выполняемую системой в целом, можно разбить на 4 подфункции (рисунок А.2):

– «Задание параметров проведения анализа» – во время выполнения данной функции экономист на основании параметров проведения анализа и данных об использовании оборудования формулирует в понятной системе форме параметры анализа. После этого, на основании сформулированного и исходных данных система производит вычисление параметров текущего и предыдущего периодов, необходимых для дальнейшего анализа. Диаграмма для данной функции приведена на рисунке А.3.
– «Выполнение расчета системы показателей» – данная функция осуществляет вычисление показателей использования оборудования в соответствии с формулами (1.1)-(1.8). Диаграмма для данной функции приведена на рисунке А.4, она включает 3 подфункции: «Выполнение анализа использования численности оборудования» (рисунок А.7), «Выполнение анализа времени работы оборудования» (рисунок А.8) и «Выполнение анализа производственной мощности оборудования» (рисунок А.9). Назначение перечисленных подфункций соответствует этапам анализа, рассмотренным в разделе 1.
– «Выполнение анализа динамики системы показателей» – функция заключается в проведении факторного анализа влияния изменения трех параметров (количества действующего оборудования, экстенсивной нагрузки и интенсивной нагрузки) и генерации соответствующего отчета, что, соответственно, составляет 4 подфункции. Диаграмма для данной функции приведена на рисунке А.5.
– «Определение резервов увеличения выпуска продукции» – функция состоит из пяти подфункций, каждая из которых занимается анализом влияния одного отдельного параметра на увеличение выпуска продукции (ввод нового оборудования, сокращение целодневных простоев, сокращение внутрисменных простоев, повышение коэффициента сменности, повышение среднечасовой выработки). Диаграмма для данной функции приведена на рисунке А.6.

Полученные модели предметной области позволяют осуществить дальнейшее проектирование системы.

4 Разработка информационной модели системы

Для разработки информационной модели стандарта IDEF1x использовалась специализированная система ERwin фирмы Computer Associates, предназначенная для разработки структуры базы данных. ERwin сочетает графический интерфейс Windows, редакторы для создания логического и физического описания модели данных и прозрачную поддержку ведущих реляционных СУБД и настольных баз данных.

В ERwin существуют два уровня представления и моделирования – логический и физический. Логический уровень означает прямое отображение фактов реальной жизни, в то время как целевая СУБД, имена объектов и тины данных, индексы составляют физический уровень модели.

Диаграммы ERwin строятся из трех основных элементов – сущностей, атрибутов и связей. Сущность – логическое понятие, соответствующее таблице в реальных СУБД. Атрибуты являются составными частями сущностей и соответствуют столбцам (полям) таблицы. Связь – это функциональная зависимость между двумя сущностями, на физическом уровне связь соответствует внешнему ключу.

Процесс построения информационной модели можно условно разбить на следующие этапы:

- определение сущностей и связей между сущностями;
- определение атрибутов сущностей и задание первичных и альтернативных ключей;
- приведение модели к требуемому уровню нормальной формы;
- переход к физическому описанию модели и генерация базы данных.

Исходные параметры для анализа использования оборудования можно определить на основании функциональной модели. К ним относятся следующие параметры (по текущему и предыдущему периоду отдельно):

- количество наличного оборудования;
- количество установленного оборудования;
- количество действующего оборудования;
- календарный фонд времени;
- режимный фонд времени;
- плановый фонд времени;
- фактический фонд времени;
- объем производства товаров;
- количество дней в периоде;
- количество рабочих дней;
- количество смен (за весь период).

Однако, поскольку перечисленные параметры могут изменяться несколько раз в день, то имеет смысл осуществлять их вычисление непосредственно перед проведением анализа на сервере, что позволяет использовать всегда актуальные данные, а также снизить нагрузку на сеть.

В наиболее общем случае информацию об использовании оборудования можно хранить в однотабличной базе данных со следующими атрибутами:

- «Порядковый номер операции»;

- «Инвентарный номер оборудования»;
- «Наименование оборудования»;
- «Номер смены»;
- «Название цеха»;
- «Тип использования» - эксплуатация, плановый или внеплановый ремонт, не использовалось;
- «Начало использования»;
- «Окончание использования»;
- «Количество произведенного товара»;
- «Стоимость единицы товара».

Однако информация, представленная в этой таблице является избыточной, поэтому ее необходимо нормализовать и привести к третьей нормальной форме, для чего ее целесообразно разбить на следующие сущности:

1) Сущность «Цеха» – предназначена для хранения информации о структурных подразделениях предприятия. Атрибуты:

- Атрибут «КодЦеха»: первичный ключ, целое, обязательный атрибут. Предназначен для хранения номера структурного подразделения.
- Атрибут «Наименование»: строка (20 символов), обязательный атрибут. Предназначен для хранения названия структурного подразделения.

2) Сущность «Оборудование» – предназначена для хранения информации обо всем оборудовании, принадлежащем предприятию. Атрибуты:

- Атрибут «ИнвентарныйНомер»: первичный ключ, целое, обязательный атрибут. Предназначен для хранения уникального номера оборудования.
- Атрибут «Наименование»: строка (40 символов), обязательный атрибут. Предназначен для хранения названия оборудования.
- Атрибут «ДатаВвода»: дата, обязательный атрибут. Предназначен для хранения даты покупки оборудования.
- Атрибут «ДатаВывода»: дата. Предназначен для хранения даты продажи оборудования.
- Атрибут «КодЦеха»: целое, обязательный атрибут. Предназначен для определения структурного подразделения, к которому относиться оборудование.

3) Сущность «Результаты» – предназначена для хранения результатов производства каждой из единиц оборудования посменно. Атрибуты:

- Атрибут «НомерПартии»: первичный ключ, целое, обязательный атрибут. Предназначен для хранения номера партии.
- Атрибут «Количество»: целое, обязательный атрибут. Предназначен для хранения размера партии продукции (в натуральном измерении – шт., кг., л.).
- Атрибут «СтоимостьЕд»: денежное значение, обязательный атрибут. Предназначен для хранения стоимости единицы товара в натуральном исчислении, совпадающим с используемым в атрибуте «Количество».
- Атрибут «Дата»: дата, обязательный атрибут. Предназначен для хранения даты выпуска партии.
- Атрибут «ИнвентарныйНомер»: целое, обязательный атрибут. Предназна-

чен для определения оборудования, выпустившего партию.

4) Сущность «Эксплуатация» – предназначена для учета времени работы каждой единицы оборудования, принадлежащего предприятию. Атрибуты:

- Атрибут «ПорядковыйНомер»: первичный ключ, целое, обязательный атрибут. Предназначен для хранения уникального номера записи.
- Атрибут «ИнвентарныйНомер»: целое, обязательный атрибут. Предназначен для определения конкретной единицы оборудования, к которому относиться запись.
- Атрибут «ДатаНачалаРаботы»: дата, обязательный атрибут. Предназначен для хранения информации о начале работы оборудования.
- Атрибут «ДатаОкончанияРаботы»: дата. Предназначен для хранения информации об окончании работы оборудования.

5) Сущность «Ремонт» – предназначена для учета времени ремонта каждой единицы ремонтируемого (модифицируемого) оборудования и типа осуществленного ремонта. Атрибуты:

- Атрибут «ПорядковыйНомер»: первичный ключ, целое, обязательный атрибут. Предназначен для хранения уникального номера записи.
- Атрибут «ИнвентарныйНомер»: целое, обязательный атрибут. Предназначен для определения конкретной единицы оборудования, к которому относиться запись.
- Атрибут «ДатаНачалаРемонта»: дата, обязательный атрибут. Предназначен для хранения информации о начале ремонта оборудования.
- Атрибут «ДатаОкончанияРемонта»: дата. Предназначен для хранения информации об окончании ремонта оборудования.
- Атрибут «ТипРемонта»: булево число, обязательный атрибут. Предназначен для хранения информации о типе ремонта – плановый (TRUE) или внеплановый (FALSE).

Итоговая информационная модель приведена в приложении Б.

Используя разработанную модель можно определить принцип определения исходных параметров для анализа использования оборудования:

- Количество наличного оборудования – определиться как количество записей в таблице «Оборудование», у которых значение поля «ДатаВвода» меньше даты начала анализируемого периода, а поля «ДатаВывода» - больше или отсутствует.
- Количество установленного оборудования – количество уникальных значений полей «ИнвентарныйНомер», встречающихся в таблицах «Эксплуатация» и «Ремонт», для которых значения дат попадают в анализируемые периоды.
- Количество действующего оборудования – количество уникальных значений поля «ИнвентарныйНомер», встречающихся в таблице «Эксплуатация», для которых значения дат попадают в анализируемые периоды.
- Календарный фонд времени – произведение количества дней в периоде на 24 и количество установленного оборудования.
- Режимный фонд времени – сумма интервалов времени всех записей таблиц «Эксплуатация» и «Ремонт», попадающих в анализируемые периоды.
- Плановый фонд времени – разница между режимным фондом и суммой

интервалов времени записей таблицы «Ремонт», у которых поле «ТипРемонта» соответствует TRUE (плановый ремонт), а даты попадают в анализируемые периоды.

- Фактический фонд времени – сумма интервалов времени записей таблицы «Эксплуатация», даты которых попадают в анализируемые периоды.
- Объем производства товаров – сумма произведений полей «Количество» и «СтоимостьЕд» записей таблицы «Результаты», даты которых попадают в анализируемые периоды.
- Количество рабочих дней – количество уникальных значений поля «Дата» таблицы «Результаты», попадающих в анализируемые периоды.
- Количество смен (за весь период) – количество значений поля «Дата» таблицы «Результаты», попадающих в анализируемые периоды.

5 Обоснование технических и программных средств реализации системы

Для разработки автоматизированной системы целесообразно применить пакет Visual Studio 2003 .NET, являющийся дальнейшим развитием широко распространенной компилятора Visual C++ 6.0, выпущенного в 1998 году и морально устаревшему к текущему времени.

Visual Studio позволяет создавать приложения для целого ряда платформ: Win32, MFC, COM, ActiveX, Active Template Library (ATL), Java, DirectX и Web-приложения. Visual C++ .NET, являющийся реализацией C++ в Visual Studio .NET, позволяет C++ -программистам писать устойчивый, безопасный и эффективный код. Многие из усовершенствований были представлены в версии 7.1 компилятора. Microsoft добавила в Visual C++ .NET 17 новых переключателей компилятора и 12 новых переключателей компоновщика, что даже без учета новых возможностей IDE и .NET является для некоторых разработчиков веским основанием для перехода на этот продукт.

Visual C/C++ .NET позволяет разработчикам применять привычные им библиотеки классов. К этим библиотекам относится MFC 7, поддерживающая Windows XP, Windows 2000 и ATL 7. Среди новшеств этой библиотеки — диалоговые окна DHTML, позволяющие создавать удобные диалоговые окна для MFC-приложений. И, наконец, ATL и MFC теперь совместно используют ряд общих классов, например CString, что позволяет ATL-программистам задействовать функциональность MFC, не загружая всю библиотеку MFC. Новая особенность ATL — появление сервера ATL ATL-сервер служит для создания высокопроизводительных Web-приложений на основе ISAPI, а также для создания Web-сервисов XML ATL-сервер можно применять для создания решений с неуправляемыми Web-сервисами XML или для интеграции существующего ATL-кода в новые .NET-решения. К прочим библиотекам классов, которых коснулось обновление, относятся Standard Template Library (STL) и C-библиотека периода выполнения (CRT).

Разработчики Visual C++ обеспечили почти полное соответствие стандартам ANSI/ ISO C++. Реализованный в Visual C++ .NET 2003 компилятор C++ совместим с ними более чем на 98%.

Таким образом, поскольку разрабатываемая система не предполагает каких-либо значительных нагрузок на систему, то для ее разработки достаточно системы, позволяющей работать с пакетом Visual Studio 2003 .NET.

6 Разработка алгоритмов работы системы

Существует несколько способов обработки запросов в клиент-серверной архитектуре, но большинство из них можно свести к последовательной или параллельной обработке. При последовательной обработке сервер обрабатывает запросы от клиентов в порядке их поступления, т.е. обработка каждого последующего запроса осуществляется только после обработки всех предыдущих. Однако при таком подходе клиенты могут подолгу простаивать после установления соединения, ожидая обслуживания.

Поэтому обычно применяется схема псевдопараллельной обработки запросов, при которой после установления соединения, сервер порождает новый сокет, которому и поручает дальнейшую работу с клиентом, после чего уходит на ожидание нового соединения. Таким образом, схему работы сервера, реализующего псевдопараллельную обработку, можно разделить на две части. Первая из них обеспечивает работу пассивного сокета, ожидающего соединение, а вторая обеспечивает непосредственный обмен данными с каждым из клиентов. Описанная схема представлена на рисунке 6.1.

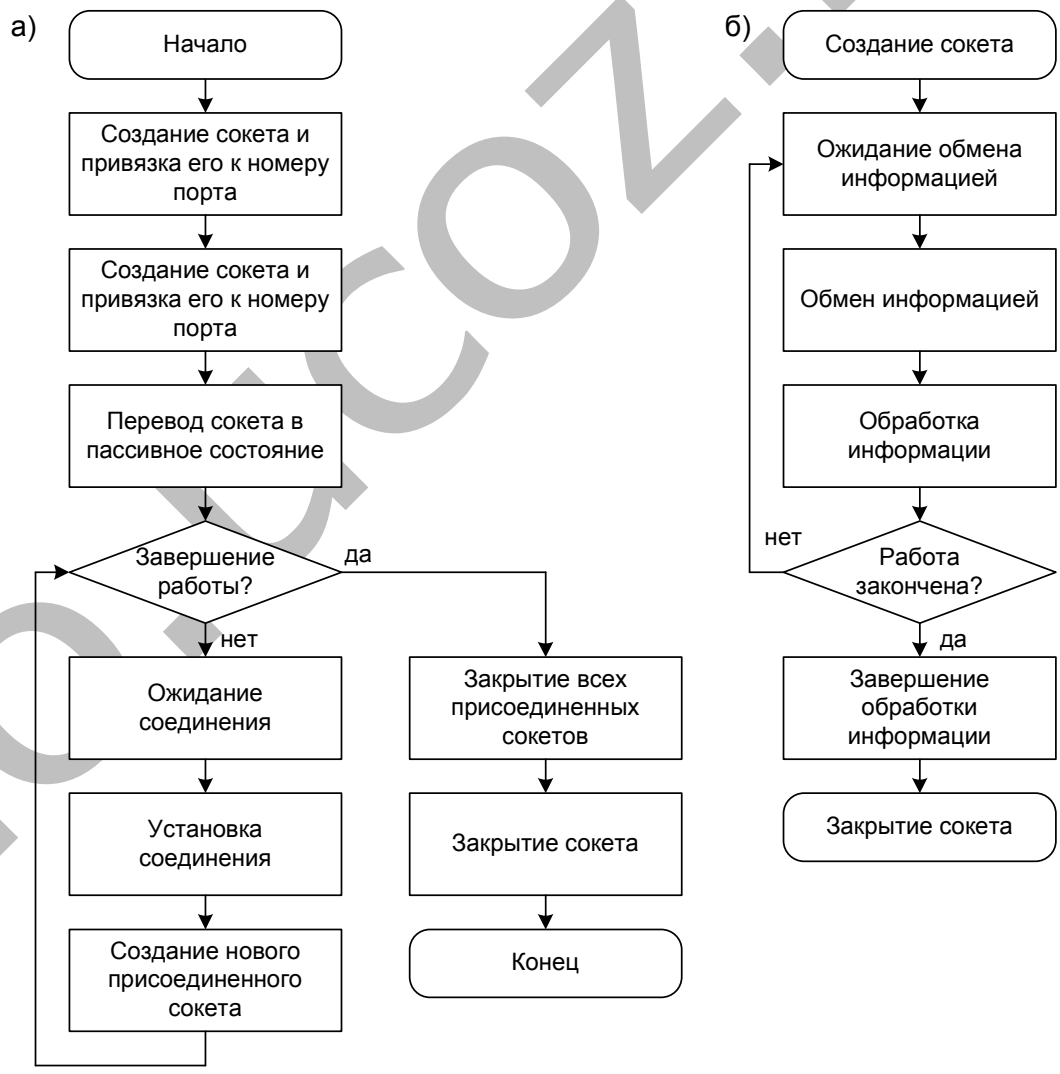

Рисунок 6.1 – Схема работы сервера с псевдопараллельной обработкой запросов: а) работа с пассивным сокетом; б) обмен данными с клиентом

Работа клиента не зависит от способа обработки запросов на сервере и реализуется с помощью использования одного сокета, который перед началом обмена данных соединяется с сервером. Для присоединения обязательно необходимо указать сетевой адрес сервера и порт, который был зарезервирован под сервер. Схема работы клиента в общем виде представлена на рисунке 6.2.

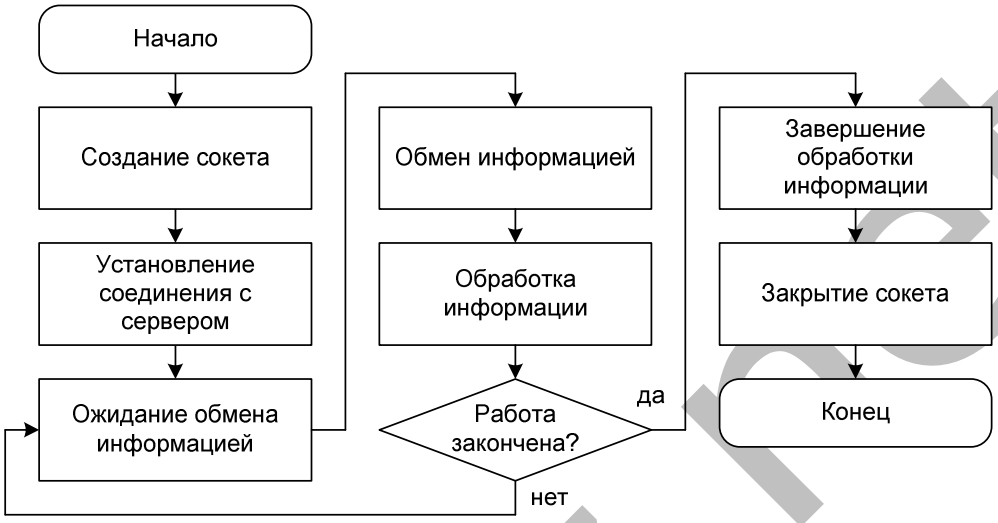

Рисунок 6.2 – Схема работы клиента

В Visual C++ сервер с псевдопараллельной обработкой запросов можно реализовать, используя класс CAsyncSocket библиотеки MFC. Данный класс является надстройкой над стандартным сокетом операционной системы Windows, в основу его работы положен асинхронный обмен сообщениями, благодаря чему легко обеспечивается параллельная работа с произвольным числом клиентов.

Необходимые для реализации сервера методы класса CAsyncSocket:

- Create(INT nSocketPort = 0, ...) – создание сокета и привязка его к порту, номер которого задается параметром nSocketPort. Если в качестве номера порта передается 0, то операционная система автоматически назначает сокету произвольный свободный порт.
- Listen(...) – перевод сокета в пассивное состояние для ожидания входящих соединений.
- Accept(CAsyncSocket& rConnectedSocket, ...) – присоединение сокета rConnectedSocket для обработки входящего соединения.
- Close( ) – закрытие сокета.
- Receive(void* lpBuf, int nBufLen, ...) – считывание полученных данных (nBufLen байт) в буфер lpBuf.
- Send(const void* lpBuf, int nBufLen, ...) – посылка данных (nBufLen байт) из буфера lpBuf.
- OnAccept(...) – событие, сообщающее о наличии клиента, установившего соединение и готового к обмену.
- OnReceive(...) – событие, сообщающее о том, что в буфере приема присутствуют данные, пришедшие от клиента.
- OnClose(...) – событие, сообщающее о том, что клиент закрыл соединение.

Для реализации клиента так же можно использовать те же методы класса CAsyncSocket, что и при реализации сервера, а также метод OnConnect(…), который сообщает, что соединение с сервером установлено.

Рассмотренные выше схемы представлены в общем виде, приемлемом для реализации любых клиент-серверных систем. Различие между системами с точки зрения сетевой реализации заключается в протоколах обмена данными между клиентами и сервером. Поскольку, как было описано в разделе 4, основные операции выполняются на сервере, то необходимость в обмене большими объемами данных отсутствует. Поэтому в основу протокола обмена можно положить обмен строковыми сообщениями, а критерием окончания строки принять символ окончания строки. Тогда протокол обмена, с учетом необходимых для анализа данных, примет вид, представленный на рисунке 6.3.

Данный протокол можно реализовать с помощью последовательной смены внутренних состояний, как на сервере, так и на клиенте. Для сервера потребуется 8 состояний, которые должен выполнять каждый из присоединенных сокетов:

- Состояние 1: начальное состояние, в котором пребывает сервер после установления связи с клиентом (блок 01 протокола). После отправки клиенту количества структурных подразделений и их названий (блоки 02 и 03) сервер переходит в состояние 2.
- Состояние 2: ожидание приема номера подразделения, по которому будет осуществляться анализ (0 – анализ по всем подразделениям предприятия), после приема переход в состояние 3.
- Состояние 3: определение граничных даты для анализа (по номеру подразделения), после чего сервер отправляет их клиенту (блоки 05 и 06) и переходит в состояние 4.
- Состояние 4: ожидание приема начальной даты текущего периода, после приема переход в состояние 5.
- Состояние 5: ожидание приема конечной даты текущего периода, после приема переход в состояние 6.
- Состояние 6: ожидание приема начальной даты предыдущего периода, после приема переход в состояние 7.
- Состояние 7: ожидание приема конечной даты предыдущего периода. После приема сервер выполняет вычисление исходных параметров для анализа, которые затем отправляет клиенту (блоки 11-30 протокола). После окончания передачи осуществляется переход в состояние 8.
- Состояние 8: ожидание завершения соединения, после чего сокет закрывается.

Сокет клиента в свою очередь должен пройти следующие состояния:

- Состояние 1: начальное состояние, в котором клиент ожидает приема количества подразделений предприятия, после приема переход в состояние 2.
- Состояние 2: считывание названий структурных подразделений предприятия. После окончания считывания осуществляется запрос у пользователя номера анализируемого подразделения, полученный номер отправляется серверу (блок 04) и выполняется переход на состояние 3.
- Состояние 3: ожидание приема начальной граничной даты, после приема переход в состояние 4.

– Состояние 4: ожидание приема конечной граничной даты, после приема осуществляется запрос у пользователя значений дат анализируемых периодов. Введенные даты отправляются на сервер (блоки 07-10), после чего выполняется переход к состоянию 5.
– Состояния 5-23: ожидание и последующий прием исходных параметров для анализа, сопровождающийся последовательным наращиванием номера состояния. После приема всех параметров выполняется непосредственный анализ, генерируется отчет и сокет закрывается.

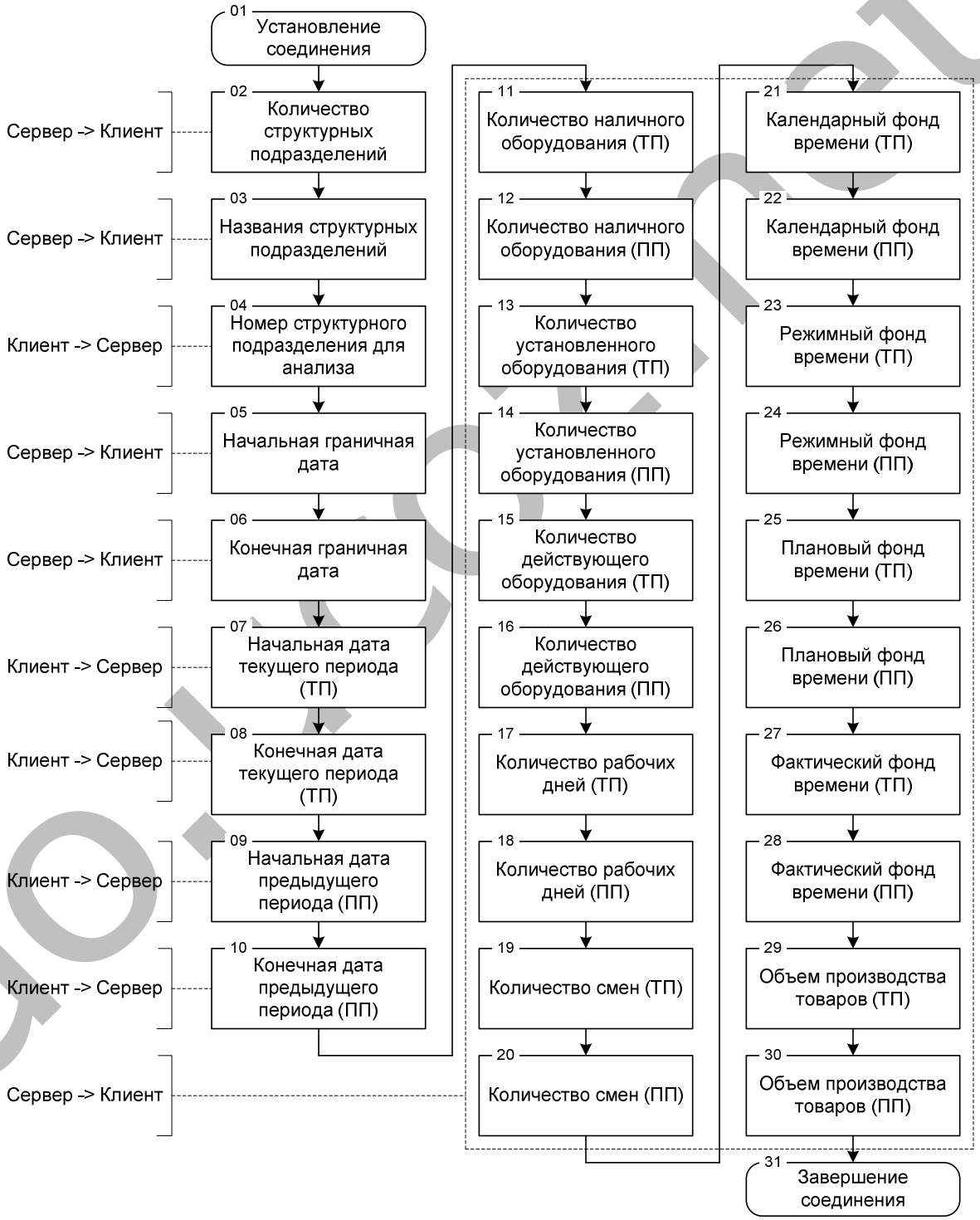

Рисунок 6.3 – Протокол обмена между клиентом и сервером

Поддержку работы с базой данных Access, связь с которой должна осуществляться с использование интерфейса OLE DB, уместно реализовать с помощью стандартных шаблонов OLE DB, входящих в библиотеку MFC. В качестве провайдера уместно использовать «Microsoft Jet 4.0 OLE DB Provider» позволяющий работать напрямую с MDB-файлами без необходимости установки Microsoft Access. При создании проекта мастер Visual Studio автоматически генерирует класс, где с помощью макроса db_source задается источник данных, с помощью макроса db_table – таблица или запрос, а макрос db_column задает связь между столбцом таблицы и переменной, через которую можно будет обращаться к соответствующим данным этого столбца. Данный класс также наследует методы, позволяющие перемещаться по строкам таблицы, устанавливать и разрывать связь с источником данных:

- Для установления связи с источником данных можно использовать метод OpenFromInitializationString атрибута m_source, которому в качестве параметра передается строка соединения.
- Для создания сессии связи с источником используется метод Open атрибута m_session, которому в качестве параметра передается m_source. Для удаления сессии можно использовать метод Close атрибута m_session.
- Для непосредственного установления связи между классом и таблицей (или запросом) используется метод Open класса, которому передается атрибут m_session и строка, в которой содержится запрос на выборку или имя таблицы. Для разрыва связи используется метод Close класса.
- Для связи класса с таблицей, заданной по умолчанию с помощью макроса, вместо использования трех вышеперечисленных методов, используется метод класса OpenAll.
- Для разрыва всех связей используется метод CloseAll.
- Для перемещения на первую строку таблицы используется метод MoveFirst, на следующую строку – метод MoveNext, на предыдущую строку – метод MovePrev.

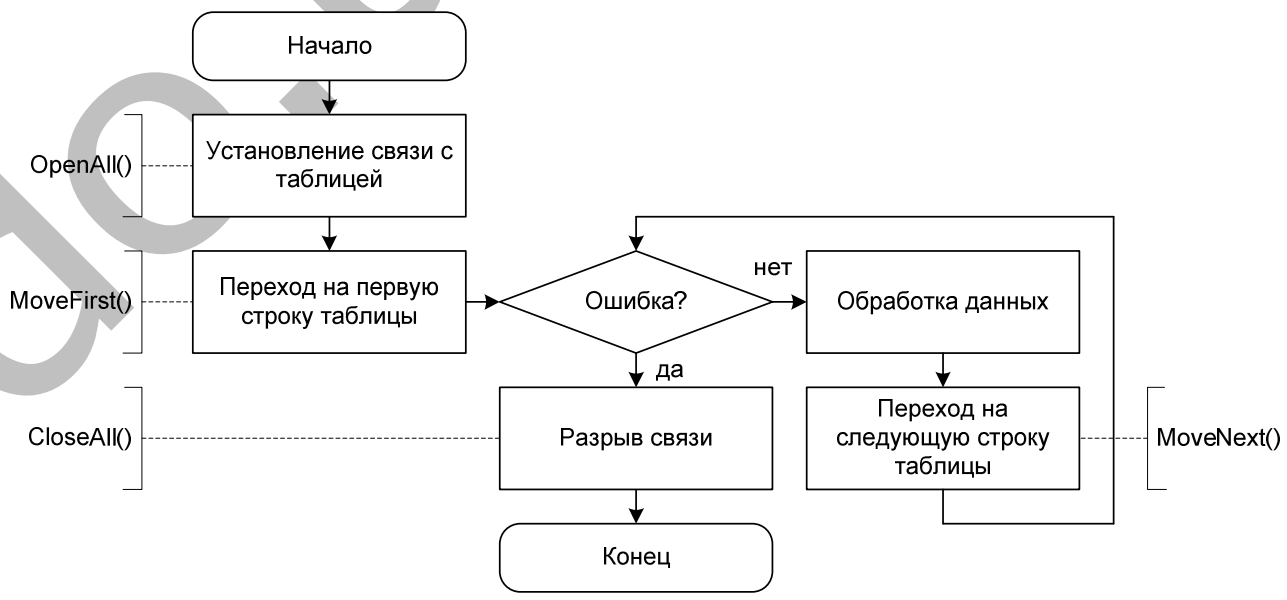

Рисунок 6.4 – Схема работы с таблицей БД

На рисунке 6.4 представлена схема работы с базой данных Access с использованием вышеприведенных методов.

В соответствии с заданием на курсовое проектирование результаты проведенного анализа необходимо представить в виде DOC-документа. Поскольку данные документы имеют закрытый формат, то целесообразно использовать возможности по дистанционному управлению средством Microsoft Word с использованием интерфейса COM. Для облегчения подключения и использования интерфейсов COM в Visual C++ введена директива #import.

Директива #import в автоматическом режиме генерирует библиотеку типов (информация о типе и свойствах COM-объектов) для заданного EXE- или DLL-файла. Содержащиеся в сгенерированных файлах описания классов и интерфейсов позволяют управлять COM-сервером и выполнять с его помощью требуемые действия. Использование директивы #import с Microsoft Word 2003 выглядит следующим образом:

```
#import "C:\Program Files\Common Files\Microsoft Shared\OFFICE11\MSO.DLL" \
        rename("RGB","_RGB") \
        rename("DocumentProperties","_DocumentProperties")
#import "C:\Program Files\Common Files\Microsoft Shared\VBA\VBA6\VBE6EXT.OLB"
#import "C:\Program Files\Microsoft Office\OFFICE11\MSWORD.OLB" \
        rename("ExitWindows","_ExitWindows")
```

Методы и атрибуты COM-сервера Microsoft Word, необходимые для выполнения работы:

- конструктор Create(L"Word.Application") – создание объекта COM-сервера;
- атрибут Visible – видимость COM-сервера;
- метод Activate – активация COM-сервера;
- метод Documents->Add – создание нового документа;
- метод Selection->Font->PutSize – задание размера текущего шрифта;
- метод Selection->ParagraphFormat->PutAlignment – задание выравнивания текста на странице;
- метод Selection->TypeText – вставка строки текста;
- метод SaveAs – сохранение текущего документа с заданным именем;
- метод Selection->TypeParagraph – вставка переноса на новую строку;
- метод Selection->Tables->Add – добавление таблица;
- и др.

Необходимо отметить, что компилятор помещает описание классов в отдельное пространство имён, соответствующее имени библиотеки типов (в данном случае – Word).

Реализованные с учетом всего ранее изложенного блок-схемы алгоритмов работы клиентской и серверной частей системы приведены в приложении В, а исходные тексты программных реализаций – в приложении Г.

7 Руководство пользователя

7.1 Использование серверной части системы

Для подготовки сервера автоматизированной системы анализа использования оборудования к запуску желательно скопировать файл сервера Kers_Server.exe и файл базы данных Equipment.mdb в одну папку (файл БД должен располагаться в рабочем каталоге сервера).

Существует 2 варианта запуска сервера:

– запуск без параметра командной строки – система автоматически назначает серверу свободный порт (рисунок 7.1);
– запуск с параметром командной строки, который содержит номер порта, назначаемый серверу (рисунок 7.2); порт должен быть свободен, поэтому рекомендуется использовать порты из диапазона 1025-65535.

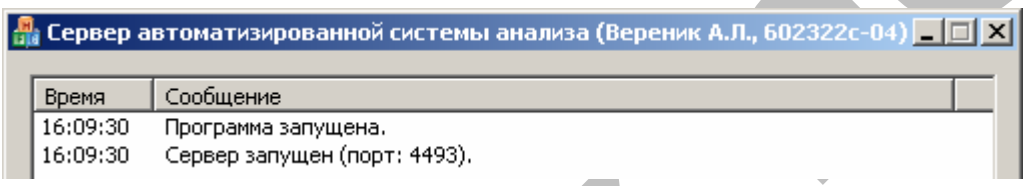

Рисунок 7.1 – Запуск сервера без параметра командной строки

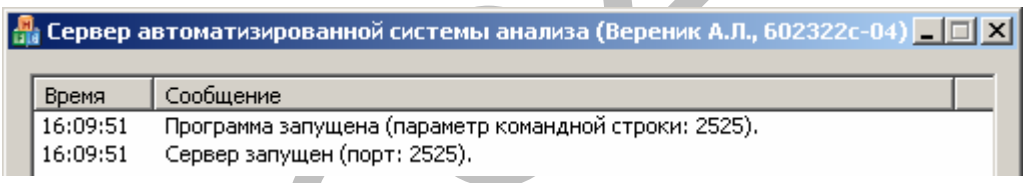

Рисунок 7.2 – Запуск сервера с параметром командной строки

Рекомендуется использовать второй способ, поскольку клиент обязательно должен знать номер порта, по которому расположен сервер (по умолчанию клиент предполагает, что сервер использует порт 2525).

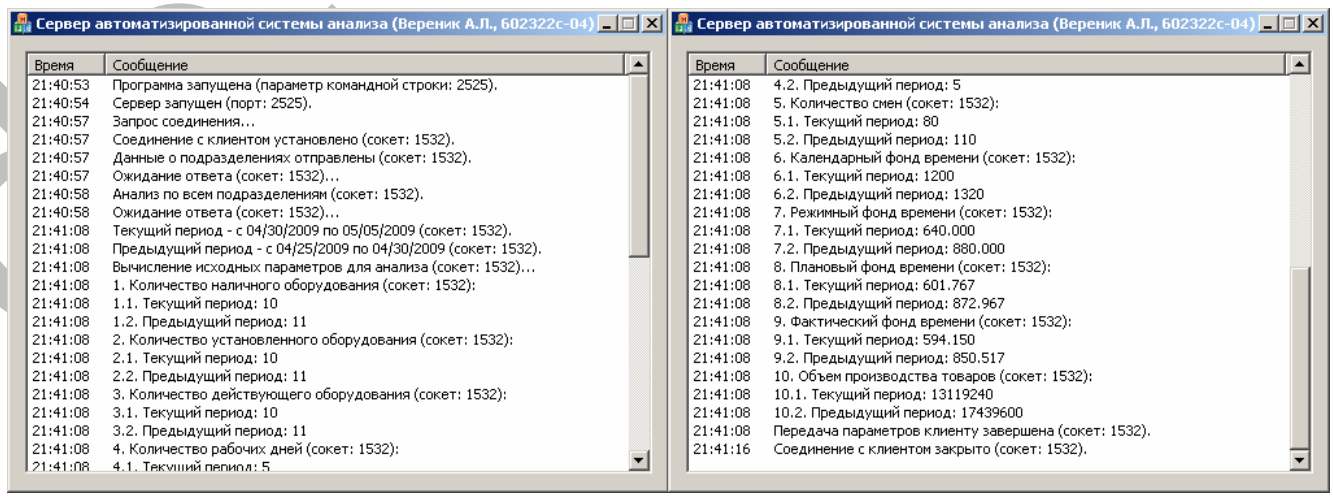

Рисунок 7.3 – Нормальная работа сервера

После запуска сервера он продолжает работать до своего закрытия, выводя в окне сервисные сообщения о выполняемых операциях или ошибках, возникших во время выполнения (рисунок 7.3).

Для работы сервера в системе должен быть установлен провайдер доступа к данным «Microsoft Jet 4.0 OLE DB Provider». В случае его отсутствия сервер выдаст сообщение об ошибке.

7.2 Использование клиентской части системы

Запуск клиента автоматизированной системы анализа использования оборудования (файл Kurs_Client.exe) производиться из любой папки. Для начала работы необходимо ввести адрес и порт сервера в соответствующие поля и нажать кнопку «Соединиться»:

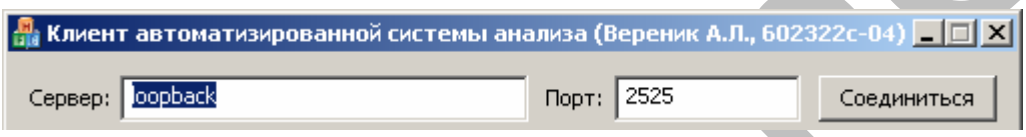

Рисунок 7.4 – Начало работы с клиентом

После соединения с сервером и получения списка подразделений предприятия будут выведены соответствующие сообщения и активированы список выбора подразделения и кнопка «Задать»:

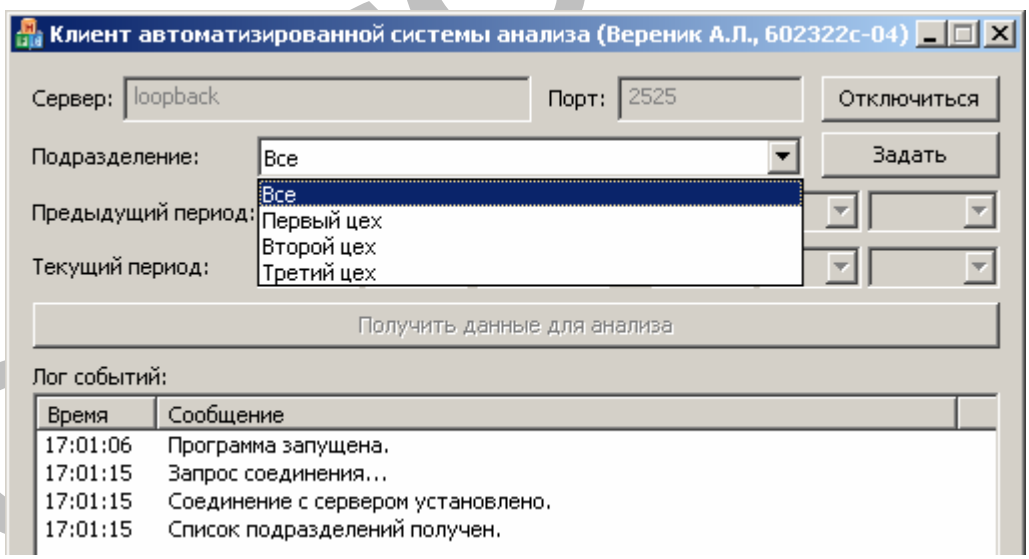

Рисунок 7.5 – Выбор структурного подразделения предприятия для анализа

После выбора необходимого подразделения по нажатию кнопки «Задать» оно отправляется на сервер, который возвращает граничные даты анализа. Одновременно активируются выпадающие списки для выбора дат периодов:

Рисунок 7.6 – Выбор периодов анализа

После нажатия кнопки «Получить данные для анализа» введенные периоды отправляются на сервер, откуда, после выполнения соответствующих выборок, возвращаются параметры для проведения анализа. Клиент запускает Microsoft Word, отображает стандартное окно сохранения файла, проводит анализ и выводит результаты в документ Word, после чего закрывает соединение:

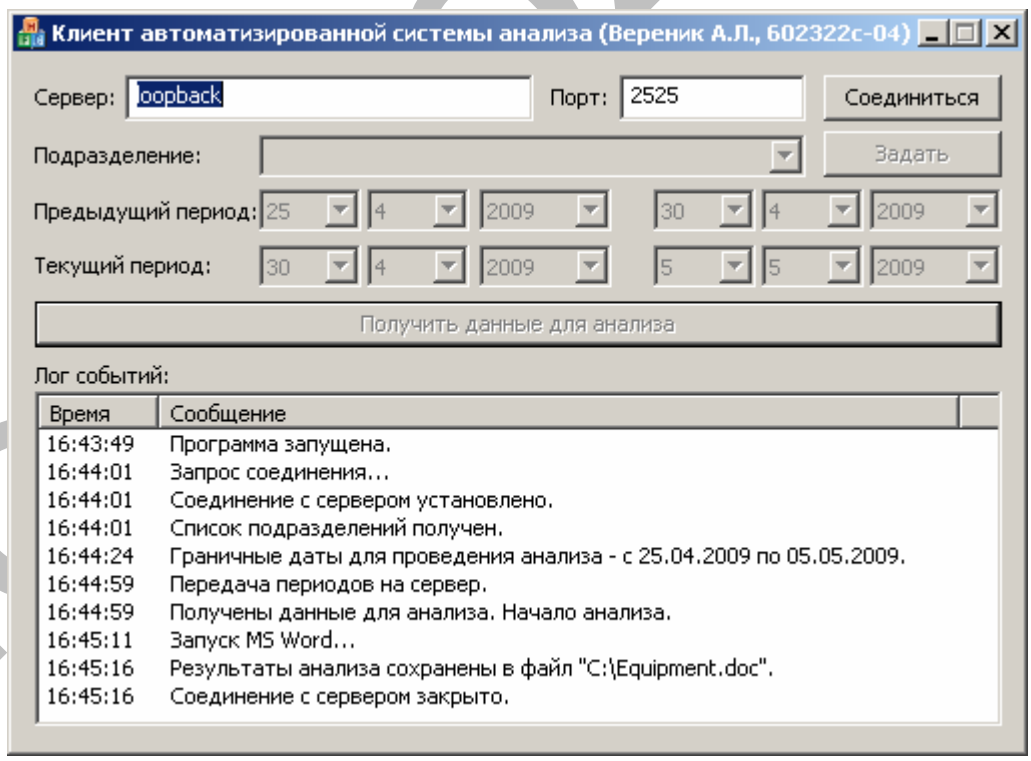

Рисунок 7.7 – Окончание анализа

Обязательным условием является наличие на компьютере, на котором запускается клиент системы наличие установленного редактора Microsoft Word версии не ниже 2003.

8 Пример работы с системой

Для проверки работоспособности системы БД была заполнена следующей информацией:

1) 3 подразделения: «Первый цех», «Второй цех», «Третий цех»;

2) 11 единиц оборудования, из которых 3 относятся к первому цеху, 4 – ко второму и 4 – к третьему; дата ввода всех станков 01.01.2009; дата вывода задана для одного из станков третьего цеха – 30.04.2009 и одного из станков второго цеха – 31.05.2009;

3) 190 результатов, за период с 25.04.2009 00:00 до 05.05.2009 00:00; все станки первого цеха работали по 2 смены ежедневно; все станки второго цеха – по 1 смене; до 29.04.2009 включительно все станки третьего цеха работали по 3 смены, а с 30.04.2009 – по 2 смены;

4) 127 записей об эксплуатации оборудования;

5) 20 записей о ремонте оборудования (11 – плановый ремонт, 9 - внеплановый) случайной продолжительности.

Записи об эксплуатации и ремонте в сумме по времени соответствуют 190 восьмичасовым сменам.

Для проверки работоспособности анализ проводился по всем цехам предприятия, за предыдущий был принят период с 25.04.2009 по 30.04.2009, а за текущий – период с 30.04.2009 по 05.05.2009.

В результате работы был сгенерирован DOC-файл следующего содержания:

Результаты проведенного анализа использования оборудования

Подразделение: все подразделения.
Текущий период: с 30.04.2009 по 05.05.2009
Предыдущий период: с 25.04.2009 по 30.04.2009

Таблица 1 - Анализ использования численности оборудования

Показатели	Предыдущий период	Текущий период	Отклонение (+, -)
1. Количество наличного оборудования, ед.	11	10	-1
1.1. установленного, ед.	11	10	-1
1.1.1 действующего, ед.	11	10	-1
1.1.2. бездействующего, ед.	0	0	0
1.2. не установленного, ед.	0	0	0
2. Коэффициенты использования:			
2.1. сданного в эксплуатацию оборудования	1.000000	1.000000	0
2.2. парка наличного оборудования	1.000000	1.000000	0
2.3. парка установленного оборудования	1.000000	1.000000	0

Таблица 2 - Анализ экстенсивного использования оборудования организации

Показатели	Предыдущий период	Текущий период	Отклонение (+, -)
1. Календарный фонд, машино-час.	1320.000	1200.000	-120.000
2. Режимный фонд, машино-час.	880.000	640.000	-240.000
3. Время планового ремонта, машино-час.	7.033	38.233	-31.200
4. Плановый фонд, машино-час.	872.967	601.767	-271.200
5. Время внеплановых простоев, машино-час.	22.450	7.617	14.833
6. Фактический фонд, машино-час.	850.517	594.150	-256.367

10. Коэффициенты использования оборудования во времени:			
- к календарному фонду	0.644331	0.495125	-0.149206
- к режимному фонду	0.966496	0.928359	-0.038137
- к реальному фонду	0.974283	0.987343	0.013060
- общий коэффициент экстенсивности	0.861703	0.803609	-0.058094

Таблица 3 - Данные об использовании оборудования в организации

Показатели	Предыдущий период	Текущий период	Отклонение (+, -)
1. Объем производства товаров, млн.руб.	17.440	13.119	-4.320
2. Количество оборудования, ед.	11	10	-1
2.1. в том числе действующее оборудование	11	10	-1
3. Производительность единицы работающего оборудования за весь период, млн.руб.	1.585	1.312	-0.273
4. Фонд времени работы всего оборудования, машино-час.			
4.1. плановый	872.967	601.767	-271.200
4.2. фактический	850.517	594.150	-256.367
4.3. единицы оборудования, машино-час.	77.320	59.415	-17.905
5. Среднечасовая выработка единицы оборудования, тыс.руб.			
5.1. плановая	19.977	21.801	1.824
5.2. фактическая	20.505	22.081	1.576
6. Коэффициент экстенсивности	0.861703	0.803609	-0.058094
7. Коэффициент интенсивности	1.026396	1.012819	-0.013576
8. Коэффициент интегральной нагрузки	0.884449	0.813911	-0.070538

Таблица 4 - Факторный анализ влияния эффективности использования оборудования на изменение объема производства товаров

Факторы	Уровень влияния на объем производства товаров, руб.	Структура факторов, %
1. Изменение количества действующего оборудования	-1585418.182	36.696
2. Изменение экстенсивной нагрузки	-3671306.700	84.977
3. Изменение интенсивной нагрузки	936364.882	-21.673
Итого	-4320360.000	100

Таблица 5 - Определение резервов увеличение выпуска продукции

Резерв	Увеличение
1. Ввод в действие дополнительной единицы оборудования, тыс.руб.	1311.924
2. Дополнительный рабочий день, тыс.руб.	2623.848
3. Дополнительная смена, тыс.руб.	163.991
4. Увеличение продолжительности смен на 1 час, тыс.руб.	1766.455
5. Прирост среднечасовой выработки на 1 тыс.руб., тыс.руб.	594.150

Полученные результаты были проверены путем проведения аналогичных расчетов традиционным способом, который подтвердил корректность полученных результатов. Следовательно, можно сделать вывод о соответствии разработанной системы поставленному заданию.

Заключение

В результате выполнения курсовой работы было разработана автоматизированная система анализа использования оборудования, использующая для проведения анализа информацию, хранящуюся в БД Access. Система была реализована на языке программирования C++ с использованием возможностей библиотеки MFC для реализации графического интерфейса пользователя, работы с БД и реализации клиент-серверной архитектуры.

В ходе выполнения работы были развиты навыки использования библиотеки MFC, изучены и практически опробованы сетевые возможности современных операционных систем Windows и средства работы с базами данных, рассмотрены стандарты моделирования IDEF0 и IDEF1x, а также получены практические умения в их построении с использованием средств BPwin и ERwin.

К достоинствам разработанной системы можно отнести:

1) Использование простого графического интерфейса, не перегруженного лишними элементами, обеспечивает интуитивно-понятное управление системой, как со стороны пользователя-экономиста, так и со стороны администратора.

2) Применение клиент-серверной архитектуры обеспечивает возможность проведения анализа пользователями с удаленных рабочих мест.

3) Использованная для реализации сервера псевдопараллельная обработка запросов обеспечивает возможность параллельной работы большого числа клиентов (точное количество определяется ресурсами конкретной системы, на которой запускается сервер).

4) Использование сервера как посредника между клиентом и базой данных, во-первых снижает требования к производительности системы для запуска клиентской части, во-вторых разгружает сеть, в-третьих обеспечивает конфиденциальность данных БД.

5) Реализованный в системе протокол обменом данных позволяет легко изменить способ получения информации (например, заменить работу с БД считыванием из файла) путем модификации серверной части без необходимости модификации клиентской.

6) Применение базы данных для хранения информации об использовании оборудования обеспечивает надежное хранение и быструю обработку данных.

Основным недостатком системы является необходимость участия экономиста для интерпретации численных результатов, полученных в результате анализа, поскольку конкретные рекомендации сильно зависят от конкретного предприятия, что, в свою очередь, затрудняет возможность их структурирования и алгоритмизации.

Список использованных источников

1) Визульные средства разработки приложений: Учеб.-метод. пособие / В.Н. Комличенко, О.П. Едемская, Н.А. Кириенко и др. - Мн.: БГУИР, 2004.

2) Карпычев В.Ю. Методология IDEF1X и программный продукт ERWin: Учебно-методическое пособие. - Нижний Новгород: ННГУ им. Н.И. Лобачевского, 2007.

3) Комплексный экономический анализ хозяйственной деятельности: Учебное пособие / А.И.Алексеева, Ю.В.Васильев, А.В., Малеева, Л.И.Ушвицкий. - М.: Финансы и статистика, 2006.

4) Методология функционального моделирования IDEF0: руководящий документ. - М.: Госстандарт России, 2000.

5) Методология функционального проектирования IDEF0: Учеб. пособие / В.В.Бахтизин, Л.А.Глухова. – Мн.: БГУИР, 2003.

6) Савицкая Г.В. Экономический анализ: учеб. - Мн.: Новое знание, 2005.

7) Структурный анализ и моделирование в среде CASE-средства BPwin: Учеб. пособие / В.В.Бахтизин, Л.А.Глухова. – Мн.: БГУИР, 2002.

8) Файзрахманов Р.А., Селезнев К.А. Учебное пособие к практическим занятиям «Структурно функциональный подход к проектированию информационных технологий и автоматизированных систем с использованием CASE-средств» / Перм. гос. техн. ун-т. – Пермь, 2005.

9) Чуев И.Н., Чуева Л.Н. Комплексный экономический анализ хозяйственной деятельности: Учебник для вузов. - М.: "Дашков и Ко", 2006.

10) Шеферд Дж. Программирование на Microsoft Visual C++ .NET. - М.: "Русская редакция", 2003.

Приложение А
(обязательное)

Модели предметной области

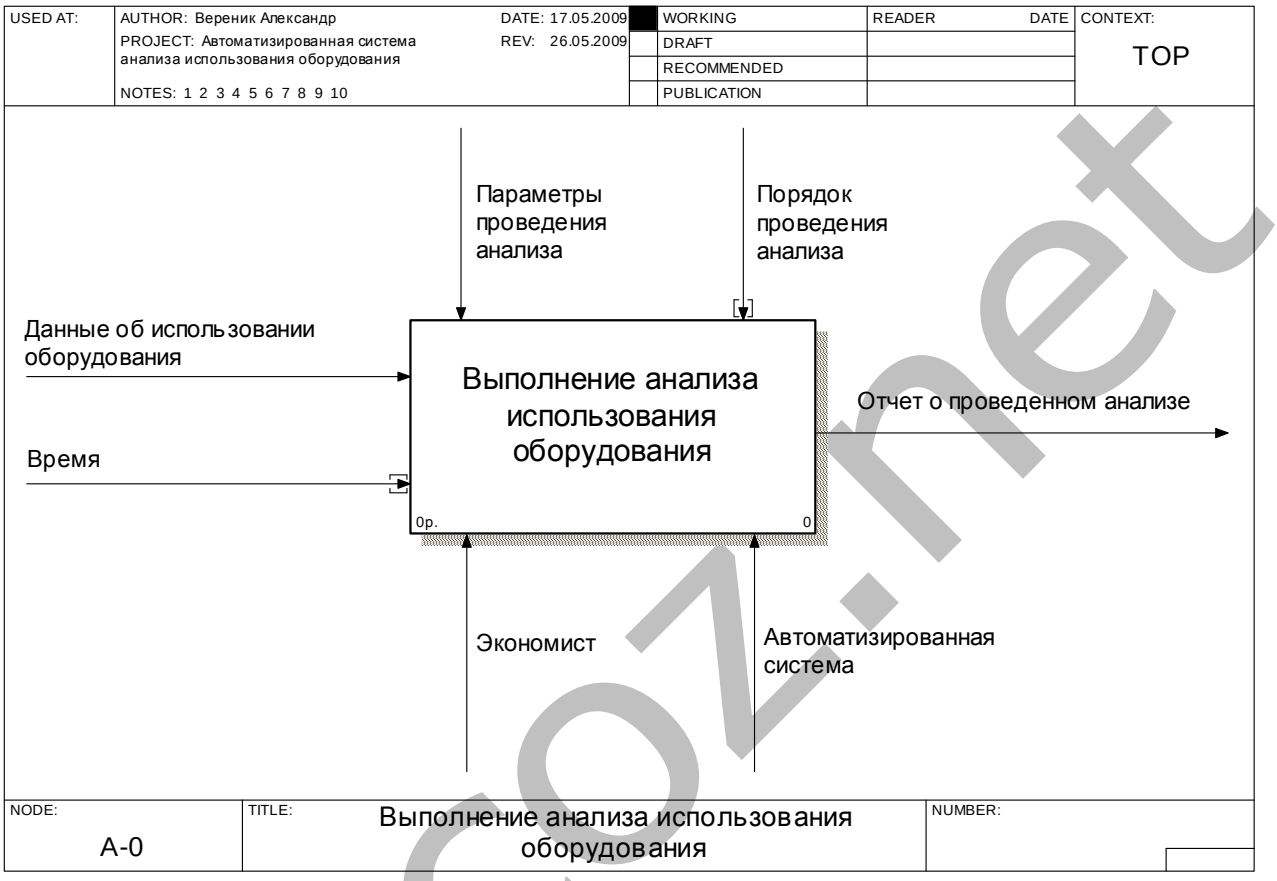

Рисунок А.1 – Контекстная диаграмма IDEF0

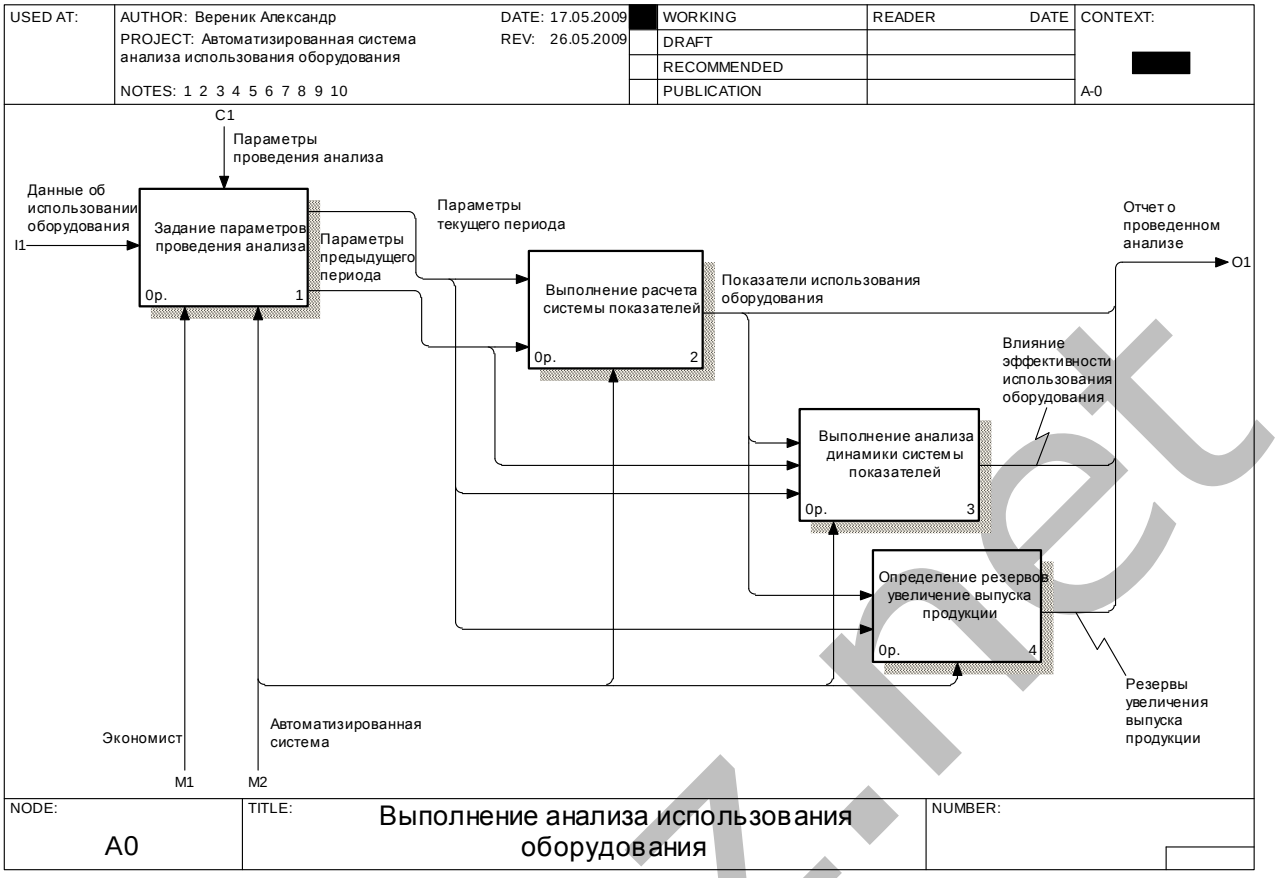

Рисунок А.2 – Выполнение анализа использования оборудования (уровень 0)

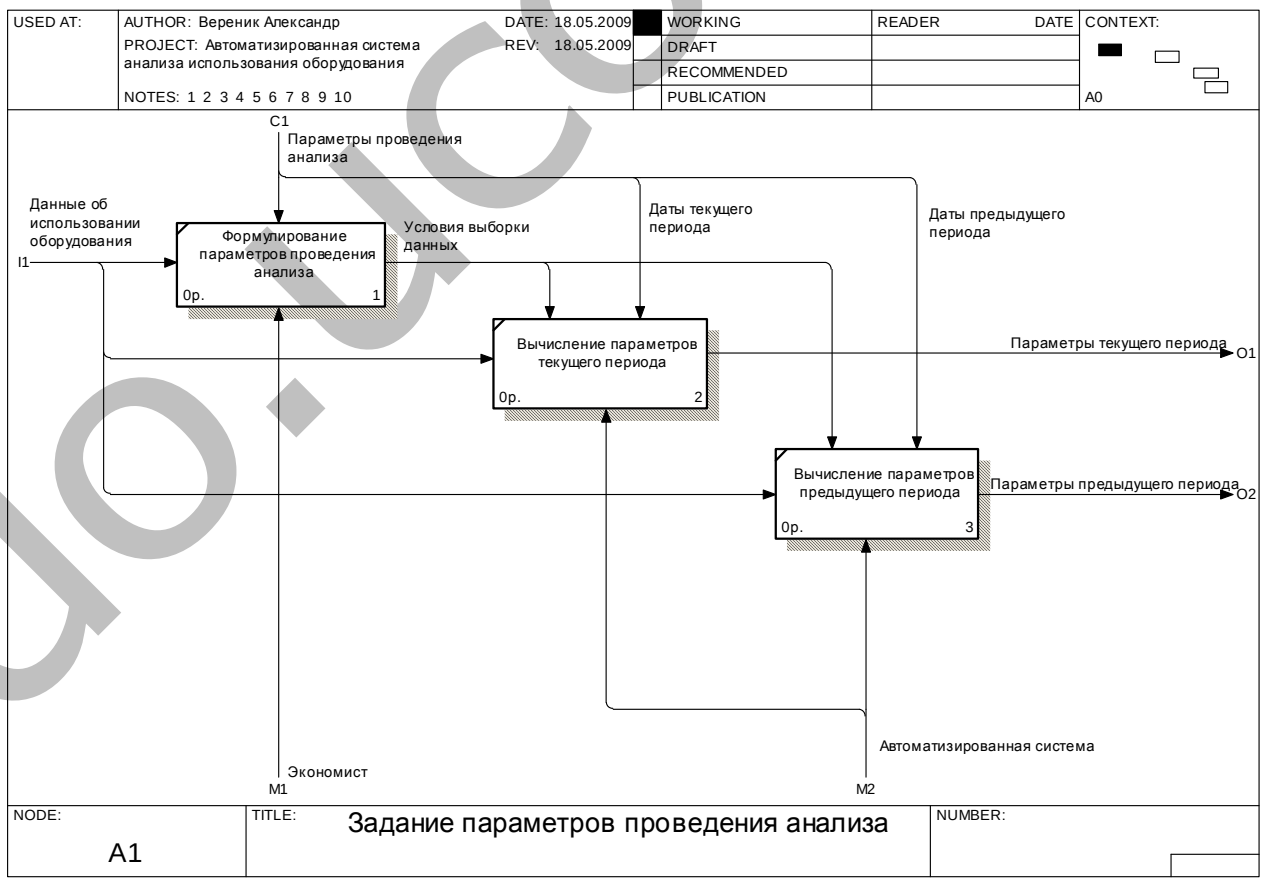

Рисунок А.3 – Задание параметров проведения анализа (уровень 1)

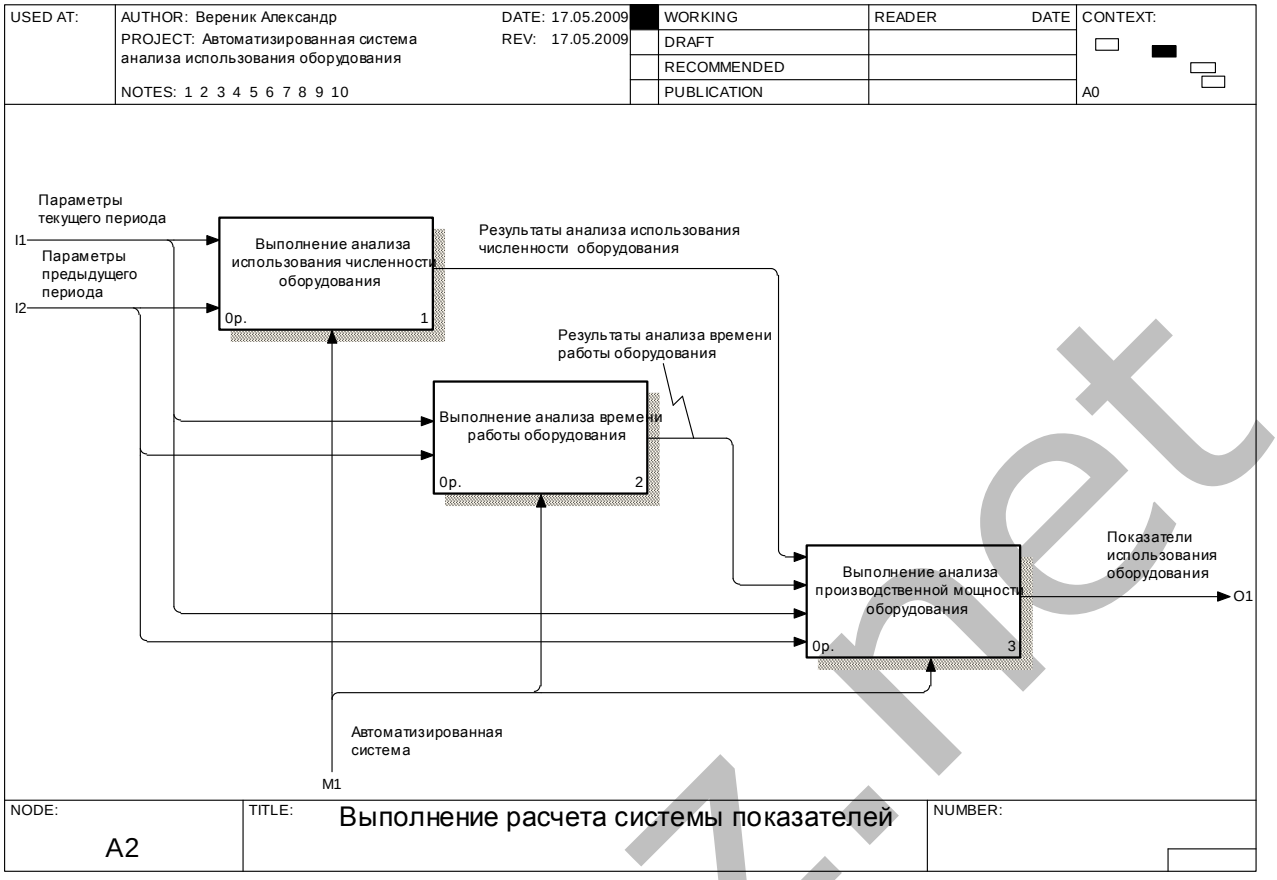

Рисунок А.4 – Выполнение расчета системы показателей (уровень 1)

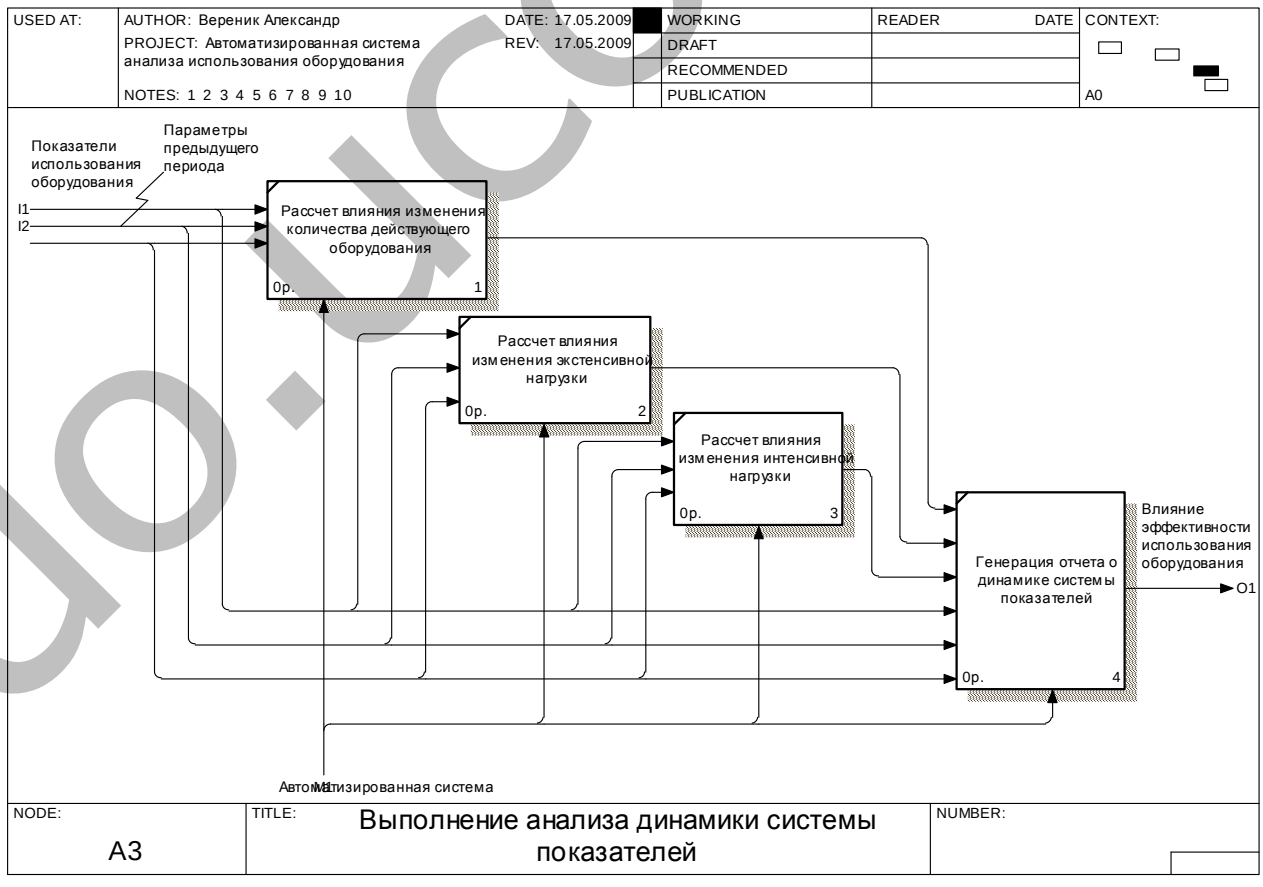

Рисунок А.5 – Выполнение анализа динамики системы показателей (уровень 1)

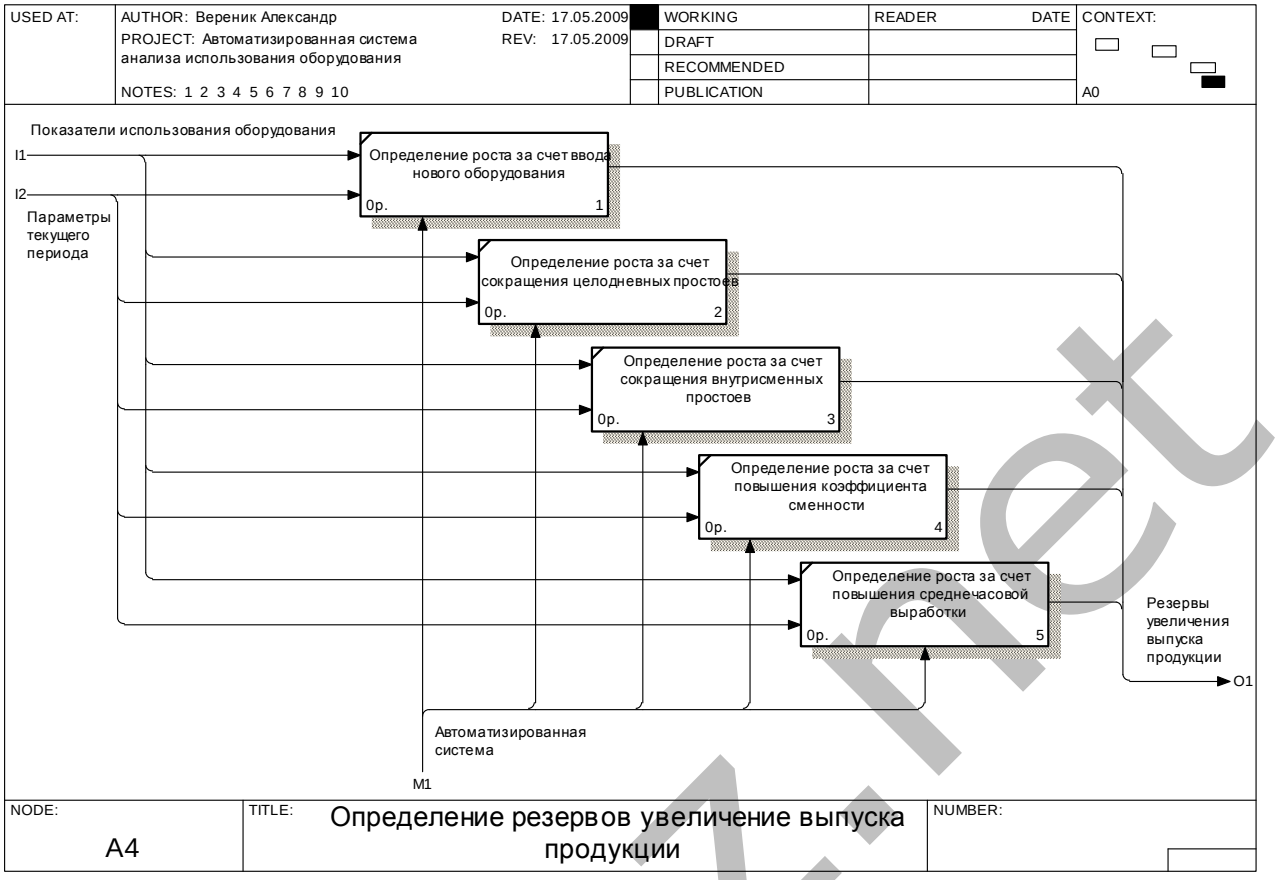

Рисунок А.6 – Определение резервов увеличения выпуска продукции (уровень 1)

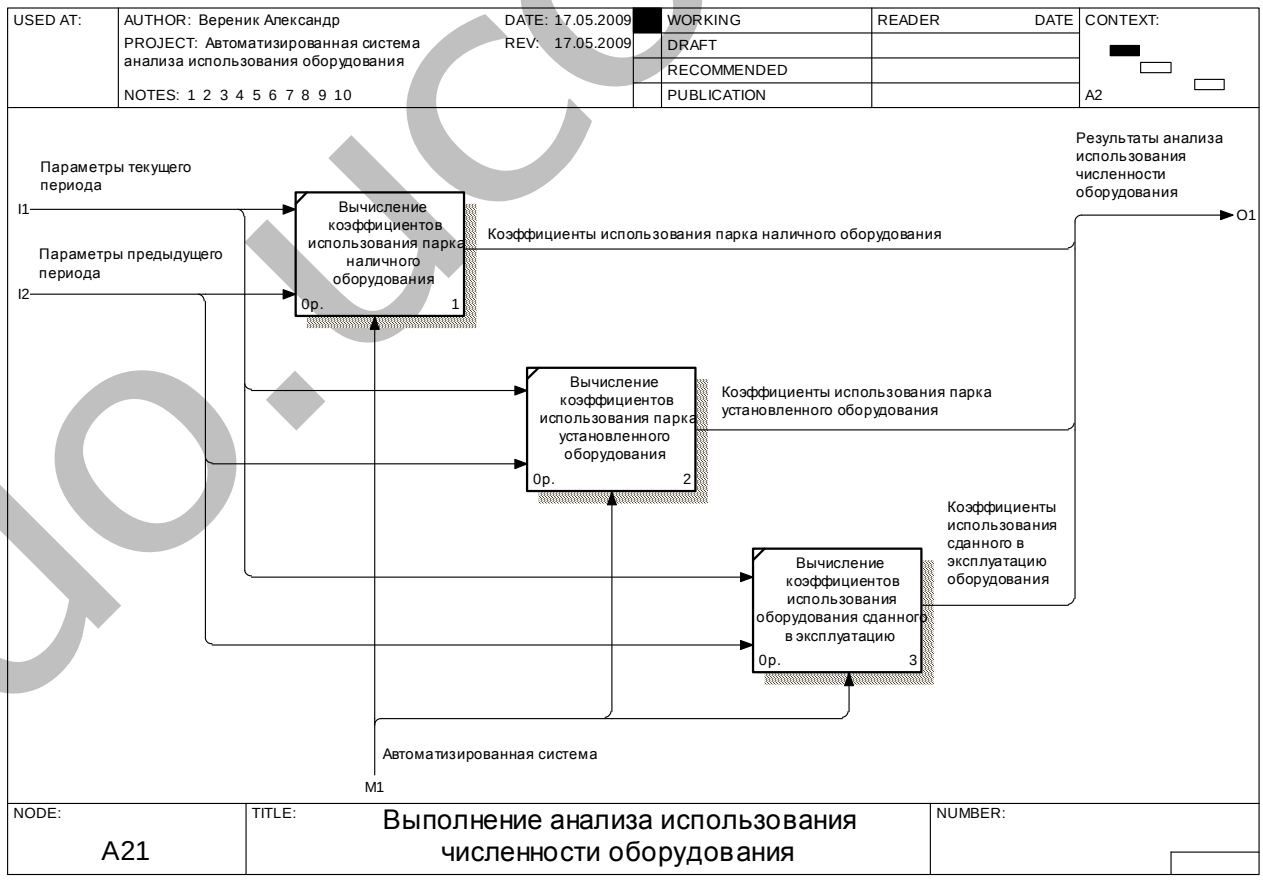

Рисунок А.7 – Выполнение анализа использования численности оборудования (уровень 2)

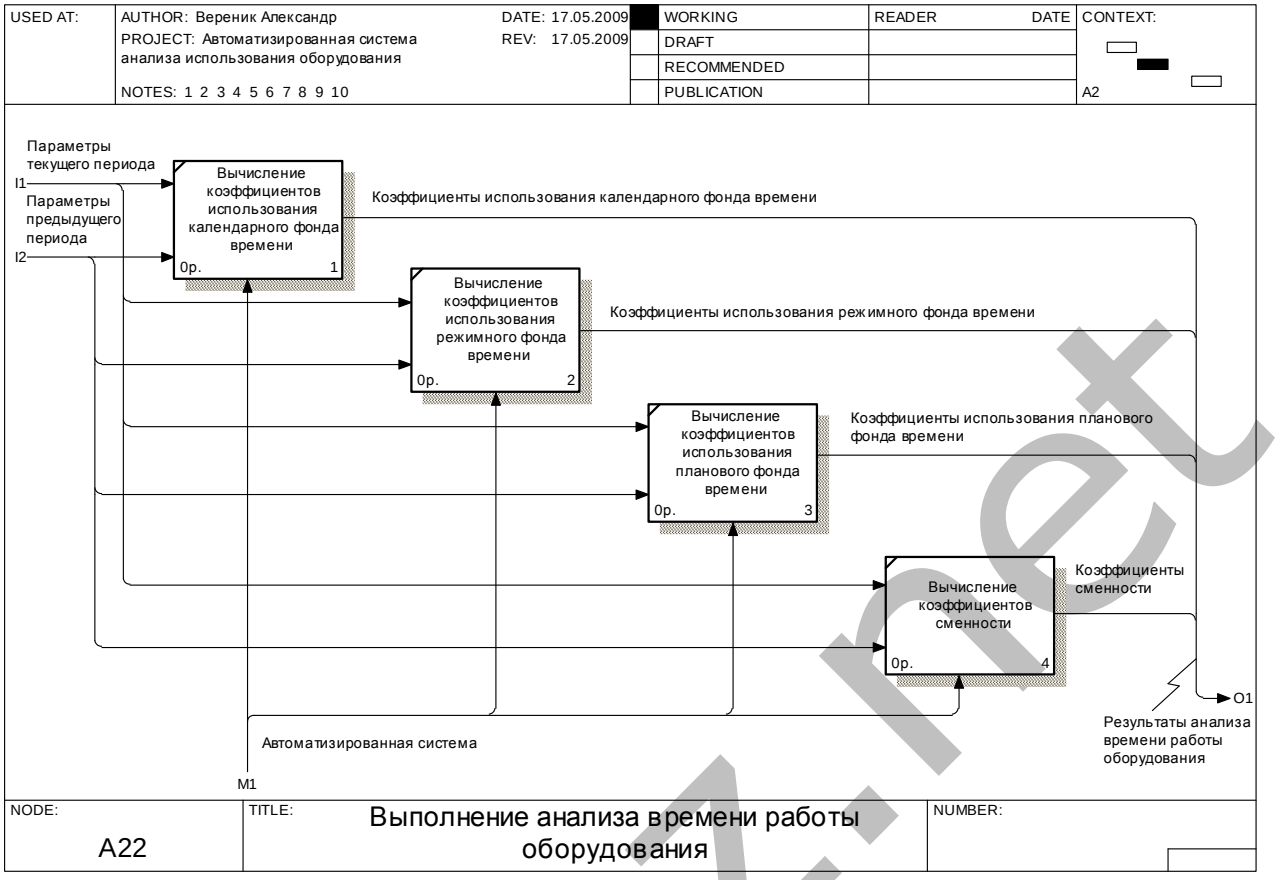

Рисунок А.8 – Выполнение анализа времени работы оборудования (уровень 2)

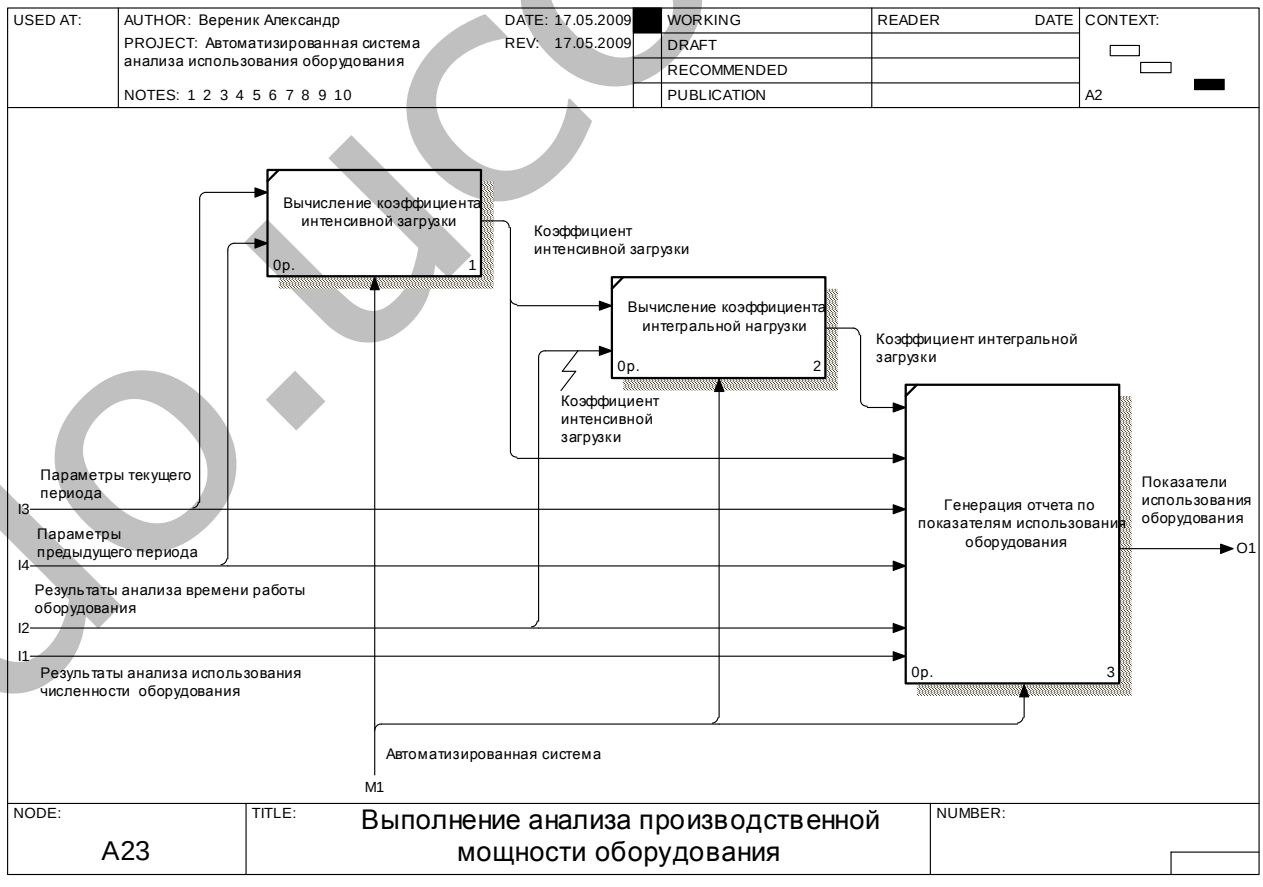

Рисунок А.9 – Выполнение анализа производственной мощности оборудования (уровень 2)

Приложение Б
(обязательное)

Информационные модели системы

Рисунок Б.1 – Информационная модель системы (логический уровень)

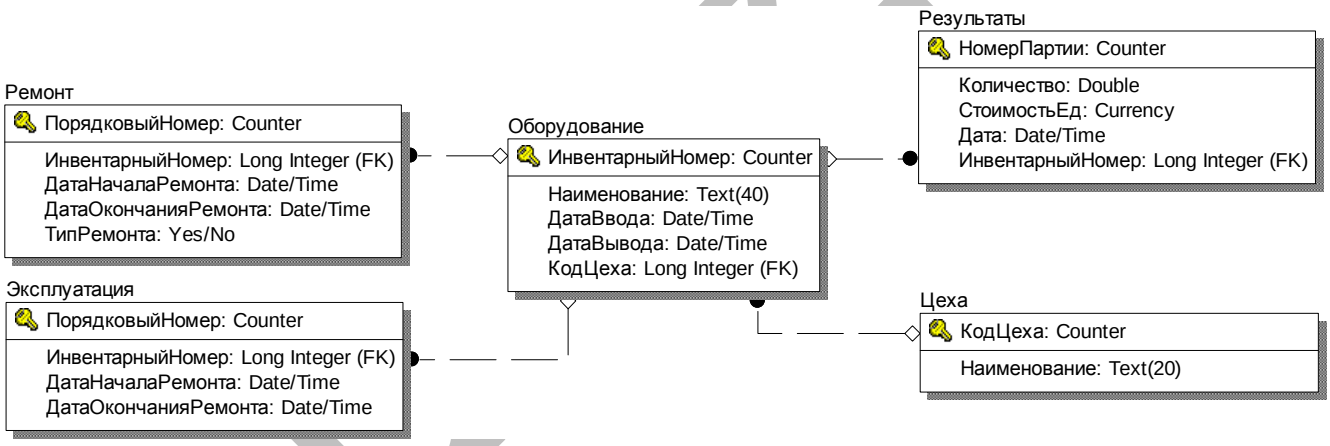

Рисунок Б.2 – Информационная модель системы (физический уровень)

Приложение В
(обязательное)

Блок-схемы алгоритмов работы программ

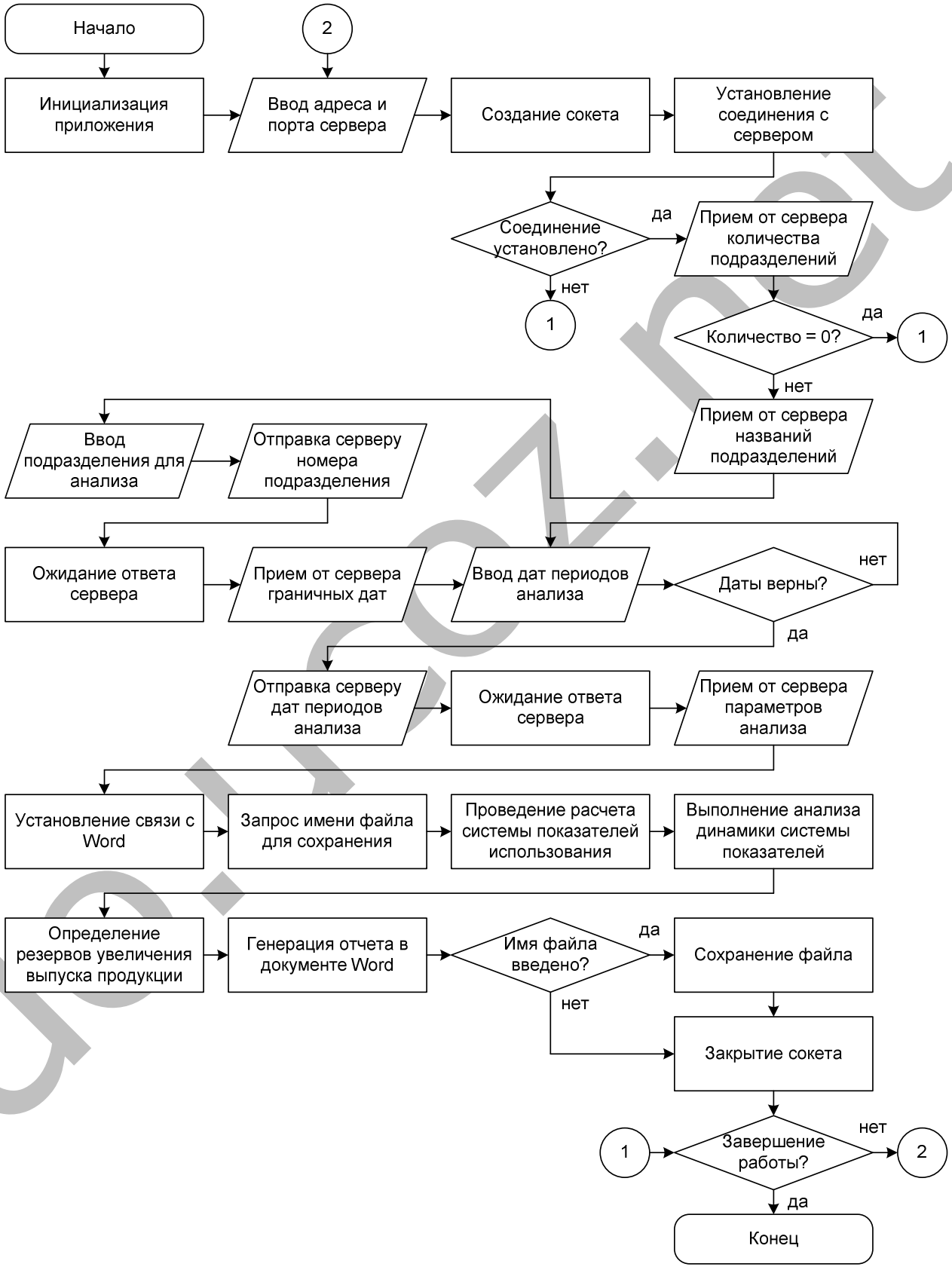

Рисунок В.1 – Блок-схема алгоритма работы клиента

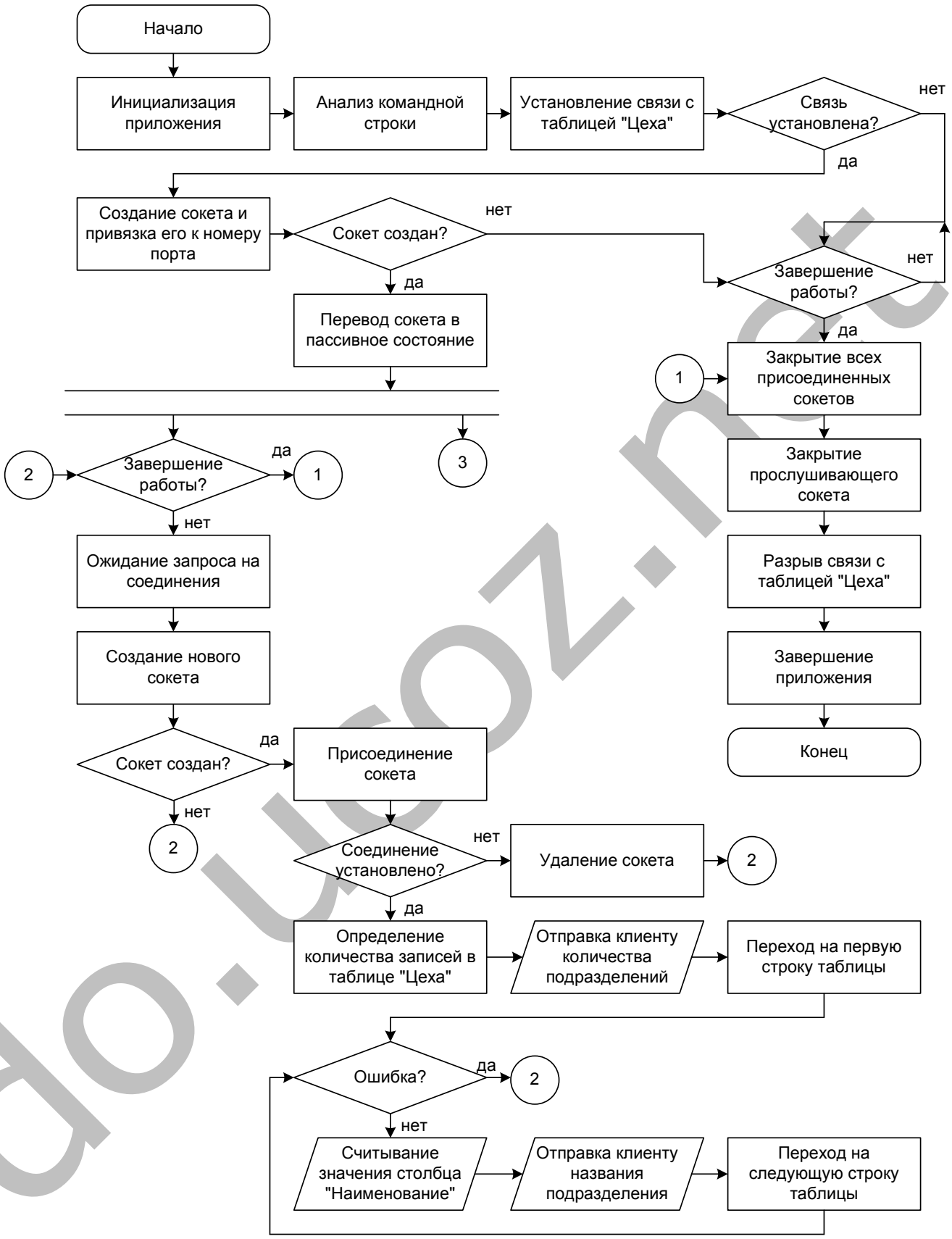

Рисунок В.2 – Блок-схема алгоритма работы сервера (часть 1)

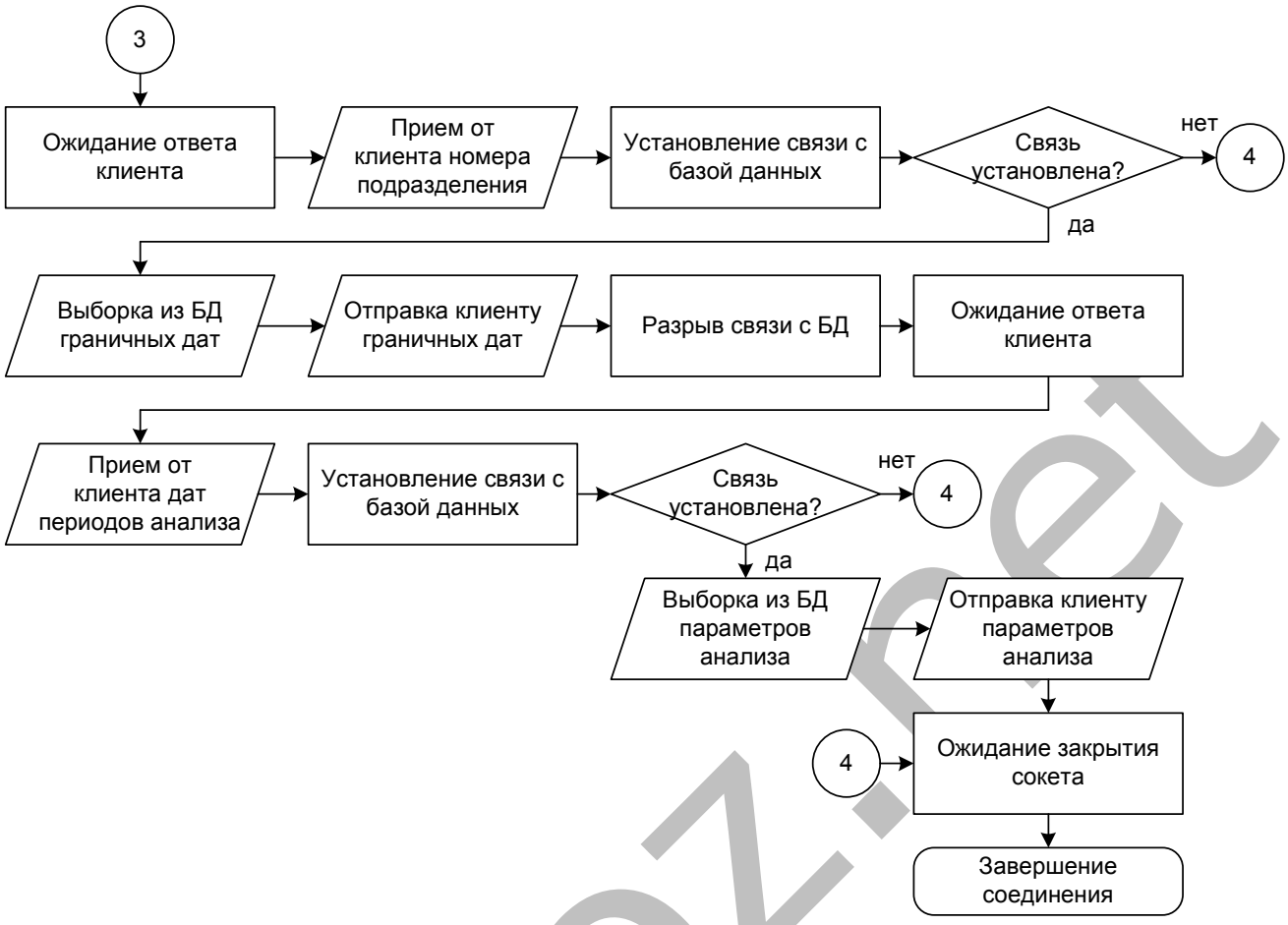

Рисунок В.3 – Блок-схема алгоритма работы сервера (часть 2)

Приложение Г
(обязательное)

Листинги программных кодов

1) Исходные тексты реализации клиента:

1.1) Заголовок классов для работы с сетью (MySocket.h):

```
#pragma once

enum SM
{ state_0, state_1, state_2, state_3, state_4, state_5, state_6, state_7, state_8, state_9, state_10,
state_11, state_12, state_13, state_14, state_15, state_16, state_17, state_18, state_19, state_20,
state_21, state_22, state_23 };
class CClientSocket_Wasja : public CAsyncSocket
{
private:
      CDialog* m_pWnd;
public:
      void SetParent( CDialog* pWnd );
      SM state;
      char read_buf [ 1025 ];
protected:
      void OnReceive( int nErrorCode );
      void OnConnect( int nErrorCode );
      void OnClose( int nErrorCode );
};
```

1.2) Реализация классов для работы с сетью (MySocket.cpp):

```
#include "stdafx.h"
#include "Kurs_Client.h"
#include "MySocket.h"
#include "Kurs_ClientDlg.h"

void CClientSocket_Wasja::SetParent( CDialog* pWnd )
{
      m_pWnd = pWnd;
}
void CClientSocket_Wasja::OnReceive( int nErrorCode )
{
      if (nErrorCode == 0)
      {
            char msg [ 2 ];
            msg[ 1 ] = '\0';
            int i = Receive( &msg, 1 );
            while (i > 0)
            {
                  if (msg[ 0 ] == '\n')
                  {
                        ((CKurs_ClientDlg_Wasja*)m_pWnd)->OnReceive();
                        FillMemory( read_buf, sizeof( read_buf ), 0 );
                  }
                  else strcat( read_buf, msg );
                  i = Receive( &msg, 1 );
            }
      }
      CAsyncSocket::OnReceive( nErrorCode );
}
void CClientSocket_Wasja::OnConnect( int nErrorCode )
{
      if (nErrorCode == 0) ((CKurs_ClientDlg_Wasja*)m_pWnd)->PrintMessage( "Соединение с сервером
установлено." );
      CAsyncSocket::OnConnect( nErrorCode );
}
void CClientSocket_Wasja::OnClose( int nErrorCode )
{
      if (nErrorCode == 0) ((CKurs_ClientDlg_Wasja*)m_pWnd)->OnClose();
      CAsyncSocket::OnClose( nErrorCode );
}
```

1.3) Заголовок класса работы с окном (Kurs_ClientDlg.h):

```
#pragma once
#include "afxwin.h"
#include "MySocket.h"
#include "afxcmn.h"
#include "Office.h"
using namespace Word;
struct A_param
{
	int equipment_total;
	// Количество наличного оборудования
	int equipment_set;
	// Количество установленного оборудования
	int equipment_work;
	// Количество действующего оборудования
	BYTE equipment_power;
	double time_calendar;
	// Календарный фонд времени
	double time_regime;
	// Режимный фонд времени
	double time_plan;
	// Плановый фонд времени
	double time_fact;
	// Фактический фонд времени
	BYTE time_power;
	double amount_products;
	// Объем производства товаров
	BYTE amount_power;
	int workday_count;
	// Количество рабочих дней
	int shift_count;
	// Количество смен (за весь период)
};
struct AUCE_param
{
	double K_exploit[ 2 ];
	// Коэффициенты использования оборудования сданного в эксплуатацию
	double K_total[ 2 ];
	// Коэффициенты использования парка наличного оборудования
	double K_set[ 2 ];
	// Коэффициенты использования парка установленного оборудования
};
struct AEUE_param
{
	double K_calendar[ 2 ];
	// Коэффициенты использования календарного фонда времени
	double K_regime[ 2 ];
		// Коэффициенты использования режимного фонда времени
	double K_plan[ 2 ];
	// Коэффициенты использования планового фонда времени
	double K_extensive[ 2 ];
	// Общие коэффициенты экстенсивности
};
struct AIUE_param
{
	double productivity[ 2 ];
	// Производительность единицы работающего оборудования за весь период
	double time_unit[ 2 ];
	// Фонд времени работы единицы оборудования
	double manufacture_plan[ 2 ];
	// Среднечасовая выработка единицы оборудования (плановая)
	double manufacture_fact[ 2 ];
	// Среднечасовая выработка единицы оборудования (фактическая)
	double K_intensive[ 2 ];
	// Общие коэффициенты интенсивности
	double K_integral[ 2 ];
	// Коэффициент интегральной нагрузки
};
struct FAEUE_param
{
	double D_count;
	// Влияние изменения количества действующего оборудования
	double D_extensive;
	// Влияние изменения количества экстенсивной нагрузки
	double D_intensive;
	// Влияние изменения количества интенсивной нагрузки
	double D_sum;
	// Влияние изменения суммарное
};
struct FRIO_param
```

```
{
	double Reserve_0;
	// Рост за счет ввода нового оборудования
	double Reserve_1;
	// Рост за счет сокращения целодневных простоев
	double Reserve_2;		// Рост за счет сокращения внутрисменных простоев
	double Reserve_3;		// Рост за счет повышения коэффициента сменности
	double Reserve_4;		// Рост за счет повышения среднечасовой выработки
};
class CKurs_ClientDlg_Wasja : public CDialog
{
public:
	CKurs_ClientDlg_Wasja(CWnd* pParent = NULL);
	enum { IDD = IDD_KURS_CLIENT_DIALOG };
protected:
	virtual void DoDataExchange(CDataExchange* pDX);
	HICON m_hIcon;
	virtual BOOL OnInitDialog();
	DECLARE_MESSAGE_MAP()
public:
	CButton m_ctlConnect;
	CString m_strName;
	int m_iPort;
	CListCtrl m_lLog;
	BOOL Connected;
	CString BeginDate;
	CString EndDate;
protected:
	CClientSocket_Wasja m_sConnectSocket;
	A_param prev_period;
	A_param curr_period;
public:
	afx_msg void OnBconnect();
	void OnReceive();
	void OnClose();
	afx_msg void OnBSend();
	void PrintMessage( const char * msg );
	CComboBox m_cDep;
	afx_msg void OnBnClickedBdep();
	CComboBox m_cCurr_Begin_Day;
	CComboBox m_cCurr_Begin_Month;
	CComboBox m_cCurr_Begin_Year;
	CComboBox m_cCurr_End_Day;
	CComboBox m_cCurr_End_Month;
	CComboBox m_cCurr_End_Year;
	CComboBox m_cPrev_Begin_Day;
	CComboBox m_cPrev_Begin_Month;
	CComboBox m_cPrev_Begin_Year;
	CComboBox m_cPrev_End_Day;
	CComboBox m_cPrev_End_Month;
	CComboBox m_cPrev_End_Year;
	inline void IntToStr( int value, char * buffer );
	void DoubleToStr( double value, char * buffer, BOOL three );
	AUCE_param Analysis_Usign_Count_Equipments( _ApplicationPtr WordApp, const A_param prev_period,
const A_param curr_period );
	AEUE_param Analysis_Extensive_Usign_Equipments( _ApplicationPtr WordApp, const A_param
prev_period, const A_param curr_period );
	AIUE_param Analysis_Intensive_Usign_Equipments( _ApplicationPtr WordApp, const A_param
prev_period, const A_param curr_period, const AEUE_param AEUE_coef );
	FAEUE_param Factor_Analysis_Effective_Usign_Equipments( _ApplicationPtr WordApp, const A_param
prev_period, const A_param curr_period, AIUE_param AIUE_coef );
	FRIO_param Find_Reserve_Increase_Output( _ApplicationPtr WordApp, const A_param curr_period );
};
```

2) Исходные тексты реализации сервера:

2.1) Заголовок классов для работы с сетью (MySocket.h):

```
#pragma once
class CServerSocket_Wasja : public CAsyncSocket
{
private:
	CDialog* m_pWnd;
public:
	void SetParent( CDialog* pWnd ) { m_pWnd = pWnd; };
protected:
	void OnAccept( int nErrorCode );
};
enum SM { state_0, state_1, state_2, state_3, state_4, state_5, state_6, state_7 };
class CConnectSocket_Wasja : public CAsyncSocket
{
```

```
private:
	CDialog* m_pWnd;
public:
	void SetParent( CDialog* pWnd ) { m_pWnd = pWnd; };
	SM state;
	char read_buf [ 1025 ];
	int NumberDepartment;
	char PrevBeginDate [ 11 ];
	char PrevEndDate [ 11 ];
	char CurrBeginDate [ 11 ];
	char CurrEndDate [ 11 ];
protected:
	void OnReceive( int nErrorCode );
	void OnClose( int nErrorCode );
};
```

2.2) Реализация классов для работы с сетью (MySocket.cpp):

```
#include "stdafx.h"
#include "Kurs_Server.h"
#include "MySocket.h"
#include "Kurs_ServerDlg.h"
void CServerSocket_Wasja::OnAccept( int nErrorCode )
{
	if (nErrorCode==0) ((CKurs_ServerDlg_Wasja*)m_pWnd)->OnAccept();
	CAsyncSocket::OnAccept( nErrorCode );
}
void CConnectSocket_Wasja::OnReceive( int nErrorCode )
{
	if (nErrorCode == 0)
	{
		char msg [ 2 ];
		msg[ 1 ] = '\0';
		int i = Receive( &msg, 1 );
		while (i > 0)
		{
			if (msg[ 0 ] == '\n') ((CKurs_ServerDlg_Wasja*)m_pWnd)->OnReceive( this );
			else strcat( read_buf, msg );
			i = Receive( &msg, 1 );
		}
	}
	CAsyncSocket::OnReceive( nErrorCode );
}
void CConnectSocket_Wasja::OnClose( int nErrorCode )
{
	if (nErrorCode==0) ((CKurs_ServerDlg_Wasja*)m_pWnd)->OnClose( nErrorCode, this );
	CAsyncSocket::OnClose( nErrorCode );
}
```

2.3) Классы для работы с базой данных (db_source.h):

```
#pragma once
[
	db_source( L"Provider=Microsoft.Jet.OLEDB.4.0;User ID=Admin;Data
Source=Equipment.mdb;Mode=ReadWrite" ),
	db_table( L"Цеха" )
]
class CDepartmentSet_Wasja
{
public:
	[ db_column( 1, status=m_dwcolumn0Status, length=m_dwcolumn0Length ) ] LONG m_column0;
	[ db_column( 2, status=m_dwcolumn1Status, length=m_dwcolumn1Length ) ] TCHAR m_column1[21];
	DBSTATUS m_dwcolumn0Status;
	DBSTATUS m_dwcolumn1Status;
	DBLENGTH m_dwcolumn0Length;
	DBLENGTH m_dwcolumn1Length;
	void GetRowsetProperties( CDBPropSet* pPropSet )
	{
		pPropSet->AddProperty( DBPROP_CANFETCHBACKWARDS, true, DBPROPOPTIONS_OPTIONAL );
		pPropSet->AddProperty( DBPROP_CANSCROLLBACKWARDS, true, DBPROPOPTIONS_OPTIONAL );
	}
};

[
	db_source( L"Provider=Microsoft.Jet.OLEDB.4.0;User ID=Admin;Data
Source=Equipment.mdb;Mode=ReadWrite" ),
	db_table( L"SELECT Min(ДатаНачалаРаботы) FROM Эксплуатация" )
]
```

```
class CGetDate_Wasja
{
public:
	[ db_column( 1, status=m_dwcolumn0Status, length=m_dwcolumn0Length ) ] DBTIMESTAMP m_column0;
	DBSTATUS m_dwcolumn0Status;
	DBLENGTH m_dwcolumn0Length;
	void GetRowsetProperties( CDBPropSet* pPropSet )
	{
		pPropSet->AddProperty( DBPROP_CANFETCHBACKWARDS, true, DBPROPOPTIONS_OPTIONAL );
		pPropSet->AddProperty( DBPROP_CANSCROLLBACKWARDS, true, DBPROPOPTIONS_OPTIONAL );
	}
};

[
	db_source( L"Provider=Microsoft.Jet.OLEDB.4.0;User ID=Admin;Data
Source=Equipment.mdb;Mode=ReadWrite" ),
	db_table( L"SELECT Count(ИнвентарныйНомер) FROM Оборудование" )
]
class CGetCount_Wasja
{
public:
	[ db_column( 1, status=m_dwcolumn0Status, length=m_dwcolumn0Length ) ] LONG m_column0;
	DBSTATUS m_dwcolumn0Status;
	DBLENGTH m_dwcolumn0Length;
	void GetRowsetProperties( CDBPropSet* pPropSet )
	{
		pPropSet->AddProperty( DBPROP_CANFETCHBACKWARDS, true, DBPROPOPTIONS_OPTIONAL );
		pPropSet->AddProperty( DBPROP_CANSCROLLBACKWARDS, true, DBPROPOPTIONS_OPTIONAL );
	}
};

[
	db_source( L"Provider=Microsoft.Jet.OLEDB.4.0;User ID=Admin;Data
Source=Equipment.mdb;Mode=ReadWrite" ),
	db_table( L"SELECT Count(ИнвентарныйНомер) FROM Оборудование" )
]
class CGetDouble_Wasja
{
public:
	[ db_column( 1, status=m_dwcolumn0Status, length=m_dwcolumn0Length ) ] double m_column0;
	DBSTATUS m_dwcolumn0Status;
	DBLENGTH m_dwcolumn0Length;
	void GetRowsetProperties( CDBPropSet* pPropSet )
	{
		pPropSet->AddProperty( DBPROP_CANFETCHBACKWARDS, true, DBPROPOPTIONS_OPTIONAL );
		pPropSet->AddProperty( DBPROP_CANSCROLLBACKWARDS, true, DBPROPOPTIONS_OPTIONAL );
	}
};
```

2.4) Заголовок класса работы с окном (Kurs_ServerDlg.h):

```
#pragma once
#include "MySocket.h"
#include "db_source.h"
#include "stdafx.h"
class CKurs_ServerDlg_Wasja : public CDialog
{
public:
	CKurs_ServerDlg_Wasja(CWnd* pParent = NULL);
	~CKurs_ServerDlg_Wasja();
	enum { IDD = IDD_KURS_SERVER_DIALOG };
	protected:
	virtual void DoDataExchange(CDataExchange* pDX);
protected:
	HICON m_hIcon;
	virtual BOOL OnInitDialog();
	afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
	DECLARE_MESSAGE_MAP()
public:
	wchar_t * connectstring;
	CListCtrl m_lLog;
	CDepartmentSet_Wasja m_DepartmentSet;
protected:
	CServerSocket_Wasja m_sListenSocket;
	CPtrList m_SocketList;
public:
	void OnAccept();
	void OnClose( int nErrorCode, CConnectSocket_Wasja * pSocket );
	void OnReceive( CConnectSocket_Wasja * pSocket );
	afx_msg void OnBSend();
	void PrintMessage( const char * msg );
```

```
};
```

3) BAT-файл для запуска сервера с номером порта 2525:

```
start Kurs_Server.exe 2525
```

4) SQL-код создания таблиц базы данных:

```
CREATE TABLE Цеха (
       КодЦеха              counter PRIMARY KEY,
       Наименование         varchar(20) NOT NULL
);

CREATE TABLE Оборудование (
       ИнвентарныйНомер     counter PRIMARY KEY,
       Наименование         varchar(40) NOT NULL,
       ДатаВвода            datetime NOT NULL,
       ДатаВывода           datetime NULL,
       КодЦеха              integer NOT NULL REFERENCES Цеха (КодЦеха)
);

CREATE TABLE Результаты (
       НомерПартии          counter PRIMARY KEY,
       Количество           float NOT NULL,
       СтоимостьЕд          money NOT NULL,
       Дата                 datetime NOT NULL,
       ИнвентарныйНомер     integer NOT NULL
                            REFERENCES Оборудование (ИнвентарныйНомер)
);

CREATE TABLE Ремонт (
       ПорядковыйНомер      counter PRIMARY KEY,
       ИнвентарныйНомер     integer NOT NULL
                            REFERENCES Оборудование (ИнвентарныйНомер),
       ДатаНачалаРемонта    datetime NOT NULL,
       ДатаОкончанияРемонта datetime NULL,
       ТипРемонта           bit NOT NULL
);


CREATE TABLE Эксплуатация (
       ПорядковыйНомер      counter PRIMARY KEY,
       ИнвентарныйНомер     integer NOT NULL
                            REFERENCES Оборудование (ИнвентарныйНомер),
       ДатаНачалаРемонта    datetime NOT NULL,
       ДатаОкончанияРемонта datetime NULL
);
```