САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

На правах рукописи

АЛИЕВ ХАДЖИМУРАД РАСУЛОВИЧ

«Модель планирования и управления разработкой сложных программных систем на основе комбинированной методики оценки трудозатрат»

Специальность 08.00.13 – «Математические и инструментальные методы экономики»

АВТОРЕФЕРАТ

диссертации на соискание ученой степени

кандидата экономических наук

Санкт-Петербург
2010

Работа выполнена в Федеральном государственном образовательном учреждении высшего профессионального образования «Санкт-Петербургский государственный университет» на кафедре информационных систем в экономике

Научный руководитель:	доктор физико-математических наук, профессор **Юрков Александр Васильевич**
Официальные оппоненты:	доктор экономических наук, профессор **Стельмашонок Елена Викторовна** кандидат экономических наук, доцент **Ермоленко Константин Юрьевич**
Ведущая организация:	Санкт-Петербургский государственный политехнический университет

Защита состоится «____» _____ 2010г. в «_____» часов на заседании Совета Д.212.232.34 по защите докторских и кандидатских диссертаций при Санкт-Петербургском Государственном Университете по адресу: 191123, Санкт-Петербург, ул. Чайковского, д.62, экономический факультет, ауд._____

С диссертацией можно ознакомиться в библиотеке им. А.М. Горького Санкт-Петербургского государственного университета.

Автореферат разослан «_____»_____ 2010 г.

Ученый секретарь
диссертационного совета,
кандидат экономических наук, доцент В.И. Капусткин

I. ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы исследования. В процессе разработки крупных программных систем на различных этапах выполнения проекта его участники сталкиваются с задачами технико-экономического обоснования (ТЭО). Прежде всего, это задачи оценки трудозатрат и ресурсов, необходимых для их успешной реализации к ним относятся: время и финансирование, инструментальное обеспечение, обеспеченность разработчиками. Точность полученных оценок является одним из определяющих факторов успешности выполняемого проекта.

Экономические исследования в сфере разработки информационных систем (ИС), в настоящее время очень актуальны. Существует множество публикаций на тему управления процессом разработки. Задачи оценки трудоёмкости и стоимости программного обеспечения (ПО), на всех стадиях жизненного цикла в полной мере в нашей стране не раскрыта.

К сожалению, современные методики решения данной задачи не достаточно эффективны. Существующие модели, обладающие достаточно хорошо проработанным алгоритмом решения, плохо адаптированы к реальному процессу проектирования ПО. Для различного рода крупных информационных систем данные методики делают задачу трудноразрешимой. К примеру, исследования, проведенные в Software Engineering Institute (SEI)[1] показал что, около 80% всех внедренных систем количественной оценки процесса разработки ПО оказываются практически невостребованными. Особую актуальность данные исследования имеют для Российских компаний разработчиков ПО, т.к. необходимо полностью пересматривать процесс управления разработкой ИС для внедрения данных решений на предприятиях.

[1] http://www.sei.cmu.edu - Software Engineering Institute.

Вместе с тем, возможности для совершенствования методических подходов к оценке трудоемкости проектирования ПО велики, рассматривая с позиции качественно нового подхода, путём объединения методов планирования и управления подобных систем.

Степень разработанности темы исследования. Вопросы оценки стоимости разработки ПО и исследования в этой области в основном проводятся на Западе, в частности, университетами США и Европы, на данный момент выполнено достаточно большое количество исследований в области программной инженерии. Начало исследований в области оценки стоимости программного обеспечения положено в 60-х г.г., в работах зарубежных (Нельсон, А. Альбрехт, Ч. Саймон, Б. Боэм, С. Макконнелл) и отечественных авторов (В.Липаев, С. Орлов). Существующие методы оценки ИС на практике оказываются трудоемкими, они недостаточно формализованы и не учитывают различные факторы, влияющие на длительность проекта.

Цель исследования. Целью исследования является разработка модели измерения трудоёмкости процесса разработки информационных систем на основе формализованной информации о предметной области.

Для достижения поставленной цели необходимо решить следующие задачи:

- Проведение комплексного анализа существующих моделей оценки стоимости программных продуктов, с целью выявления их основных характеристик, возможностей, сильных и слабых сторон;
- Разработка концепции управления разработкой программного обеспечения с учётом планирования и проведения оценки трудоемкости инициируемых программных проектов;
- Выбор модели и методики, на основе которой будет разрабатываться **новая комбинированная методика** по оценке стоимости разработки ИС;

- Разработка рекомендаций по практическому применению систем оценки стоимости с целью повышения точности оценок на ИТ-предприятиях;
- Разработка нового программного продукта (ПП) на основе комбинированной модели;

Объект и предмет исследования. Объектом исследования является процесс разработки крупных информационных систем.

Предметом исследования являются модели оценки трудоёмкости и соответствующих систем планирования и управления проектирования ИС.

Теоретическая и методологическая основа исследования. В основе диссертационного исследования лежат методы системного анализа процессов создания программного обеспечения, теория управления информационными проектами, метрики, экспертные оценки, анализ и синтез характеристик и показателей инициируемых проектов. В том числе специализированные методы и подходы к оценке трудоёмкости, ресурсоёмкости (методы аналогий) ряда крупных информационных компаний, Санкт-Петербурга и других регионов России.

Инструментальная поддержка разработанных методов заключается в применении средств, основанных на использовании программных пакетов MS Visual Studio 2008 (язык имплементации - C#), MS SQL, FirebirdSQL.

Научная новизна исследования. Научная новизна исследования заключается в разработке единого набора требований для моделей оценки стоимости и программных продуктов по оценке стоимости, с целью упрощения процесса выбора близкой к оптимальной модели трудозатрат для нужд ИТ-предприятия; в усовершенствовании моделей конструктивной стоимости (COCOMO II) и функциональных точек (Functional Point) с целью повышения точности оценок; в разработке нового программного продукта (пакета) для автоматизации получения оценок стоимости проекта,

позволяющего производить прогнозирование стоимости, сроков проекта с учетом изменяющихся требований заказчика, размера команды (персонала разработчиков), с возможностью адаптироваться под нужды конкретного предприятия, позволяющего хранить большой объем данных и получать быстрый доступ к ним.

Научная новизна диссертационной работы заключается в разработке новой экономико-математической модели и методики оценки трудоёмкости разработки ПО.

К числу основных результатов, определяющих научную новизну диссертационного исследования, относятся следующие:

1. сравнительный анализ моделей оценки стоимости на основе новых критериев, применяемых ведущими компаниями по разработке ПО, выделены их слабые и сильные стороны;
2. новая экономико-математическая модель оценки трудозатрат на различных этапах проектирования программного обеспечения, определения оптимального размера команды и система планирования;
3. концепция управления проектами с описанием процессами оценки трудозатрат инициируемых программных проектов, с целью повышения точности стоимостных факторов и параметров;
4. рекомендации для решения проблем неправильной оценки стоимости, учитывающие специфику методологии разработки ПО на ИТ-предприятиях;
5. автоматизированная система, которая выдаёт в виде отчётов оценку трудозатрат, на всех стадиях проектирования с различной глубиной анализа и детализации;

Практическая значимость работы. Создана и внедрена в ряде крупных ИТ-предприятии автоматизированная система оценивания и управления

процесса разработки ПО, на всех стадиях жизненного цикла информационных систем.

Публикации. Основные результаты и выводы диссертационной работы представлены в работах [1-5], опубликованных в научных изданиях, три из которых входит в список изданий, рекомендованных ВАК.

Научная апробация работы и внедрение результатов исследования. Основные результаты и выводы диссертационной работы докладывались автором на различных конференциях:

1. Общеуниверситетская научная конференция: «Социально-экономические тенденции в российском бизнесе». (СПб., СПбГУ, 27-29 марта 2008г.);
2. Весенняя конференция молодых учёных экономистов: «Пути развития национальной экономики». (СПб., СПбГУ, 18 апреля 2008г.) ;
3. Всероссийская конференция «V Всероссийская межвузовская конференция молодых учёных». (СПБ., СПБГУ ИТМО, 15-18 апреля 2008г.);
4. 14 международная конференция молодых ученых экономистов: «Предпринимательство и реформы в России». (СПБ., СПбГУ, 27-28 ноября 2008г.);
5. Политехнический симпозиум: «Молодые ученые - промышленности Северо-Западного региона». (СПБ., СПбГПУ, 9 декабря 2008г.);
6. IFPUG’s 4th Annual ISMA Conference and Fall Workshops Sunday, Chicago, IL, USA, 13-16 September 2009, Palmer House Hilton. “The combined technique of an estimation processes in the software development on the basis of COCOMO II and Functional Point.” Aliev Kh. Материалы опубликованы в общих тезисах конференции.
7. Результаты диссертационного исследования внедрены и опубликованы в н**аучно-исследовательской разработке**: «Разработка методологических основ проектирования и обоснование принципов применения корпоративных информационных систем в условиях инновационной

экономики» - **Код ГРНТИ: 20.15.05** СПбГУ, СПБ., 2009г. Руководитель: к.т.н., проф. Ботвин Г.А.

Результаты работы были интегрированы в состав практических методов на этапах предпроектного исследования при разработке заказных программных систем в таких ИТ-компаниях как:

- ✓ ООО «Эксиджен-Сервисес» (Exigen-Services) г. Санкт-Петербург,
- ✓ ЗАО «ЭЙСИ-Нильсен» (Nielsen Russia) г. Москва,
- ✓ ООО «Беволекс» (Bevolex) Республика Дагестан, г. Махачкала.

Структура и объем диссертации. Цели и задачи диссертационного исследования обусловили структуру диссертационной работы, которая состоит из введения, трёх глав, заключения, списка литературы и приложения. Общий объём основного текста содержит 151 стр., 28 таблиц, и 17 рисунков и схем., приложения содержит 55 стр.. Список использованных источников включает 119 наименований.

II. ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Во **Введении** обосновывается актуальность темы исследования, формулируется цель и задачи диссертационной работы, её теоретико-методологическая база и научная новизна.

В **Главе 1.** («Методики оценивания стоимости программных проектов») даны определения и понятия программного проекта (информационного проекта), определены основные признаки информационного проекта:

- ✓ Наличие чёткой цели проекта, определенных в техническом задании;
- ✓ Изменения предметной области, в которой реализуется программный проект;
- ✓ Ограниченность во времени, иными словами, сроки проекта;

- ✓ Ограниченность требуемых ресурсов, финансов, людей, техники, оборудования, материалов и др.;
- ✓ Комплексность и разграничение;
- ✓ Специальная организация проекта.

Проведена классификация проектов по составу, структуре и типу проектов, с учетом длительности, сложности и масштаба проекта. Схема классификации проектов приведена на рис.1

Рис.1. Схема классификации проектов

Основные затраты при разработке (создании) крупных информационных систем - это человеческие ресурсы, другими словами, это трудозатраты (трудоёмкость), в меньшей степени это ресурсо-затраты (ресурсоёмкость). С точки зрения экспертов по управлению проектами, трудозатраты на разработку ПО и есть эквивалент значения стоимости разработки ПО. Необходимо отметить, что на реальную стоимость выполнения программного продукта и окончательной цены ИС, оказывают влияние различные экономические факторы, которые непосредственно не относятся к программной инженерии ставка разработчика, ситуация на рынке и.т. д.

Основные факторы, влияющие на стоимость разработки ПО:

- **Размер конечного программного продукта,** который обычно измеряется числом строк исходного кода (SLOC);
- **Особенности процесса (методологии),** используемой для получения конечного программного продукта;
- **Возможности персонала (Personal capability),** участвующего в разработке ПО;
- **Среда разработки программного продукта,** состоящая из автоматизированных инструментов и методов, используемых для эффективного процесса разработки ПО;
- **Требуемое качество продукта (quality assurance),** включающая в себя функциональные возможности, производительность, надежность и адаптируемость и др.

В первой главе описываются особенности различных методов оценки трудоёмкости применяемых на сегодняшний день в различных компаниях, занимающихся разработкой крупных ИС. В контексте этих особенностей рассматривается эффективность, и определяются наиболее неэффективные методики оценивания.

Первая глава состоит из нескольких разделов, освещены вопросы предметной области, определен ряд требований к системам оценки и обоснована необходимость разработать новый информационный пакет, отвечающий требованиям, как исполнителей, так и заказчиков крупных информационных систем.

Методика оценки стоимости это чёткий математический алгоритм, используемый для того, чтобы оценить стоимость продукта (информационного проекта).

Во **второй главе** («Проектирование системы оценки стоимости программного проекта») даны теоретические основы диссертационного

исследования, в ней обосновывается эффективность применения комбинированного метода оценки трудоёмкости при проектировании и разработки ИС, приводятся модели, которые относятся к таким направлениям как, экспертные системы, методы аналогий, сложно формализуемые методики.

Вторая глава состоит из шести разделов, они посвящены разработке системы управления оценки стоимости на основе требований, выделенных в первой главе. В качестве базовой методики оценки трудозатрат выбраны две модели: COCOMO II (конструктивная модель стоимости) и Functional Point (Функциональные точки). Эти модели являются наиболее востребованными для оценки стоимости ИТ-проектов в крупных компаниях. В качестве доработки и дополнения данных моделей предложены следующие методики:

- Техника оценки PERT (Program Evaluation and Review Technique);
- Теория Брукса - классический концептуальный подход к управлению разработкой программных проектов.

Комбинированная методика COCOMO II и PERT. PERT - это способ анализа задач, необходимых для выполнения проекта. В особенности, анализа времени, которое требуется для выполнения каждой отдельной задачи, а также определение минимального необходимого времени для выполнения всего проекта. PERT был разработан в 50-ые годы главным образом для упрощения планирования и составления графиков больших и сложных проектов. Метод подразумевал наличие неопределённости, давая возможность разработать рабочий график проекта без точного знания деталей и необходимого времени для всех его составляющих. Самая известная часть PERT — это «Сети PERT» — графики соединённых между собой временных линий. PERT предназначен для очень масштабных, единовременных, сложных, нерутинных проектов.

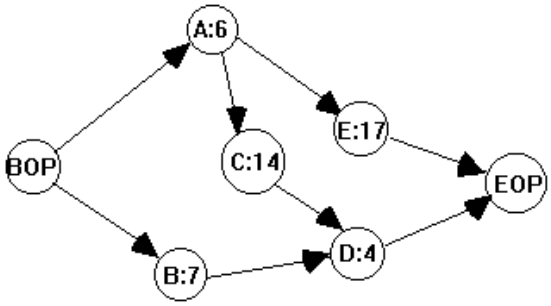

Рис. 2. Пример сети PERT.

Планирование процесса в классической PERT имеет следующие шаги:

1) Идентифицировать компоненты и итерации разработки проекта;
2) Определить соответствующую последовательность разработки компонентов;
3) Создать сетевой график на основе этой последовательности;
4) Определить время, необходимое для каждого компонента;
5) Выявить критический путь;
6) Обновлять сетевой график по мере продвижения в разработке проекта;

Итерации являются маркерами начала и конца групп компонентов разработки. Последовательность компонентов разрабатывается исходя из логической очереди в реализации проекта, в которой один компонент, например, не может быть начат раньше, чем завершен другой или некая группа компонентов. В то время как некоторые компоненты могут разрабатываться параллельно между собой. Время для каждого компонента разработки определяется 3 способами.

1) Оптимистическая оценка – с вероятностью 1% данный компонент потребует столько времени для реализации.
2) Номинальная – эта оценка имеет максимальную вероятность быть полученной на практике.
3) Пессимистическая оценка – максимальное время, которое может потребовать на реализацию компонент.

Критический путь в сетевой диаграмме – это самый длинный путь. Он определяет календарное время для выполнения проекта, остальные пути называются некритическими. Количество времени, на которое может быть задержан некритический путь без задержки всего проекта, называется пассивным временем. Считается, что время для компонентов критического путь имеет нормальное распределение, если количество этих компонентов достаточно велико.

В методе COCOMO II первоначальная оценка производится в строках кода. В PERT существует понятие компонент, который есть узел сетевого графика. В основе идеи предложенной автором диссертационной работы лежит в комбинации этих методов – замена единицы измерения реализации компонентов в PERT (время) на количество строк кода (соответственно оптимистическое, пессимистическое, наиболее вероятное). В теории PERT основным является критический (самый длинный) путь, который определяет время реализации проекта.

Оценка средней суммарной трудоемкости:

$$E_{LOC} = \sum_{i=1}^{n} E_{LOC_i} \quad \textbf{(1)}$$

, где

n – количество всех модулей необходимых для разработки всего проекта,

E_{LOC_i} – ожидаемое количество строк кода по пути для i-ого компонента на всём пути.

Ожидаемое количество строк кода по компоненту:

$$E_{LOC} = \frac{O_{LOC} + 4 \cdot N_{LOC} + P_{LOC}}{6} \quad \textbf{(2)}$$

, где

O_{LOC} (Optimistic estimation lines of code) – оптимистическая оценка количества строк кода по компоненту,

M_{LOC} (Most likely estimation lines of code) – наиболее вероятностная оценка количества строк кода по компоненту,

P_{LOC} (Pessimistic estimation lines of code) – пессимистическая оценка количества строк кода по компоненту.

По аналогии с техникой PERT , нам необходимо вычислить отклонение (Pert Deviation), для этого предлагается расчёт среднеквадратичного отклонения в строках кода по пути:

$$StDev_{LOC} = \sqrt{\sum_{i=1}^{n}\left(\frac{(P_{LOC_i} - O_{LOC_i})}{6}\right)^2} \quad \textbf{(3)}$$

, где

n – количество всех модулей необходимых для разработки всего проекта.

Миграция компонентов PERT в COCOMO II.

$$m\sum_{i=1}^{n} Sloc\,(i) = Sloc\,(P) \quad \textbf{(4)}$$

, где

m – коэффициент масштаба ожидания.

Масштабировать количество строк кода по каждому компоненту так, чтобы суммарное количество строк кода по проекту было равно ожидаемому с заданной вероятностью. Критический путь в строках кода поставляется на вход COCOMO II для обработки и оценки длительности проекта.

Надо еще раз заметить, что метод PERT создавался для крупных проектов. Из этого следует, что исполнитель обладает большими ресурсами для их реализации. Они достаточно велики, чтобы в расчете длительности проекта учитывать только критический путь, для определения бюджета

малых проектов необходимо учитывать сетку загрузки каждого разработчика на базе сетевого графика, являющегося последовательным планом проекта.

Одним из направлений данного исследования стало определение оптимального размера команды (Team Size), на базе классическом концептуальном подходе (Теории Брукса), учёта факторов коммуникаций внутри проектной команды. В уже существующие группы факторов функциональных точек (Functional Point) **были добавлены факторы определяющие уровень производительности**, (**Productivity Drivers**) – всего 7 факторов, для разных ролей в команде разработчиков проекта. Это подход позволяет оценить уровень каждого члена команды разработчиков индивидуально (тестера, программиста, дизайнера, документалиста, бизнес-аналитика, и.т.п.), производится оценка уровня производительности по выполняемому проекту.

При планирование разработки проекта производится построение планов разработки и тестирования. Определяется размер команды разработчиков и инженеров по качеству (тестеров) необходимый для выполнения разработки в оговоренные с заказчиком сроки. Посторенние планов проекта ведётся таким образом, чтобы свести риски по проекту к минимуму.

При формировании команды необходимо учитывать, сколько специалистов, какого уровня необходимо для успешного выполнения проекта в установленные сроки с выделенным бюджетом.

Факторы коммуникации членов команды:

- Насколько отдельные требования связаны между собой;
- Насколько производственные процессы, выполняемые разными людьми связаны между собой;
- Насколько быстро новый человек в проекте способен «втянуться» в процесс и начать работать на общем уровне производительности.

Длительность проектов с учётом коммуникационных затрат будем иметь следующий вид:

$$Duration = E_0 * \frac{1 + LO * RI * (TS - 1) + CO * RI * TS * (TS - 1)}{TS} \quad (5)$$

, где

E_0 – трудозатраты без учета коммуникационных издержек, человеко-дни (man-days)

TS – Размер команды

LO – Обучаемость (Learning Overheads), %

CO – Коммуникации (Communication Overheads), %

RI – Влияниние требований (Requirements Interference), %

NI – Number of Interdependencies показывает, каково количество связей между отдельными требованиями. Каждая связь означает невозможность реализации заданного требования без учета особенностей реализации другого требования, то есть без обсуждения деталей между членами команды.

$RI = NI/NI_{max} * 100\%$ – процент взаимозависимости требований

RI*TS*(TS-1) – количество актов коммуникации.

Учёт различных ролей в модели имеет следующий вид:

$$DG = \frac{UEF}{\max(UEF_development, UEF_testing)} \quad (6)$$

, где UEF - удельная трудоемкость на 1 функциональную точку в часах.

UEF_development - удельная трудоемкость разработки 1 FP,

UEF_testing - удельная трудоемкость тестирования 1 FP.

Модель COCOMO II условно разделяют на три класса проекта – ограниченные, полу-интегрированные, встроенные системы.

Ограниченные проекты – относительно маленькие и разрабатываются командами, знакомыми с прикладной областью, с менее жёсткими требованиями.

Полу-интегрированные проекты – (промежуточные) системы среднего размера и сложности, разрабатываемые группами разработчиков с различным опытом, требования к ПО более жёсткие.

Проекты «встроенных систем» - выполняются при значительных аппаратных, программных и организационных ограничениях, ограничения по жёсткой спецификации.

Основа COCOMO II - стоимость разработки программного обеспечения усилий (и затрат) в зависимости от размера программы, выраженные в соответствии с оценками размера кода.

В модели определены 15 поправочных факторов, принадлежащих к одной из четырех категорий: атрибуты продукта, такие, как его сложность и требования к его надежности; атрибуты системы, такие, как ограничения на оперативную память и время выполнения; атрибуты команды, такие, как опыт в прикладной области; и атрибуты проекта, такие, как используемые средства разработки.

Модель COCOMO II существует в трех видах. Выбор того или иного вида модели для оценки стоимости программного обеспечения зависит от типа проекта и стадии разработки.

Первый вид модели, модель композиции приложения, включает в себя оценку прототипа пользовательского интерфейса. Она используется на ранних стадиях разработки проекта.

Модель раннего проектирования включает в себя изучение альтернативных архитектур и концепций работы. На этой стадии недостаточно общего описания проекта, требуются детали.

Модель пост-архитектуры связана с реальной разработкой и эксплуатацией программного продукта. Эта модель работает наиболее эффективно, если разработана архитектура жизненного цикла, проверено ее соответствие миссии компании, концепции взаимодействия, рискам, определена структура компонентов разрабатываемого продукта.

Таблица 1. Модели, входящие в состав COCOMO II

Наименование	Формула
Модель композиции приложения	**ЗАТРАТЫ = NOP/PROD [чел.-мес.]**, где NOP=(Объектные указатели)x[(100-%REUSE)/100] — количество новых объектных указателей; PROD = f_T (опытность/возможности разработчика; зрелость/возможности среды разработки) — производительность/скорость разработки, выраженная в терминах объектных указателей
Модель раннего этапа проектирования	**ЗАТРАТЫ =AxРАЗМЕР^{B}x M_e + ЗАТРАТЫ$_{auto}$ [чел.-мес.]**, где A =2,5 — масштабный коэффициент; РАЗМЕР — размер системы в тыс. LOC; B = f_T (PREC, FLEX, RESL, TEAM, PMAT) — 5 характеристик: PREC -опыт подобных разработок, FLEX - определенность предполагаемого процесса разработки, RESL - степень выполняемого анализа риска, TEAM - слаженность команды, PMAT -зрелость процесса организации; M_e = f_T (PERS, RCPX, RUSE, PDIF, PREX, FCIL, SCED) — 7 характеристик: PERS – возможности персонала RCPX – надежность и сложность продукта RUSE – требуемое повторное использование PDIF – трудность платформы PREX – опытность персонала FCIL – средства поддержки SCED - график ЗАТРАТЫ$_{auto}$ —затраты на автоматическую генерацию кода, если таковая имеется
Модель этапа пост-архитектуры	**ЗАТРАТЫ =AxK$_{-req}$xРАЗМЕР^{B}xM$_p$+ ЗАТРАТЫ$_{auto}$ [чел.-мес.]**, где A =2,5 — масштабный коэффициент; K_{-req} = 1+(BRAK/100) — коэффициент, учитывающий возможные изменения в требованиях BRAK — процент отброшенного кода РАЗМЕР = РАЗМЕР$_{new}$ + РАЗМЕР$_{reuse}$ — размер системы в тыс. LOC; B = f_T (PREC, FLEX, RESL, TEAM, PMAT) — 5 характеристик PREC -опыт подобных разработок, FLEX - определенность предполагаемого процесса разработки, RESL - степень выполняемого анализа риска, TEAM - слаженность команды, PMAT -зрелость процесса организации); M_p = f_T (RELY, DATA, CPLX, REUSE, DOCU; TIME, STOR, PVOL; ACAP, PCAP, AEXP, PEXP, LTEX, PCON; TOOL, SITE, SCED) — 17 характеристик **Факторы продукта:** RELY – требуемая надежность ПО DATA – размер базы данных CPLX – сложность продукта REUSE – требуемая повторная используемость DOCU – документирование требований жизненного цикла Факторы платформы (виртуальной машины): TIME – ограничения времени выполнения

	STOR – ограничения памяти PVOL – изменчивость платформы Факторы персонала: ACAP – возможности аналитика PCAP – возможности программиста AEXP – опыт работы с приложением PEXP – опыт работы с платформой LTEX – опыт работы с языком и утилитами PCON – непрерывность персонала Факторы проекта: TOOL – использование программных утилит SITE –мультисетевая разработка SCED – требуемый график разработки $ЗАТРАТЫ_{auto}$ —затраты на автоматическую генерацию кода, если таковая имеется

Методология **функциональных точек** в оценке разработки программного обеспечения. Задача метода разбить систему на мелкие части так, чтобы они могли быть лучше понимаемы и анализируемы. Это обеспечивает более структурированный подход для решения проблемы. **Функциональные точки** – это своего рода подход измерения размера и сложности ПП. Программное приложение, в сущности, определяется набором элементарных процессов. Существует два основных типа элементарных процессов – данные в движении и данные в покое. Данные в движении описывают потоки данных из приложения вовне и извне в приложение. На концептуальном уровне анализ функциональных точек помогает определить два абстрактных уровня данных – в покое и данных в движении.

Формула, определяющая продуктивность программного обеспечения:

Продуктивность = Количество функциональных точек / количество инпутов (за период времени, с учетом качества).

Уравнение коэффициента подстройки:

$$VAF = 0.65 + \left[\left(\sum_{i=1}^{14} C_i \right) / 100 \right]$$

Где C_i – влияние каждой характеристики системы (всего 14 характеристик).

Описание общих характеристик системы:

1. Связи данных. Как много связей используются для передачи данных внутри системы.

2. Распределенная обработка данных. Как обрабатываются распределенные данные.

3. Какие требования по производительности.

4. Какие требования по аппаратной части.

5. Как часто выполняются транзакции (ежедневно, еженедельно …)

6. Какой процент информации вводится онлайн.

7. Эффективность для конечного пользователя.

8. Как много ILF обновляются онлайн.

9. Используются ли сложная логическая или математическая обработка.

10. Была ли спроектирована программа для одного или многих пользователей.

11. Насколько сложна инсталляция.

12. Насколько эффективны или автоматизированы старт, резервное копирование и восстановление.

13. Было ли приложение спроектировано, разработано для множества сотрудников множества организаций.

14. Было ли приложение спроектировано для упрощения дальнейших изменений в нем.

По каждому пункту используется пятибалльная система оценки.

В работе предложена **комбинированная методика COCOMO II и Functional Point**, каждая из перечисленных характеристик, таких как распределенные функции, требования к производительности и переиспользованию, например, имеют максимальный вклад 5% в оценку трудоемкости. Это противоречит модели COCOMO II. Поэтому в рамках COCOMO II нельзя использовать подстроечный коэффициент для расчета количества функциональных точек. Методика подсчета функциональных точек для COCOMO II следующая:

Обозначения:

ILF – внутренний логический файл (Internal Logical File)

EIF - внешний интерфейсный файл (External Interface File)

EI - внешний входной элемент (External Input)

EO – внешний выходной элемент (External Output)

EQ – внешний запрос (External Inquery)

Таблица 2. Определения уровня сложности.

Для ILF и EIF				Для EQ и EO				Для EI			
Кол-во записей	Кол-во элементов данных			Кол-во типов файлов	Кол-во элементов данных			Кол-во типов файлов	Кол-во элементов данных		
	1-19	20-50	51+		1-5	6-19	20+		1-4	5-15	16+
1	L	L	A	0 или 1	L	L	A	0 или 1	L	L	A
2-5	L	A	H	2-3	L	A	H	2-3	L	A	H
6+	A	H	H	4+	A	H	H	3+	A	H	H

L – низкий уровень сложности

A – средний уровень сложности

H – высокий уровень сложности

Таблица 3. Вес-сложность для подсчета количества UFP

Тип функции	L	A	H
ILF	7	10	15
EIF	5	7	10
EI	3	4	6
EO	4	5	7
EQ	3	4	6

$$UFP = \sum_{i=1}^{n} ILF_i + \sum_{i=1}^{n} EIF_i + \sum_{i=1}^{n} EI_i + \sum_{i=1}^{n} EO_i + \sum_{i=1}^{n} EQ_i$$

Существует среднестатистическое отношение между количеством UFP и SLOC (кол-вом строк кода) для каждого языка имплементации. Если в

проекте комбинируются несколько языков, то для преобразования UFP->SLOC можно воспользоваться формулой:

$$SLOC = \sum_{i=1}^{n} UFP_i \cdot sloc_per_fp_i \text{ ,} \qquad \textbf{(7)}$$

где UFP_i – количество функциональных точек на i язык имплементации, $sloc_per_fp_i$ – усредненное приблизительное количество строк кода для языка i на функциональную точку, примеры, которых даны в таблице 4.

Таблица 4. Количество строк кода на функциональную точку.

Язык п-я	Sloc_per_fp
C#	55
C++	55
JAVA	55
SQL	13
Perl	20
HTML	15

Определяем оптимистическую и пессимистическую оценки для 3 моделей в зависимости от полученной трудоемкости:

Таблица 5. Оценки для трех моделей.

Модель	Оптимистическая оценка	Пессимистическая оценка
Построение прототипа (Application Composition)	0.50	2.00
Модель раннего дизайна (Early Design Model)	0.67	1.50
Модель пост-архитектуры (Post Architecture Model)	0.80	1.25

С течением проекта во времени диапазон оптимистических и пессимистических оценок сужается, в пределе к фактической длительности проекта. Поскольку накапливается больше информации, и уточняются оценки, в процессе реализации проекта, модель может калиброваться для пересмотра первоначальной оценки.

В **заключительной, третьей главе** диссертационного исследования («Практическое применение и программная реализация методов и моделей оценки стоимости разработки программного обеспечения») приведены результаты использования предлагаемой автором методики и автоматизированной системы процессе управления производством программного обеспечения в ряде крупных проектов ИТ-предприятий.

В результате последовательной реализации всех этапов методики, автор доказал целесообразность выбора именно комбинированных методов анализа оценки трудозатрат, с учётом специфики производства ПП и применяемых методологий этих компаний. Также в третьей главе приведен условный пример применения разработанной методики.

В **Заключении исследования** сформулированы основные теоретические и практические выводы, сделанные автором диссертационного исследования.

III. ПУБЛИКАЦИИ АВТОРА ПО ТЕМЕ ДИССЕРТАЦИИ

1. Алиев Х.Р. Эффективная модель оценки разработки программного обеспечения // Исследовано в России - МФТИ – М., Том 11, №30, 2008г. 0,8 п.л. **http://zhurnal.gpi.ru/articles/2008/030.pdf**
2. Алиев Х.Р. IT-аутсорсинг в России, оценка контрактов // Российский экономический интернет журнал – М., 2010г. 0,6 п.л. **http://www.e-rej.ru/Articles/2010/Aliev.pdf**

Статьи в рецензируемых научных журналах, рекомендованных ВАК:

1. Алиев Х.Р. Комбинированная модель оценки трудоемкости разработки программного обеспечения // Научно-технические ведомости СПБГПУ (серия: экономические науки) – Спб., Издательство СПБГПУ, №3, 2010 г. 0,4 п.л.
2. Алиев Х.Р., Андржевский С.О., Борисов М.И. Модели оценки стоимости информационных систем в классической и перспективных методологиях разработки программного обеспечения // Прикладная информатика – М., издательство МАРКЕТС ДС, №5(23), 2009г. 1,08 п.л. (вклад автора – 0,6 п.л.)
3. Алиев Х.Р. К проблеме моделирования стоимости разработки информационной системы // Обозрение прикладной и промышленной математики – М., 2009, том 16, вып. 2. 0,1 п.л.