Автоматизация работы врача терапевта в городской поликлинике на основе спиралевидной модели внедрения информационных систем (Часть 1)

Дудкин Дмитрий Игоревич

Аннотация: в статье описывается разработка программного обеспечения для использования на рабочем месте врача-терапевта. Программное обеспечение позволит сохранять всю информацию о пациенте, его посещениях, историю болезни, облегчить и ускорить работу врача. Разрабатываемая система исключает проблему потери амбулаторных карт пациентов, так как вся информация будет храниться в одной базе данных.

Введение

За последние годы компьютерные технологии сильно развиваются и внедряются во все возможные сферы жизни человека. Они имеют большой перечень функциональных возможностей, в том числе и возможность повысить эффективность работы врача лечебного учреждения.

В недалеком прошлом лечебные учреждения: поликлиники и работающие в них врачи, имели определенные затруднения при приеме пациентов. Например, очень долго и трудоемко проходил процесс документального оформления приема пациентов, актуальными были проблемы потери карты пациента в регистратуре и поиска её у других врачей.

Все это приводило к образованию очереди пациентов на прием к врачу, затруднялась работа врачей, медицинского персонала и в целом работа лечебного учреждения. В связи с чем остро назрела задача рационализировать механизм документального оформления осмотра больных и приема пациентов в лечебном учреждении, что послужило стимулом к широкому внедрению компьютерных технологий в область здравоохранения. Врачам необходимо, чтобы технические средства и разработанное к ним программное обеспечение были простыми и удобными при использовании, не нарушали привычного стиля работы.

Главное проблемой, с которой сталкиваются врачи при приёме пациента – это недостаток времени. Значительное время врача при приеме расходуется на ознакомление с медицинской картой обследуемого. Разработка медицинской базы данных и ее использование в работе врача позволит ускорить данный процесс, более качествен-

но проводить обследование и снизить нагрузку на врача, даст возможность быстрого поиска в базе данных нужной информации о пациенте. Например, посмотреть всю историю болезни, даты приема, проведенные обследования и их результаты и др.

Разработку такого приложения можно произвести в программе MS Access. Данная среда совместима с операционной системой Microsoft Windows, которая используется на большинстве компьютеров медицинских учреждений, что важно для удобства использования.

Цель и задачи

Целью данной работы является разработка программного обеспечения для использования на рабочем месте врача-терапевта в виде автоматизированной системы. Программное обеспечение позволит сохранять всю информацию о пациенте, его посещениях, историю болезни, облегчить и ускорить работу врача. Благодаря данной системе снимается проблема с потерей амбулаторных карт пациентов, так как вся информация будет храниться в одной базе данных. Чтобы достичь вышеуказанной цели нам нужно реализовать следующие задачи:

- произвести детальный анализ спиралевидной методологии внедрения систем;
- идентифицировать требования и сформулировать список требований;
- произвести проектирование процессов и оргструктуры в моделях AS-IS и TO-BE нотации ARIS VACD и UML AD до 3-4 уровней детализации;
- моделирования разрабатываемых пользовательских интерфейсов;
- проектирования структуры данных и нормализация таблиц данных.

В спиралевидной методологии внедрения систем для каждого витка спирали произвести:

- реализация операции ключевого процесса в среде MS Access;
- тестирование и количественную оценку результатов тестирования.

Разработанная база данных позволит упорядочить и систематизировать работу врача-терапевта. А это, в свою очередь, может значительно повысить качество работы, ускорить приём пациентов и соответственно решить проблему с очередями на прием к врачу.

1. Идентификация требований

Проблемы, которые могут возникнуть при создании программного обеспечения для автоматизации какого-либо процесса, не всегда можно понять. Очень трудно чётко описать те действия, которые должна выполнять система. Описание функциональных возможностей и ограничений, накладываемых на систему, называется требованиями. Требования к продукту должны быть установлены таким образом, что могло бы гарантировать их адекватность и верный «перевод» с языка пользователя. Для начала необходимо выявить все возможные требования, предъявляемые к разрабатываемому программному продукту.

Выявить их можно с помощью анализа требований – часть процесса разработки программного обеспечения, включающая в себя сбор требований к программному обеспечению (ПО), их систематизацию, выявление взаимосвязей, а также документирование. В процессе сбора требований важно принимать во внимание возможные противоречия различных заинтересованных лиц, таких как: заказчики, разработчики или пользователи. Процесс формирования и анализа требований проходит через ряд этапов:

- Анализ предметной области.
- Сбор требований.
- Назначение приоритетов.

1.1. Анализ предметной области

Терапевт – специалист в области терапии, врач, специализирующийся на выявлении, лечении, профилактике внутренних болезней. Он осуществляет первичную диагностику, координирует взаимодействие пациента с остальными специалистами, выписывает направления на большинство обследований и процедур, оформляет медицинскую документацию. Основные задачи этого специалиста:

- первичный прием пациентов, сбор анамнеза, проведение осмотра и других объективных методов обследования;
- ранняя диагностика на основании результата обследования пациента и анализа его жалоб;
- первичная консультация, разъяснение пациенту причин его недуга;
- назначение лекарств, физиотерапевтических процедур и других лечебных мероприятий в пределах своей компетенции;
- назначение лабораторных анализов и инструментального обследования;

- в случае осложненного течения или неясного генеза заболевания – направление к профильному специалисту для более детальной диагностики и прохождения лечения в соответствии с его рекомендациями;
- разработка единой схемы лечения с учетом рекомендаций разных узких специалистов;
- принятие решения о госпитализации;
- оценка риска развития хронического заболевания и принятие мер к его снижению;
- консультации относительно укрепления иммунитета, профилактики осложнений, рецидивов и перехода заболевания в хроническую форму;
- регулярное наблюдение пациентов с хроническими заболеваниями;
- разработка рекомендаций относительно изменения образа жизни, условий труда, санаторно-курортного лечения и прочих;
- назначение схемы комплексного медицинского обследования при прохождении профосмотра, медкомиссии;
- осмотр перед вакцинацией и принятие решение относительно ее проведения.

В дальнейшем мы будем работать с такими ключевыми бизнес-процессами, как: приём пациента, лечение пациента, реабилитация пациента.

1.2. Пользовательские и функциональные требования, их приоритизация

Ниже дан список выявленных пользовательских требований, обеспечивающих:

- хранение данных о пациенте (Фамилия, Имя, Отчество, дата рождения и т.д.);
- хранение данных о записи на приём (когда и сколько раз был на приёме, по какой причине);
- хранение данных об истории лечения пациента (какие лекарства были назначены, направление на анализы);
- хранение данных о реабилитации пациента (какими лекарствами происходило лечение, помогли ли они; если не помогли, назначение новых препаратов);
- сведения, полученные путём расспроса пациента (обследуемого);
- управление данными о записи на прием (изменение, удаление, добавление, сортировка);
- управление данными, полученными путём расспроса пациента (изменение, удаление, добавление, сортировка);
- управление данными о лечении пациента (изменение, удаление, добавление, сортировка);

- разграничение доступа (для того, чтобы защитить данные пациента);
- разработку интерфейса (для ускорения приема пациентов).

Следующим шагом выявим функциональные требования (functional requirements), определяющие возможности разрабатываемого ПО для реализации пользовательских требований. Выделим подобные требования для наших ключевых бизнес-процессов:

- база данных, содержащая таблицы «Личные данные пациентов», «Даты посещений», «Причины посещения», «Лечение пациента», «Направление на анализ», «Анализ мочи», «Анализ кала», «Анализ крови», «Реабилитация пациента»;
- вывод на экран данных из вышеперечисленных таблиц;
- добавление данных о пациенте;
- редактирование данных о пациенте во всех таблицах;
- поиск конкретного пациента по ФИО во всех таблицах;
- программа должна корректно работать на всех компьютерах медучреждения;
- данные хранятся непосредственно в создаваемом приложении;
- установление пароля на саму базу данных;
- разработка интерфейса для простоты использования продукта.

Приоритезация требований позволяет понять, какие требования пользователю необходимо реализовать в первую очередь и на что следует обратить особое внимание. Это позволит избежать излишних материальных и временных затрат на проектирование и разработку модулей или функционала. Расставим приоритеты для ранее представленных пользовательских и функциональных требований:

- приоритет №1 – данные требования является основой разрабатываемой системы. Без выполнения данного пункта реализовать систему не получится;
- приоритет №2 – критичные требования для функциональных возможностей системы, но не сильно отражающиеся на работе программы при их отсутствии;
- приоритет №3 – требования не являются ключевыми для полноценного функционирования самой программы, но их было бы неплохо реализовать.

1.3. Список требований

После сбора и анализа пользовательских и функциональных требований создается реестр требований. Он позволяет связать и сопоставить все требования, их приоритеты и программные компоненты, отвечающие за реализацию требований. Список требований облегчает процесс отслеживания требований разработчиком и позволяет

удостовериться в их полном выполнении перед окончанием работ. Требования для разрабатываемого продукта представлены в Таблице 1.1.

Таблица 1.1 – Список требований для разрабатываемого продукта

№	Пользовательские требования	Функциональные требования	Программный компонент	Приоритет требования
1	Хранение данных о пациенте	Таблица данных «Личные данные пациентов»	Программа по введению данных	Приоритет №1
2	Хранение данных о записи на прием	Таблица данных «Даты посещений»		
3	Сведения, полученные путём расспроса пациента	Таблица данных «Причины посещения»		
4	Хранение данных об истории лечения пациента	Таблица данных «Лечение пациента»		
5	Хранение данных о том, на какой анализ был направлен пациент	Таблица данных «Направление на анализ»		
6	Хранение данных о результатах анализа мочи	Таблица данных «Анализ мочи»		
7	Хранение данных о результатах анализа кала	Таблица данных «Анализ кала»		
8	Хранение данных о результатах анализа крови	Таблица данных «Анализ крови»		
9	Хранение данных о том, как происходит реабилитация пациента	Таблица данных «Реабилитация пациента»		
10	Управление данными о записи на прием (изменение, удаление, добавление, сортировка)	Возможность редактирования, удаления, добавления, поиска и сортировки записей в таблице данных «Даты посещений»	Программа для добавления, редактирования, удаления, поиска, сортировки и просмотра данных	Приоритет №2
11	Управление данными, полученными путём расспроса пациента (изменение, удаление,	Возможность редактирования, удаления, добавления, поиска и сортиров-		

№	Пользовательские требования	Функциональные требования	Программный компонент	Приоритет требования
	добавление, сортировка)	ки записей в таблице данных «Причины посещения»		
12	Управление данными о лечении пациента	В таблице данных «Лечение пациента»		
13	Управление данными о направлении на анализ	В таблице данных «Направление на анализ»		
14	Управление данными об анализе мочи	В таблице данных «Анализ мочи»		
15	Управление данными об анализе кала	В таблице данных «Анализ кала»		
16	Управление данными об анализе крови	В таблице данных «Анализ крови»		
17	Управление данными о реабилитации пациента	В таблице данных «Реабилитация пациента»		
18	Разграничение доступа	Установление пароля на саму базу данных	Программа авторизации пользователя	Приоритет №3
19	Разработка интерфейса	Разработка интерфейса	Программа упрощения работы пользователя	

1.4. Задание циклов разработки по спиралевидной модели

Разработка программного продукта согласно спиралевидной модели будет производиться в следующем порядке:

- Начало:
  - анализ требований (19 требований – см. табл. 1.1);
  - моделирование ключевых бизнес-процессов с помощью нотаций ARIS VACD и UML AD;
  - моделирование данных и интерфейсов программ;
  - составление плана разработки по спиральной модели.
- 1-й виток спирали:

- планирование текущего цикла разработки (реализовать требования 1-9 в среде MS Access);
- моделирование данных и интерфейсов программ;
- реализация требований 1-9 в среде MS Access;
- тестирование реализованных возможностей программы;
- демонстрация прототипа программы заказчику.

- 2-й виток спирали:
  - планирование текущего цикла разработки (реализовать требования 10-17 в среде MS Access);
  - моделирование данных и интерфейсов программ;
  - реализация требований 10-17 в среде MS Access;
  - тестирование реализованных возможностей программы;
  - демонстрация прототипа программы заказчику.
- 3-й виток спирали:
  - планирование текущего цикла разработки (реализовать требование 18-19 в среде MS Access);
  - реализация требований 18-19 в среде MS Access;
  - тестирование реализованных возможностей программы;
  - подготовка к релизу (исправление мелких недостатков, ошибок и т.д.);
  - тестирование конечного продукта;
  - релиз конечного продукта.

2. Проектирование ключевых бизнес-процессов

Бизнес-процесс – это устойчивая целенаправленная последовательность исполнения функций, направленная на создание результата, имеющего ценность для потребителя [1]. При проектировании ключевых бизнес-процессов используют две модели: AS-IS (как есть) и TO-BE (как будет):

- модель AS-IS, которая описывает состояние моделируемой предметной области на текущий момент;
- модель TO-BE, описывающая возможное будущее состояние предметной области, в которое она перейдёт в результате внедрения новых технологий [2].

Прежде чем пытаться выбрать существующую или создать собственную информационную систему, требуется проанализировать, как работает система на данный момент времени. После этого строится функциональная модель AS-IS. Анализ этой модели позволит выявить недостатки и понять, в чем будут состоять преимущества новых бизнес-процессов. В модели TO-BE как раз есть возможность исправление та-

ких недостатков. Данная модель нужна для оценки последствий внедрения информационной системы и анализа альтернативных путей выполнения работы и документирования того, как система будет функционировать в будущем.

2.1. Проектирование на основе нотации ARIS VACD

Преимущества методологии ARIS заключаются в ее комплексности, которая проявляется во взаимосвязи моделей, построенных в различных нотациях. Методология ARIS позволяет описывать деятельность организации с разных точек зрения, при этом полученные модели в определенной степени связаны между собой [3]. Одной из важнейших нотаций ARIS является нотация Value-added Chain Diagram. Она используется для описания бизнес-процессов организации на верхнем уровне.

Главным отличием данной модели от других процессных моделей является то, что информационные и материальные потоки на схеме VACD изображаются не стрелками, а объектами. При этом для каждого типа потока используется свой объект. В модели VACD в отличие от классического подхода также используются логические связи между работами, которые позволяют отобразить логическую последовательность выполнения работ.

Данная нотация подходит для изучения организации в целом. Но для выявления возможных проблем этой нотации может быть недостаточно, она помогает определить те процессы, которые и нуждаются в изменениях и доработке. При описании бизнес-процессов воспользуемся приложением ARIS Express.

2.2. Проектирование на основе нотации UML Activity Diagram

При моделировании поведения проектируемой системы появляется необходимость представить процесс изменения ее состояний и детализировать особенности алгоритмической и логической реализации выполняемых системой операций. Обычно для этого использовались блок-схемы или структурные схемы алгоритмов. Блок-схема — распространенный тип схем, описывающих алгоритмы или процессы, в которых отдельные шаги изображаются в виде блоков различной формы, соединенных между собой линиями, указывающими направление последовательности. В данной работе был выбран язык UML и нотация, схожая с блок=схемой, Activity Diagram.

Язык UML (Unified Modeling Language), или унифицированный язык моделирования, предназначен для описания, визуализации, проектирования и документирования

объектно-ориентированных систем и бизнес-процессов с ориентацией на их последующую реализацию в виде программного обеспечения [4].

Для моделирования процесса выполнения операций в языке UML используются диаграммы деятельности или Activity Diagram. Они необходимы для моделирования последовательности действий, которые выполняются различными элементами, входящими в состав системы. Этот вид диаграмм раскрывает детали алгоритмической реализации операций, выполняемых системой, поэтому диаграмма деятельности похожа на обычную блок-схему [5]. Но в отличие от блок-схемы, диаграмма деятельности также показывает одновременно параллельную и последовательную деятельность.

2.3. Проектирование процессов в моделях AS-IS и TO-BE с помощью UML AD и ARIS VACD

На рисунках 2.1-2.2 рассматривается первый уровень проектирования процесса работы врача-терапевта в аннотации ARIS VACD и моделях «AS-IS», которая подразумевает под собой проектирование процессов в настоящий момент времени и «TO-BE» – после внедрения электронной базы данных.

Для того чтобы создать программное обеспечение, требуется больше информации, поэтому необходимо провести более подробное проектирование. Для этого следует произвести второй и третий уровень детализации. На рисунках 2.3-2.4 представлен второй уровень детализации для уточнения процесса «Прием пациента».

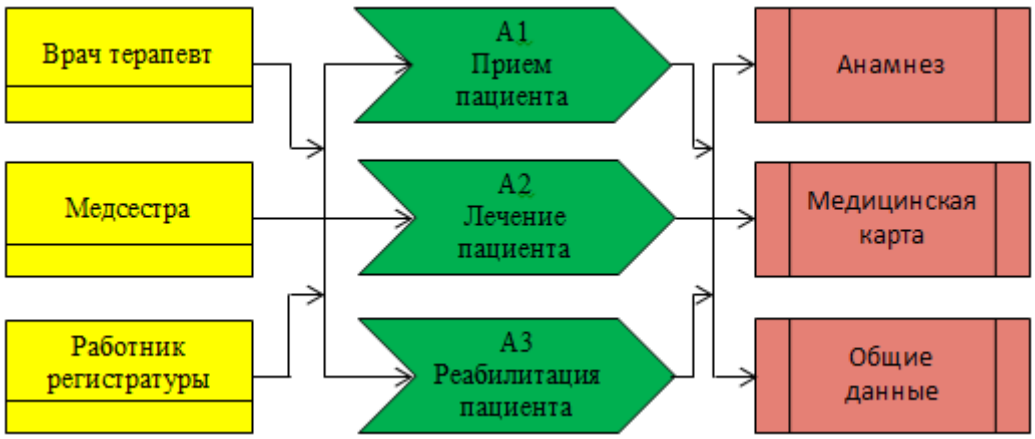

Рис. 2.1. – *Представление процесса в ARIS VACD на первом уровне в модели «AS-IS»*

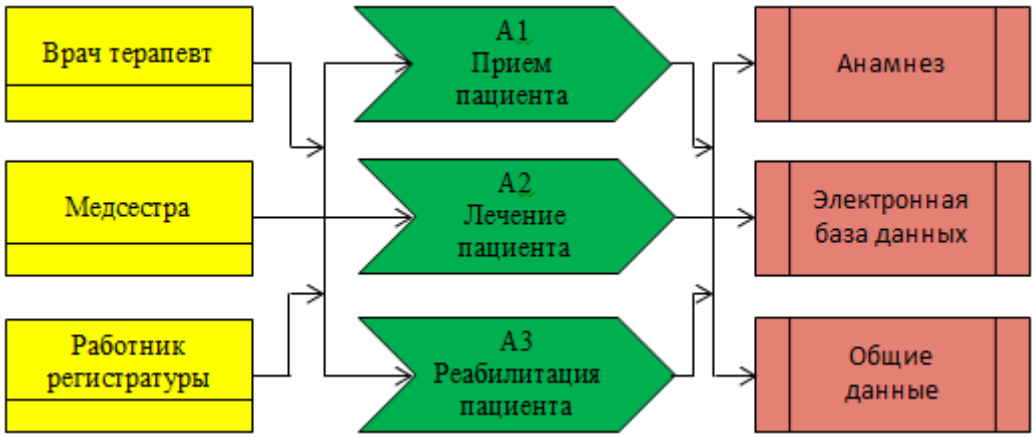

Рис. 2.2. – *Представление процесса в ARIS VACD на первом уровне в модели «TO-BE»*

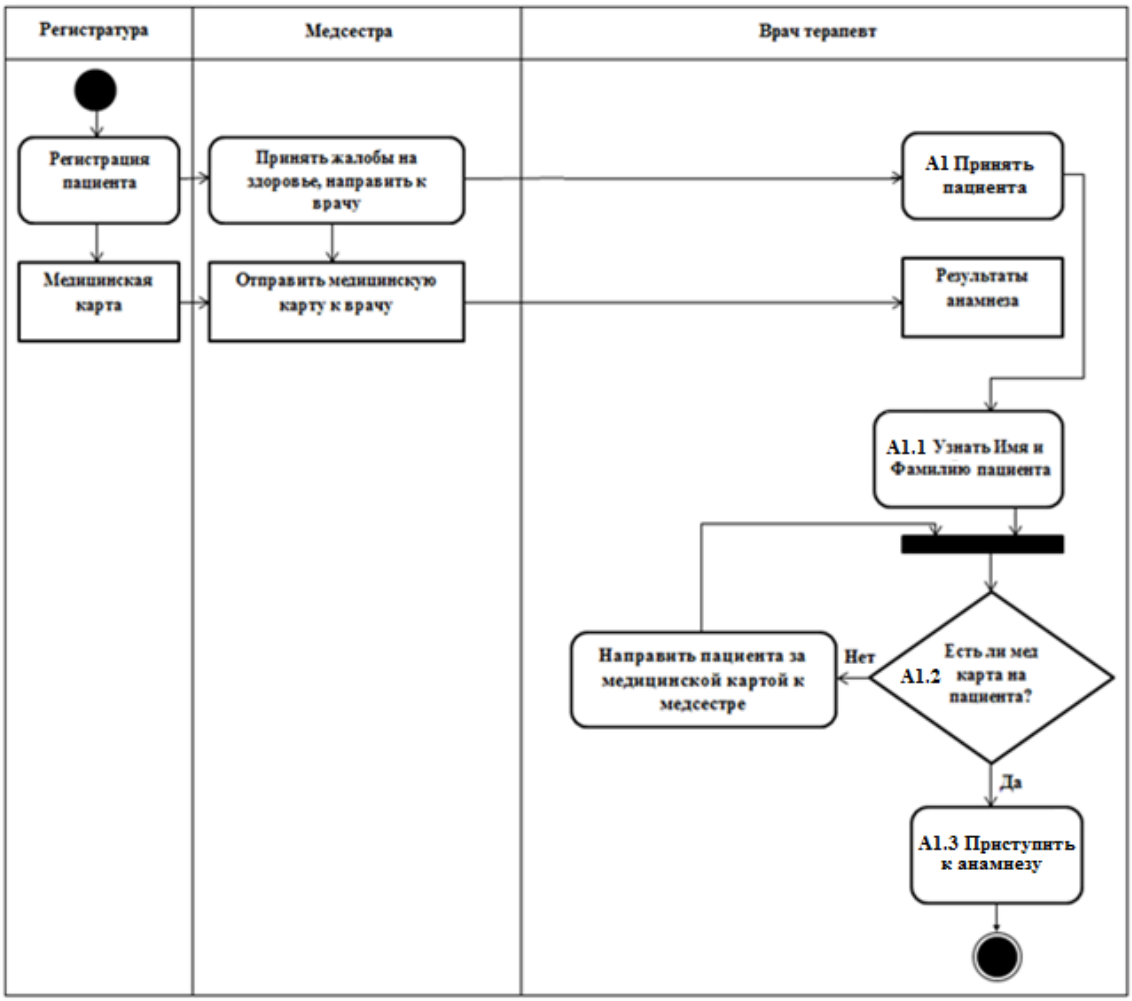

Рис. 2.3. – *Представление подпроцесса в UML AD на втором уровне в модели «AS-IS» для уточнения процесса «Прием пациента»*

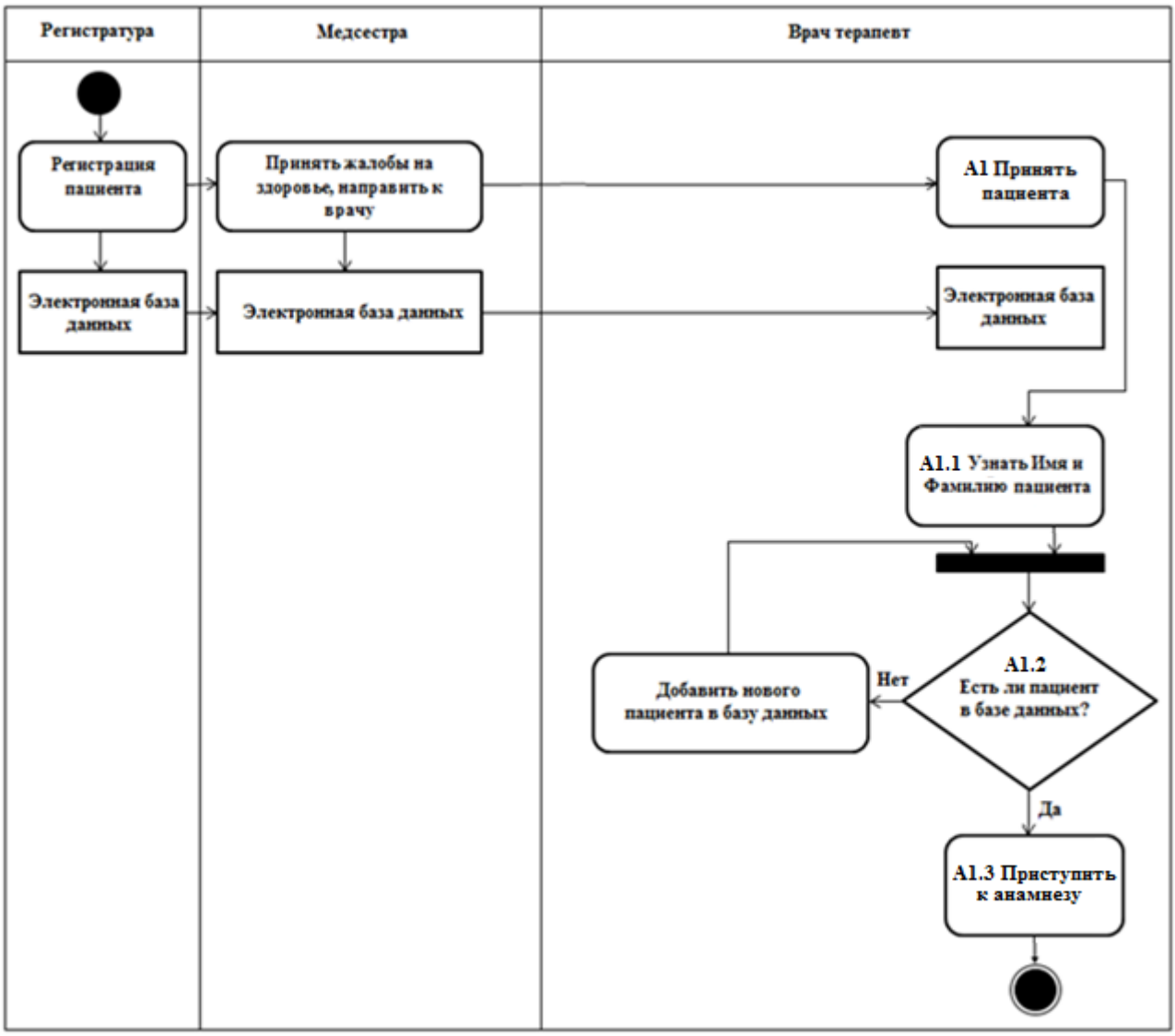

Рис. 2.4. – Представление подпроцесса в UML AD на втором уровне в модели «TO-BE» для уточнения процесса «Прием пациента»

Далее на рис. 2.5-2.6 дан третий уровень детализации для уточнения процесса «Приступить к анамнезу».

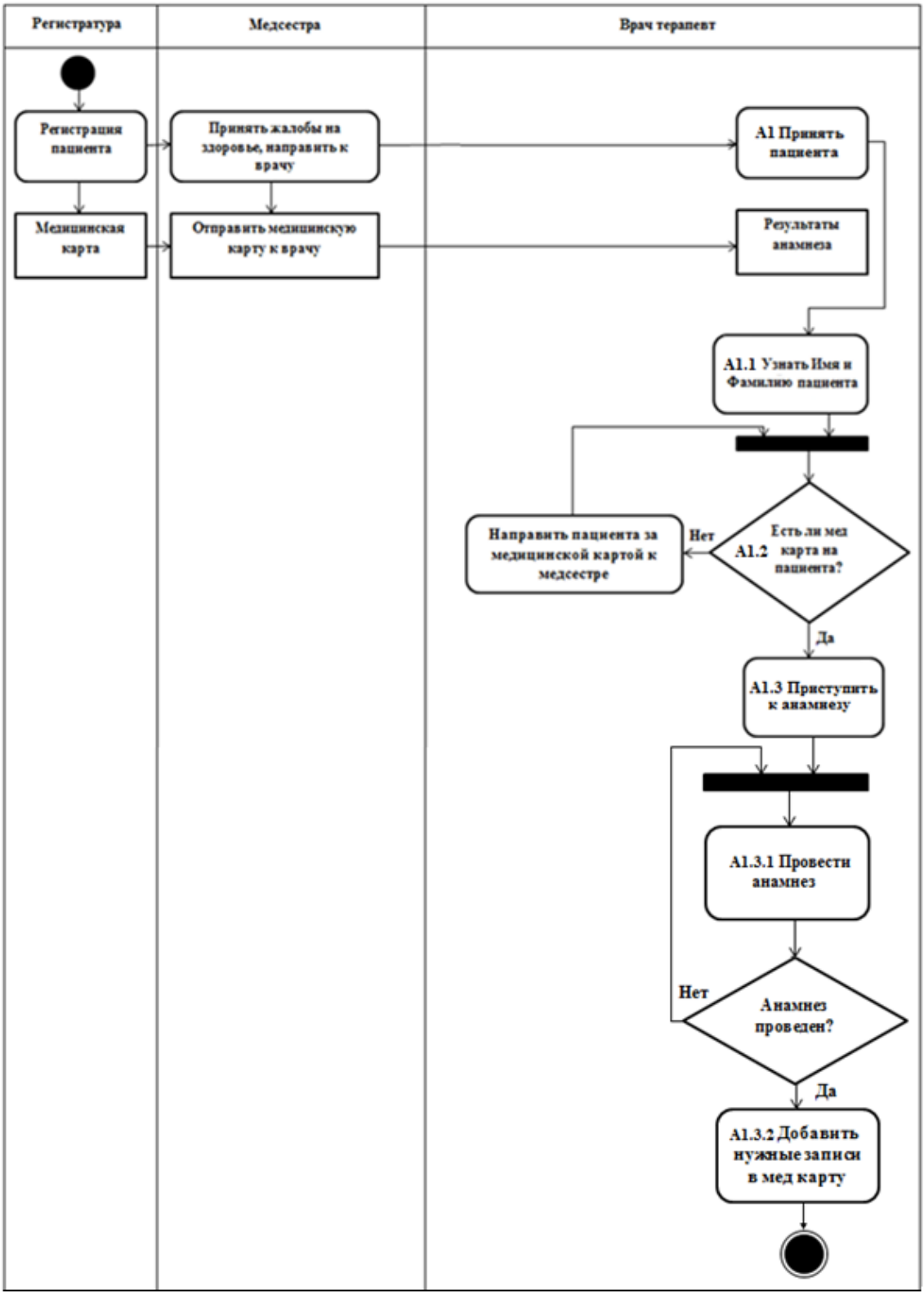

Рис. 2.5. – *Представление процесса в UML AD на третьем уровне в модели «AS-IS» для уточнения процесса «Приступить к анамнезу»*

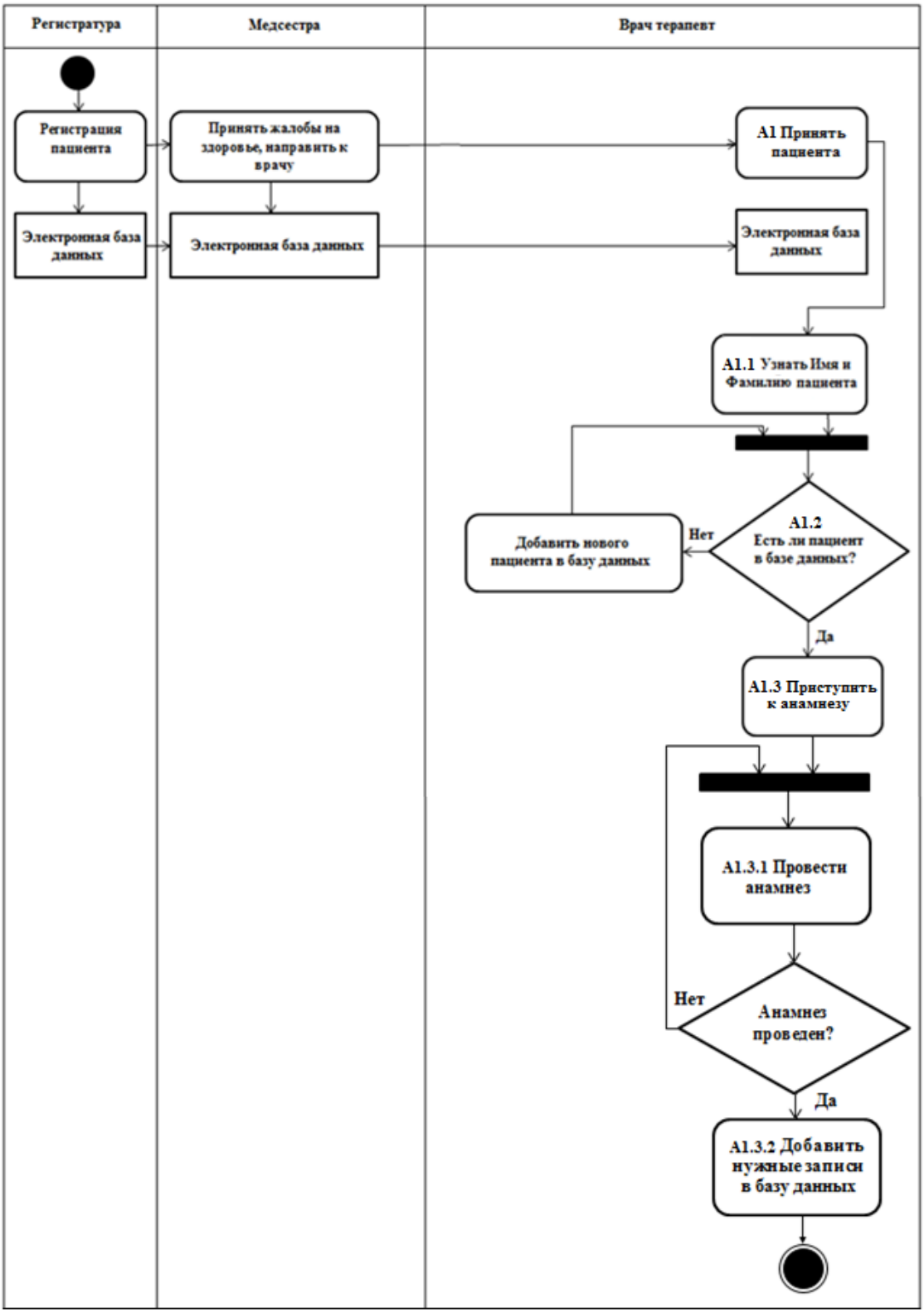

Рис. 2.6. – *Представление процесса в UML AD на третьем уровне в модели «TO-BE» для уточнения процесса «Приступить к анамнезу»*

Для более наглядного представления вышеуказанных бизнес-процессов на 1-3 уровнях, построена карта бизнес-процессов в модели «TO-BE» (рисунок 2.7).

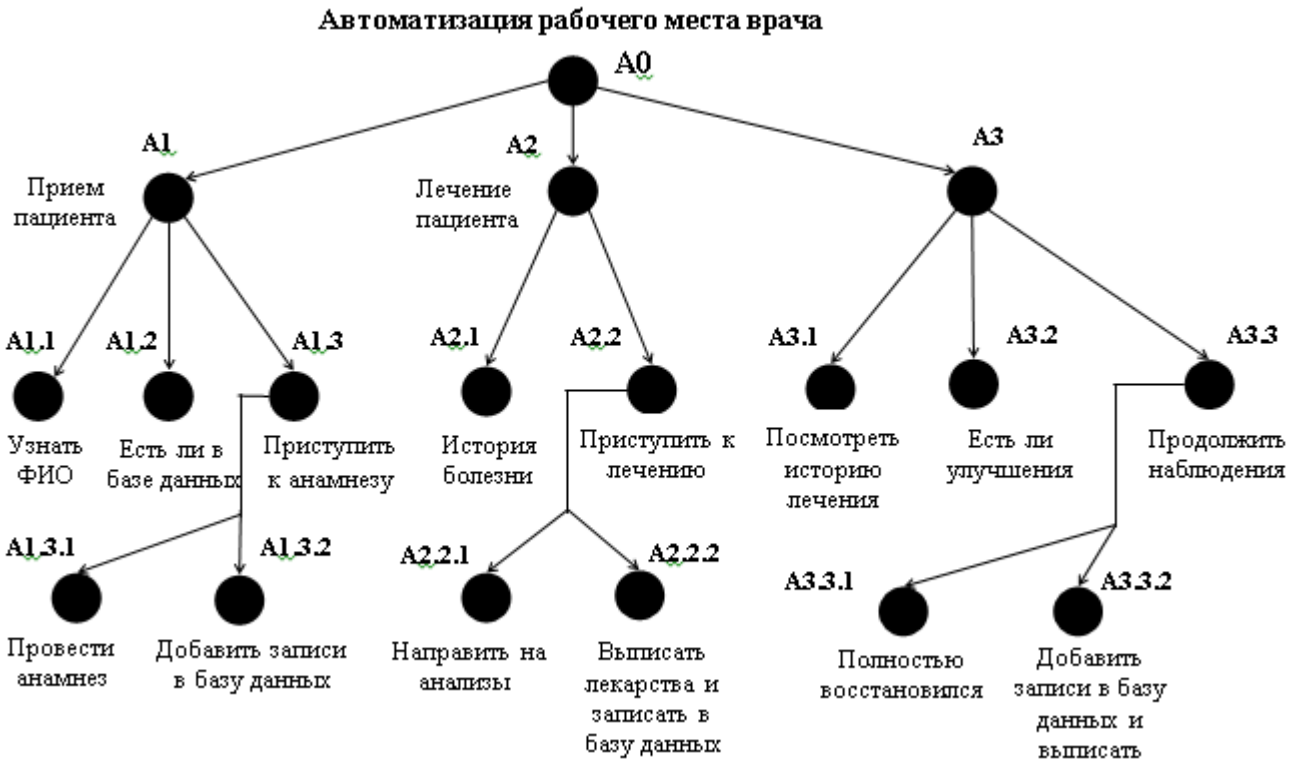

Рис. 2.7 – *Карта бизнес-процессов в модели «TO-BE»*

Литература

1. А.В. Варзунов, Е.К. Торосян, Л.П. Сажнева. "Анализ и управление бизнес-процессами". Учебное пособие, 2010 – 212 с.
2. И.В. Абрамов. Методические указания по дисциплине "Модели и методы информационно-управляющих систем". Ижевск, 2004 – 314 с.
3. Владимир Репин, Виталий Елиферов. "Процессный подход к управлению". Моделирование бизнес-процессов. Издательство "Манн, Иванов и Фербер", Москва, 2013 – 215 с.
4. Ю.А. Ларина. "Основы объектно-ориентированного моделирования с использованием языка UML". Учебное пособие, Ярославль, 2010 – 152 с.
5. Ю.А. Ларина. "Основы объектно-ориентированного моделирования с использованием языка UML". Учебное пособие, Ярославль, 2010 – 152 с.

6. Дейт К. Дж. Введение в системы баз данных / пер. с англ. и ред. К. А. Птицына – 8-е изд. – М.: Вильямс – 2016. – 327 с.
7. Штенников Д.Г. Разработка информационных систем в образовании. Учебное пособие. – СПб: СПбГУ ИТМО, 2012. – 242 с.

Выходные данные статьи

Дудкин Д.И. Автоматизация работы врача терапевта в городской поликлинике на основе спиралевидной модели внедрения информационных систем (Часть 1) // Корпоративные информационные системы. – 2021. – №3 (15) – С. 43-58. – URL: https://corpinfosys.ru/archive/issue-15/182-2021-15-therapistautomation.

Об авторе

Дудкин Дмитрий Игоревич – студент 4-го курса кафедры оптических и биотехнических систем и технологий физико-технологического института РТУ МИРЭА. Тема выпускной квалификационной работы бакалавра «Автоматизация ключевого бизнес-процесса врача терапевта в городской поликлинике на основе спиралевидной модели внедрения информационных систем». Электронная почта: mail@corpinfosys.ru.