ПРИМЕНЕНИЕ МЕТОДОЛОГИИ AGILE SCRUM ДЛЯ РЕАЛИЗАЦИИ АВТОМАТИЗИРОВАННОГО РАБОЧЕГО МЕСТА ВРАЧА ПЕДИАТРА В ГОРОДСКОЙ БОЛЬНИЦЕ

Болохнов Александр Андреевич, Чеканова Елизавета Романовна

bolokhnov@sumirea.ru, elizabeth-k@bk.ru

студенты 4 курса напр. «Биотехнические системы и технологии»,

МИРЭА - Российский Технологический Университет, г. Москва

Аннотация: выполняется обзор и детальный анализ методологии Agile Scrum, способов проектирования бизнес-процессов в нотациях ARIS VACD и eEPC для реализации автоматизированного рабочего места врача педиатра в городской больнице.

Ключевые слова: информатизация здравоохранения, автоматизированное рабочее место врача, база данных, модель AS-IS (как есть), модель TO-BE (как будет), ARIS VACD, eEPC, приоритезация, спринт, бизнес - процесс, пользовательские требования, функциональные требования.

В современном мире компьютерные технологии имеют большой смысл во многих сферах человеческой деятельности, в том числе и в медицине. Множество функциональных возможностей компьютерной техники позволяет повысить эффективность работы врача. Проанализировав литературу на данную тему, можно сделать вывод, что врачам как пользователям нужны максимально простые, удобные и продуктивные программно-технические средства информационного сервиса. Немаловажно, чтобы автоматизированное рабочее место не нарушало привычного для врача ритма и стиля работы.

Объем информации у современного врача-педиатра с каждым годом все увеличивается, а времени на лечение детей не становится больше. Поэтому на сегодняшний день улучшение рабочего места врача-педиатра является значимым фактором в его оперативности.

ЦЕЛЬ И ЗАДАЧИ

Цель данной работы заключается в детальном анализе методологии Agile Scrum для разработки автоматизированного рабочего места врача педиатра в городской больнице.

Чтобы достичь вышеуказанной цели нужно реализовать следующие задачи:

- детальный анализ методологии внедрения Agile Scrum;
- идентификация требований и формирование бэклога;
- проектирование процессов и оргструктуры в моделях AS-IS и TO-BE нотации ARIS VACD и eEPC до 3-4 уровней детализации;

для каждого спринта:

- моделирование разрабатываемых пользовательских интерфейсов;
- проектирование структуры данных и нормализация таблиц данных;
- реализация операции ключевого процесса в среде MS Access.

ДЕТАЛЬНЫЙ АНАЛИЗ МЕТОДОЛОГИИ ВНЕДРЕНИЯ AGILE SCRUM

Для разработки программного обеспечения требуется своевременное принятие верных решений, которое надо «упорядочить». Бизнес-процесс - это устойчивая целенаправленная последовательность исполнения функций, направленная на создание результата, имеющего ценность для потребителя [1]. При построении современного бизнес-процесса нужен руководитель, одно из качеств которого будет умение управлять командой, работающей над проектом. Для этого требуется особый гибкий подход, доверие коллегам и готовность к изменениям рынка. Принципы гибкого управления включены в концепцию Agile Scrum, разработанные компаниями, занятыми в сфере IT.

Agile (от англ. Agile-проворный) - это семейство «гибких» подходов к разработке программного обеспечения [2]. Согласно данной методологии, проект разбивается не на последовательные фазы, а на маленькие подпроекты, которые затем «собираются» в готовый продукт. Таким образом, верхнеуровневое планирование проводится для всего проекта, а последующие этапы: разработка, тестирование и прочие проводятся для каждого мини-проекта отдельно [3]. Agile возник в IT-среде, но затем распространился и в другие сферы. Agile предполагает, что при реализации проекта не нужно опираться только на заранее созданные подробные планы: важно ориентироваться на постоянно меняющиеся условия внешней и внутренней среды и учитывать обратную связь от заказчиков и пользователей. Это поощряет разработчиков и инженеров экспериментировать и искать новые решения, не ограничивая себя жесткими рамками и стандартами [4].

Одним из Agile-подходов является Scrum. Scrum («подход структуры») - это набор принципов, на которых строится процесс разработки, позволяющий в жестко фиксированные и небольшие по времени циклы, называемые спринтами (Sprints), предоставлять конечному пользователю работающее ПО с новыми возможностями, для которых определен наибольший приоритет. Возможности ПО к реализации в очередном спринте определяются в начале спринта на этапе планирования и не могут изменяться на всём его протяжении. При этом строго фиксированная небольшая длительность спринта придает процессу разработки предсказуемость и гибкость. В основе методологии Scrum лежит простая идея: когда бы ни был запущен проект, ничто не мешает регулярно проверять ход работ и последовательно выяснять: справляетесь ли вы с заданием; в нужном ли направлении движетесь; создаете ли именно то, что на самом деле хочет получить заказчик. Вам также ничто не мешает постоянно поднимать следующие вопросы: есть ли способы усовершенствовать методы разработки и выполнять работу наиболее качественно и быстро; существуют ли факторы, препятствующие вашим задачам [5]. Подход, лежащий в основе методики Scrum, можно применять в разных видах деятельности, в которых требуется

коллективная работа. Важными характеристиками Scrum являются ее гибкость и ориентированность на клиента, так как она предполагает непосредственное участие клиента в процессе работы. Scrum не требует внедрения каких-либо дорогостоящих инструментов. Схему методики Scrum вкратце можно описать следующим образом:

- первым делом надо выбрать «Владельца продукта» (Product owner) - человека, способного видеть то, что вы собираетесь создать или достигнуть. Он является связующим звеном между командой разработки и заказчиком. Его задачей является максимальное увеличение ценности разрабатываемого продукта и работы команды;
- затем нужно собрать «Команду разработки» (Development team), в которую войдут люди, непосредственно выполняющие работу. Они должны обладать навыками и знаниями, которые помогут воплотить идею владельца продукта в жизнь. А также следует иметь такие качества и характеристики, как самоорганизация и многофункциональность;
- далее необходимо выбрать «Скрам-мастера» (Scrum master) - того, кто будет следить за ходом реализации проекта и помогать команде устранять препятствия на пути достижения цели;
- приступая к работе, нужно создать максимально полный список всех требований, предъявляемых к продукту или цели, упорядоченный по их степени важности, подлежащих реализации. Пункты этого списка должны быть расставлены по приоритету. Список носит название «Бэклог продукта» (Product Backlog). Он может развиваться и изменяться на протяжении всего срока реализации проекта;
- участники команды должны оценить по своей системе оценок каждый пункт на предмет сложности и затрат, которые потребуются для его выполнения. Затем участники, скрам-мастер и владелец продукта должны провести первое скрам-собрание, на котором они запланируют спринт (определенное время для выполнения части заданий). Его продолжительность не должна превышать один месяц. Тем не менее, считается, что чем короче спринт,

тем более гибким является процесс разработки, релизы выходят чаще, быстрее поступают отзывы от потребителя, меньше времени тратится на работу в неправильном направлении. С другой стороны, при более длительных спринтах команда имеет больше времени на решение возникших в процессе проблем, а владелец продукта уменьшает издержки на совещания, демонстрации продукта и тому подобное. Бывает такое, что приходится останавливать спринт, но это происходит в исключительных случаях. Спринт может быть остановлен до того, как закончатся отведенные для него дни, или его может остановить команда, если понимает, что не может достичь цели спринта в отведенное время (рисунок 1). Спринт может остановить владелец продукта, если необходимость в достижении цели спринта исчезла;

- команда должна все время стремиться к тому, чтобы превзойти в новом спринте свои собственные результаты за предыдущий спринт;
- по завершении спринта команда делает его обзор - проводит встречу, на которой участники рассказывают, что сделано за спринт;
- после показа результатов работы за спринт участники проводят ретроспективное собрание, на котором обсуждают, что команда сделала хорошо, что можно сделать лучше, что можно улучшить прямо сейчас [6].

Рисунок 1 – Пример процесса реализации согласно Scrum

Достоинства данной методологии в ее гибкости и адаптивности. Всегда можно что-то поменять в продукте, добавить еще одну функцию. Agile Scrum очень удобен, когда заказчик сам до конца не знает, чего он хочет. Таким образом, получается выпустить программу с основными функциями, а с каждым последующим спринтом добавлять к ней новые. Еще один плюс этой методологии - это самостоятельность и самоорганизованность каждого участника проекта [7]. Но в таком случае достаточно много внимания уделяется отбору персонала. Особенностью Agile Scrum является вовлеченность в процесс всех участников команды, причем у каждого участника есть своя определенная роль. Команда самостоятельно регулирует собственную работу, устанавливает временные рамки и условия работы в границах спринта. Руководитель проекта, выступающий мастером Scrum, не может указывать команде, каким образом создавать продукт [8]. При этом в команду разработки входят функциональные специалисты, обладающие необходимыми навыками для работы. В структуре команды нет разделения на подразделения, выполняющие отдельные функции. Даже если отдельные члены команды владеют узкоспециализированными знаниями в различных областях, ответственность за продукт лежит на всей команде в целом. Однако существует такая проблема, как неопределенность [9]. Количество спринтов не ограничено, поэтому сложно поставить конечную дату в проекте. Таким образом, можно выделить плюсы Agile Scrum:

- быстрый запуск;
- быстрое реагирование на изменения (гибкость и адаптивность);
- связь в реальном времени между командой и клиентом;

а также минусы:

- требует высококвалифицированных и мотивированных специалистов;
- требует много времени от заказчика;
- отсутствие долгосрочных планов.

ИДЕНТИФИКАЦИЯ ТРЕБОВАНИЙ И ФОРМИРОВАНИЕ БЭКЛОГА

Идентификация предварительных требований к новому продукту - это то, что позволяет определить характеристики нового продукта в соответствии с требованиями клиента. Команда, которая занимается проектом, ставит определенные цели, соответствующие либо каким-то чертам продукта конкурента, либо требованиям рынка, либо идентифицирует какие-то признаки, которые конечный пользователь хотел бы добавить к уже существующему продукту. Описание функциональных возможностей и ограничений, накладываемых на систему, называется требованиями к этой системе. Требования к продукту должны быть установлены таким образом, что могло бы гарантировать их адекватность и верный «перевод» с языка пользователя. В первую очередь необходимо выявить все возможные требования, предъявляемые к разрабатываемому программному продукту. Выявить их можно с помощью анализа требований - процесса сбора требований к программному обеспечению, их систематизации, документирования, анализа, выявления противоречий, неполноты, разрешения конфликтов в процессе разработки программного обеспечения. В процессе сбора требований важно принимать во внимание возможные противоречия требований различных заинтересованных лиц, таких как заказчики, разработчики или пользователи. Анализ требований включает в себя следующие шаги:

- сбор требований: общение с клиентами и пользователями, чтобы определить, каковы их требования;
- анализ требований: определение, являются ли собранные требования неясными, неполными, неоднозначными, или противоречащими и затем решение этих проблем;
- документирование требований: требования могут быть задокументированы в различных формах, таких как простое описание, сценарии использования, пользовательские истории, или спецификации процессов.

Таким образом, первым делом была проведена работа с заказчиком системы. Педиатр – специалист, в задачи которого входят профилактика, диагностика, лечение заболеваний и реабилитация детей до 18 лет, решение вопросов рационального питания и правильного развития подрастающего поколения, а также оценка психического и физического развития ребенка при рождении, в дошкольных учреждениях, школах [10].

После этого был составлен список выявленных пользовательских требований (требования, формулируемые пользователем к конечному продукту), на которые будет полагаться разработка приложения:

- возможность хранения данных (о ребенке, о родителях ребенка, о рецептах на молочную кухню);
- возможность поиска данных о конкретном ребенке, о конкретном рецепте;
- возможность добавления новых рецептов на молочную кухню;
- возможность просмотра данных о ребенке, о рецептах на молочную кухню;
- возможность работы с приложением на любом компьютере медицинского учреждения;
- простота и удобство пользования интерфейса;
- защита информации от несанкционированного доступа.

Важным этапом работы с бэклогом продукта является выстраивание требований в порядке их приоритета. Необходимо определить, в какой последовательности будет проходить реализация задач, что будет выпущено в первой версии продукта, что может быть включено в последующие поставки, а что может быть оставлено на тот случай, если останется время. Провести сортировку бэклога и выстроить тот или иной порядок задач можно базируясь на различных критериях, исходя из разных ценностей. Существует много вариантов техник, с помощью которых можно провести сортировку бэклога. Методика приоритезации - одна из них. При оценке приоритета этим методом для каждой задачи подбирается наиболее соответствующее ей одно утверждение из четырех:

1 – требование, не подлежащее обсуждению и оно в любом случае должно быть реализовано; 2 – в эту категорию входят требования, которые критичны для функционала, но не для текущего релиза; 3 – категория требований, которую желательно включить, но она не влияет на релиз успеха; 4 – для этой категории ключевым моментом является то, что данные требования могут быть также важны (также, как и 1), однако, их можно оставить для более поздних поставок продукта. Каждое из утверждений является индикатором определенного уровня важности задачи.

Следующим шагом были выявлены функциональные требования, определяющие поведение программной системы, чтобы выполнить пользовательские требования:

- база данных, содержащая таблицы «Личные данные ребенка», «Рецепты», «Личные данные родителей», «Лечить ребенка», «Выписать ребенка», «Принять ребенка», «Заключительная»;
- возможность просмотра данных из таблицы «Личные данные ребенка»;
- возможность просмотра данных из таблицы «Рецепты»;
- возможность программы добавлять новый рецепт на молочную кухню с последующим сохранением в базе данных;
- поиск конкретного ребенка по индивидуальному коду ребенка;
- поиск конкретного рецепта по индивидуальному коду из таблицы «Рецепты»;
- вывод на экран данных из таблицы «Личные данные ребенка»;
- вывод на экран данных из таблицы «Рецепты»;
- программа должна корректно работать на всех компьютерах медучреждения;
- установление пароля на саму базу данных;
- доступность (легкость восприятия).

Исходя из вышеуказанной информации, нужно создать максимально полный список всех требований, предъявляемых к продукту, то есть создать

бэклог. Данный список будет состоять из пользовательских и функциональных требований. Бэклог продукта является списком планируемых улучшений продукта, расставленных по приоритету. В данной работе этот список будет в виде таблицы (таблица 1). Разработка программного обеспечения будет производиться следующим образом (таблица 2).

Таблица 1 – Бэклог продукта.

№	Пользовательские требования	Функциональные требования	Приоритет требований
1	Возможность хранения данных	Хранение информации о ребенке, его родителях, о рецептах на молочную кухню непосредственно в создаваемом приложении	1
2	Возможность просмотра данных о ребенке, о рецептах на молочную кухню	Возможность просмотра данных из таблиц «Личные данные ребенка», «Рецепты»; вывод на экран данных из таблиц «Личные данные ребенка», «Рецепты»	2
3	Возможность добавления новых рецептов на молочную кухню	Возможность программы добавлять новый рецепт на молочную кухню с последующим сохранением в базе данных	2
4	Возможность работы с приложением на любом компьютере медицинского учреждения	Программа должна корректно и удобно для пользователей работать на всех компьютерах медицинских учреждений	2
5	Возможность поиска определенных данных о конкретном ребенке или рецепте	Поиск конкретного ребенка по коду ребенка или рецепта по коду рецепта	3
6	Защита информации от несанкционированного доступа	Установление пароля на базу данных	4
7	Простота и удобство пользования интерфейса	Доступность (легкость восприятия)	4

Таблица 2 – Бэклог спринтов.

№ спринта	Требования - задачи, реализуемые в спринте
0	Анализ требований; формирование бэклога продукта; проектирование процессов AS-IS и TO-BE на основе ARIS VACD и eEPC; проектирование базы данных и интерфейсов.
1	Реализация требования 1: хранение информации. Проектирование таблиц в определенной среде разработки.
2	Реализация требований 2 - 4: возможность просмотра данных, а также возможность внесения изменения информации.
3	Реализация требования 5: возможность поиска данных.
4	Реализация требования 6: защита от несанкционированного входа. Реализация требования 7: создание простого и легкого интерфейса; исправление ошибок.

НУЛЕВОЙ СПРИНТ

Согласно бэклогу спринтов, в нулевом спринте реализуются анализ требований; формирование бэклога продукта; проектирование процессов AS-IS и TO-BE на основе ARIS VACD и eEPC; проектирование базы данных и интерфейсов.

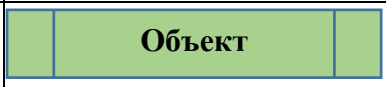
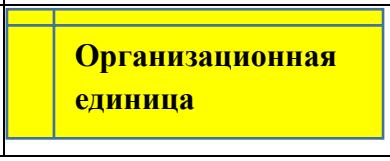
Бизнес-процесс - это деятельность или набор действий, которые выполняют определенную организационную цель. Бизнес-процессы должны иметь целенаправленность, быть максимально конкретными и иметь последовательные результаты. Одним из эффективных методов преобразования идеи в конкретные результаты является разработка и заполнение диаграмм AS-IS (как есть) и TO-BE (как будет). Диаграмма AS-IS описывает текущее состояние процесса, возможностей организации. Диаграмма TO-BE описывает будущее состояние, другими словами, какие процессы и возможности организации появятся в будущем [6]. Первый шаг - определение необходимого процесса AS-IS, как только он определён его надо проанализировать. В этот момент необходимо задокументировать текущее состояние процесса, определить точку отсчета. Необходимо четко понимать функционирование и возможности процесса, системы. После этого можно приступать к графическому

отображению процессов AS-IS. Второй шаг - выявить любые недостатки в существующих процессах. Третий шаг – документирование и реализация процесса TO-BE. На этом этапе также происходит детальный анализ желаемых улучшений процесса и формируются конкретные предложения по его улучшению. Далее, как и в модели, AS-IS создается графическое отображение процессов (блок-схема). Это наиболее сложный этап. При его реализации часто может оказаться, что желаемые улучшения не столь эффективны, как планировалось ранее. Последним шагом будет отслеживание реальной эффективности и актуальности введенных улучшений.

Одна из методологий для моделирования бизнес-процессов – методология ARIS (Architecture of Integrated Information Systems) или «архитектура интегрированных информационных систем». Методология состоит из различных нотаций (система условных обозначений, принятая в конкретной модели) и позволяет описать с различных точек зрения деятельность организации. ARIS рассматривает деятельность организации с четырех точек зрения: организационная структура, информационная структура, функциональная иерархия, управление бизнес-процессами. Преимуществом моделирования в ARIS является его наглядность в представлении сложных фактов, также это позволяет систематизировать подход к анализу. В зависимости от интересующей информации используются различные методы моделирования. К числу нотаций ARIS относятся: Value-added Chain Diagram (диаграмма цепочки процесса, добавляющего стоимость); нотации eEPC, Extended Event-driven Process Chain (расширенная нотация цепочки процесса, управляемого событиями) и другие [7]. Далее будет рассмотрена нотация Value-added Chain Diagram (VACD). Нотация Value-added Chain Diagram (VACD) или «диаграмма цепочки процесса, добавляющего стоимость) – это тип модели, используемый для формулировки основных уровней процесса. VACD в основном используется для описания процессов верхнего уровня внутри организации. Основной объект нотации – процесс или группа функций организации, служащие для получения

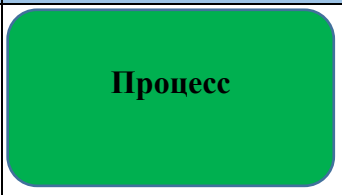
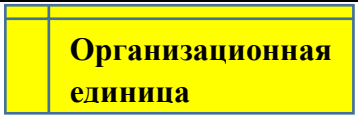
добавленной стоимости. Элементы, используемые для моделирования нотации VACD, представлены в таблице 3 [8].

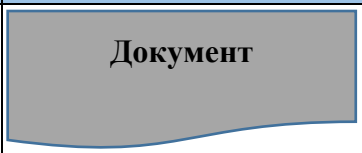
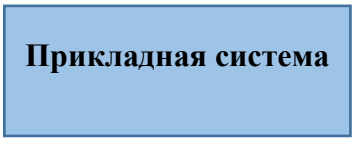
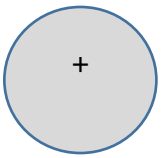
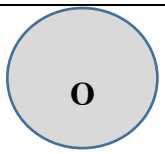
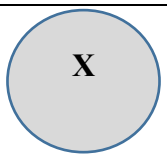
Таблица 3 – Графические элементы ARIS VACD.

Наименование	**Описание**	**Графическое представление**
Процесс	Элемент отражает процессы, выполняемые в организации	
Входящий или исходящий объект	Элемент отражает реальные носители информации	
Организационная единица	Элемент, отражающий различные организационные звенья организации, ответственный за исполнение процесса	

Нотация ARIS extended Event Driven Process Chain (ARIS eEPC) подходит для более низкоуровневого (2-3 уровень) описания процессов. По сути это расширенная версия нотации ARIS VACD, которая позволяет более детально изучить протекающие в организации процессы. Как правило, именно на данном этапе проектирования становятся очевидны проблемы, мешающие развитию организации. Элементы, используемые в данной нотации, приведены в таблице 4.

Таблица 4 – Графические элементы ARIS eEPC.

Наименование	**Описание**	**Графическое описание**
Процесс	Объект "Процесс" служит для описания функций (процедур, работ), выполняемых подразделениями/сотрудниками предприятия	
Событие	Объект "Событие" служит для описания реальных состояний системы, влияющих и управляющих выполнением функции	
Организационная единица	Объект, отражающий реальные различные организационные звенья предприятия	

Наименование	Описание	Графическое описание
Документ	Элемент отображает бумажные документы, сопровождающих выполнение функции	
Прикладная система	Объект отражает реальную прикладную систему, используемую в рамках технологии выполнения функции	
Стрелка связи между объектами	Объект описывает тип отношений между другими объектами	
Оператор «И»	Оператор ставится в случае, когда начало/завершение выполнения функции должно инициировать одновременно несколько событий	
Оператор «ИЛИ»	Оператор ставится, когда начало/завершение выполнения функции может инициировать несколько событий	
Исключающий оператор «ИЛИ»	Оператор ставится, когда начало/завершение выполнения функции может инициировать только одно из событий в зависимости от условия	

На данном этапе проектирования описывается основной бизнес-процесс врача-педиатра. Рисунок 2 показывает первый уровень проектирования процессов работы врача-педиатра в нотации ARIS VACD модели «AS-IS», которая подразумевает под собой проектирование процесса в настоящий момент времени.

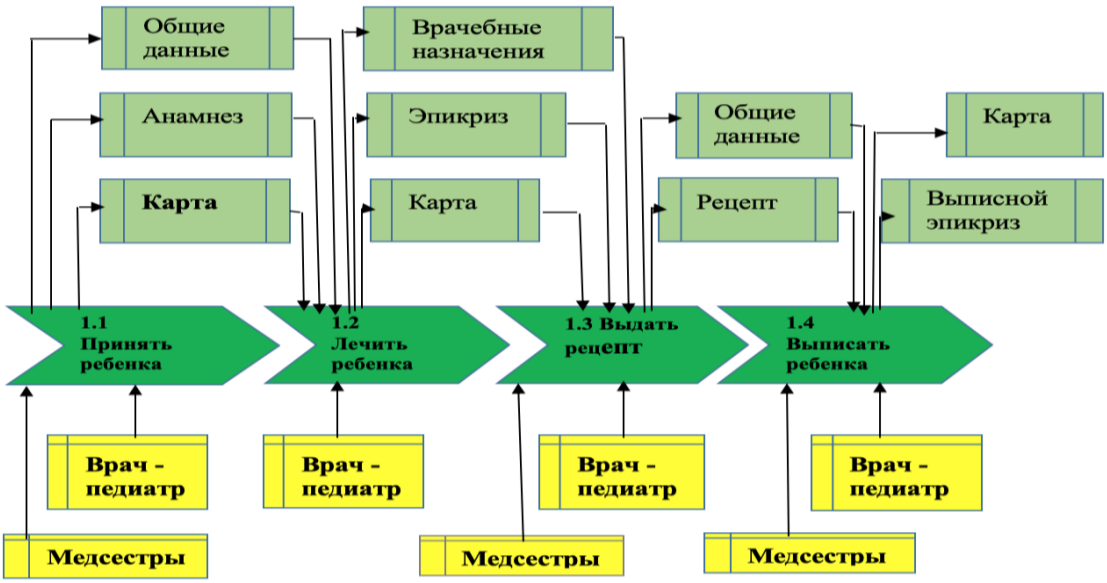


Рисунок 2 – Первый уровень описания процессов

Второй уровень процесса описывается с помощью нотации ARIS eEPC модели «AS-IS». Рассмотрим процесс «Выдать рецепт». Данный процесс состоит из нескольких процессов: 1.3.1 посмотреть данные о ребенке; 1.3.2 осмотреть, провести первичное обследование ребенка; 1.3.3 заполнить шаблон рецепта; 1.3.4 выдать рецепт. Графическое представление представлено в приложении А на рисунке А.1.

Рисунок 3 показывает первый уровень проектирования процессов работы врача-педиатра в нотации ARIS VACD модели «TO-BE».

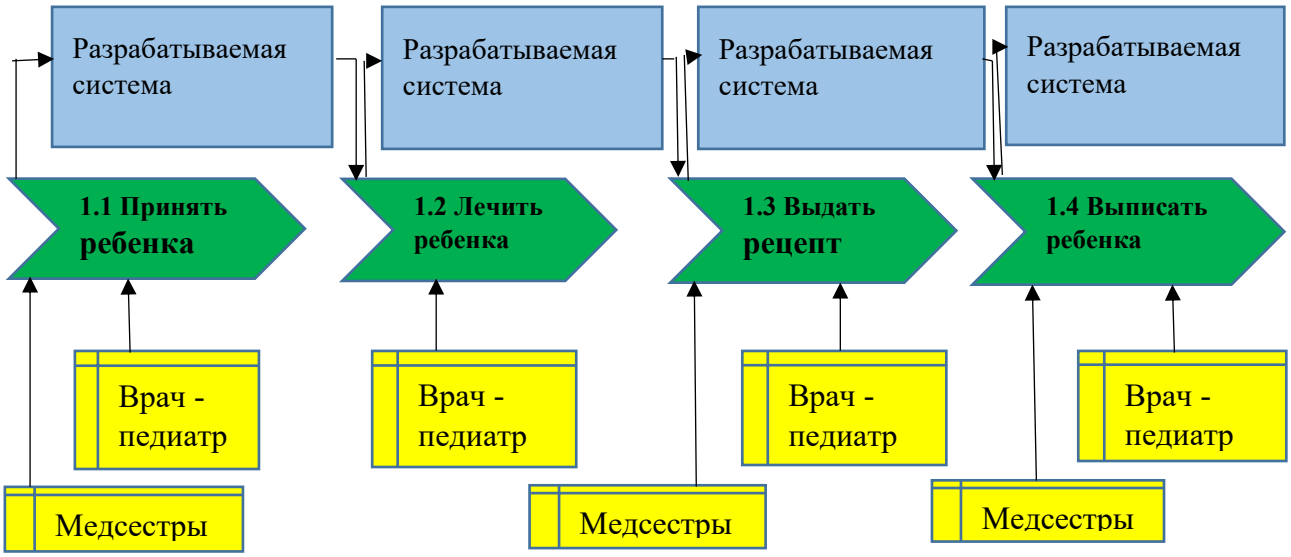


Рисунок 3 – Первый уровень описания процессов

В приложении А на рисунке А.2 показан второй уровень проектирования процесса «Выдать рецепт» работы врача-педиатра в нотации ARIS eEPC модели «TO-BE».

Так же данная часть проектирования подразумевает создание карты процессов, которая представлена в приложении А на рисунке А.3; проектирование базы данных и интерфейсов.

СПРИНТ ПЕРВЫЙ

На данном этапе реализуется требование 1: хранение информации. Реализация программы на первом спринте представляет собой обычную таблицу с элементами управления самой среды разработки MS Access. Здесь нет никакого интерфейса, который мог бы облегчить работу пользователя с программой. Ниже представлен фрагмент таблицы «Личные данные ребенка», созданной в программе MS Access (рисунок 4).

Код ребени	Имя	Фамилия	Отчество	Дата рожд	Полис ОМС	Адрес прог	Адрес фак	Домашний	Группа кр	Резус факт	Инвалидно	Пол
1	Дарья	Пешая	Романовна	15.12.2001	77000000000	г. Москва, у	г. Москва, у	84999049782	3	Положитель	нет	жен
2	Максим	Пеший	Алексеевич	14.05.2005	77000000000	г. Москва, у	г. Москва, у	84999096326	4	Положитель	нет	муж
3	Эдуард	Алексеев	Владимирови	01.01.2003	77000000000	г. Москва, у	г. Москва, у	84999098877	1	Отрицатель	1 степень	муж
4	Полина	Орлова	Андреевна	08.08.2008	77000000000	г. Москва, у	г. Москва, у	84999046678	2	Положитель	нет	жен
5	Николай	Незнаев	Владимирови	28.09.2007	77000000000	г. Москва, у	г. Москва, у	84999045573	3	Положитель	2 степень	муж
6	Марина	Лапова	Артемьевна	30.04.2005	77000000000	г. Москва, у	г. Москва, у	84999097611	2	Отрицатель	нет	жен
7	Артемий	Артемов	Николаевич	03.03.2003	77000000000	г. Москва, у	г. Москва, у	84999090101	3	Положитель	нет	муж
8	Мария	Петрова	Петровна	26.07.2008	77000000000	г. Москва, у	г. Москва, у	84999099191	1	Положитель	нет	жен
9	Арина	Петрова	Олеговна	15.03.2004	77000000000	г. Москва, у	г. Москва, у	84999096161	2	Положитель	нет	жен
10	Елена	Талыкова	Вячеславовн	05.07.2005	77000000000	г. Москва, у	г. Москва, у	84999096555	3	Положитель	нет	жен

Рисунок 4 – Фрагмент таблицы «Личные данные ребенка» базы данных

На рисунке 5 показана схема данных и связи всех используемых таблиц данных, необходимых для приложения. Схема данных строилась путем добавления необходимых таблиц данных, выделением из них ключевых полей, необходимых для создания связей между ними. Внутри данной базы данных создано несколько таблиц («Личные данные ребенка», «Рецепты», «Личные данные родителей», «Лечить ребенка», «Выписать ребенка», «Принять ребенка», «Заключительная») и связи между ключевыми полями (поле «код ребенка» из

таблицы «Личные данные ребенка» связано с полем «Код ребенка» из таблицы «Рецепты»).

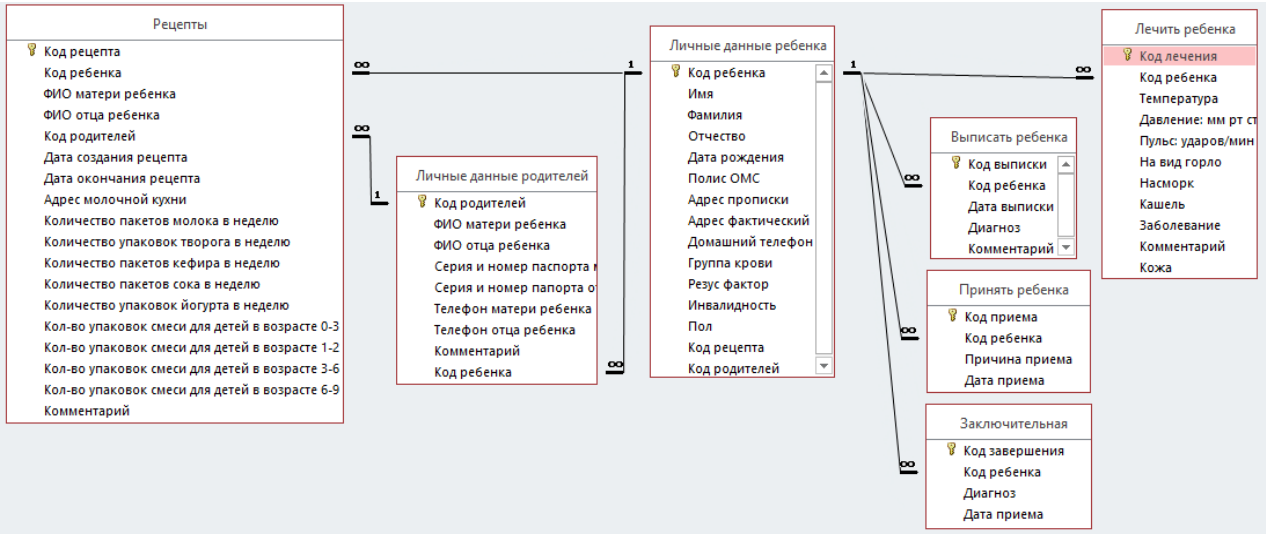


Рисунок 5 – Архитектура данных разрабатываемого продукта

СПРИНТ ВТОРОЙ

На данном этапе реализовываются требования 2 - 4: возможность просмотра данных, а также возможность внесения изменения информации. Для наиболее удобной работы конечного пользователя с программой необходимо создать максимально понятные пользовательские интерфейсы программы. Когда пользователь только открывает программу, ему предстоит выбрать, какой процесс будет происходить. Интерфейс «Главное меню программы» представлен на рисунке 6.

Рисунок 6 – Интерфейс «Главное меню программы»

Допустим, пользователь решил выдать новый рецепт. В таком случае, он нажимает кнопку «Принять ребенка», далее ему откроется следующее диалоговое окно (рисунок 7), в котором он выбирает «Выдать рецепт».

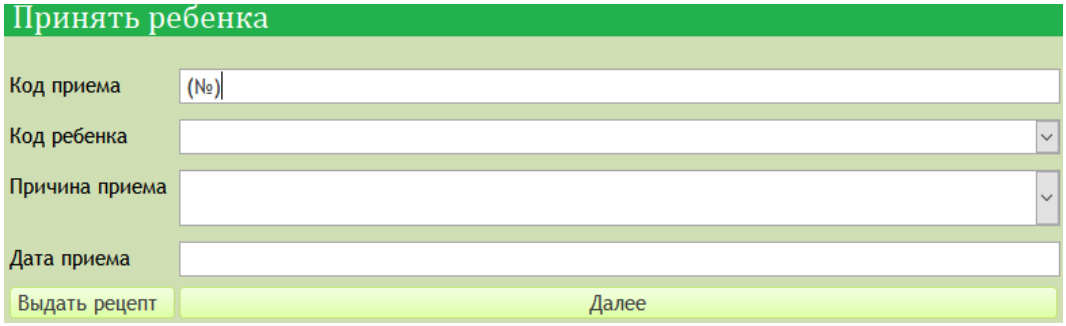


Рисунок 7 – Интерфейс «Принять ребенка»

Если пользователь нажал кнопку «Выдать рецепт», то на экране появляется интерфейс «Рецепты» (рисунок 8). Чтобы добавить новый рецепт, нужно выбрать ребенка, которому будет добавляться рецепт, для этого необходимо ввести код ребенка в строку «Код ребенка». Код нового рецепта будет введен автоматически.

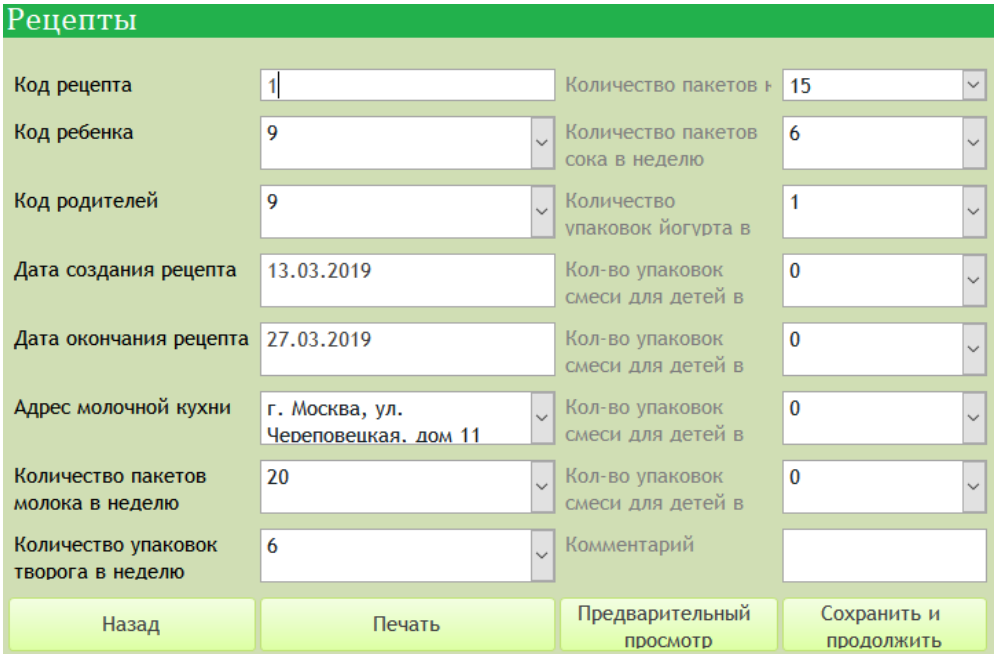


Рисунок 8 – Интерфейс «Рецепты»

Если же пользователь нажал кнопку «Далее» в интерфейсе «Принять ребенка», то появляется интерфейс «Лечить ребенка» (рисунок 9).

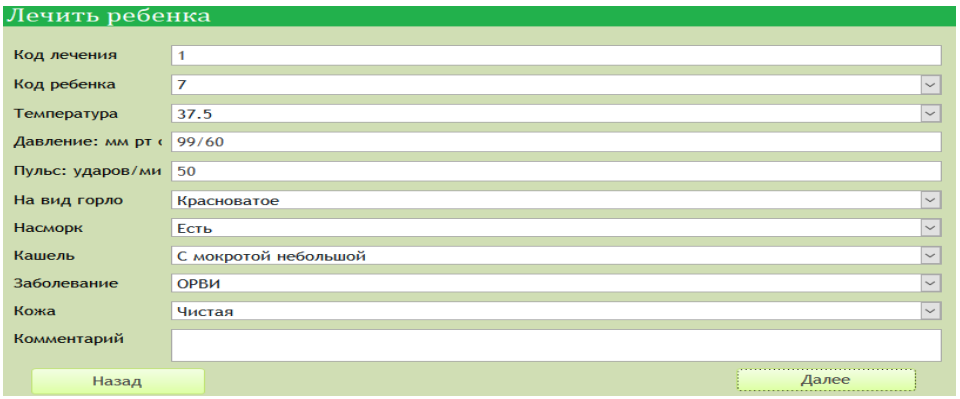


Рисунок 9 – Интерфейс «Лечить ребенка»

Если пользователь нажал кнопку «Далее» в интерфейсе «Лечить ребенка» или «Сохранить и продолжить» в интерфейсе «Рецепты», то появляется интерфейс «Завершение приема» (рисунок 10).

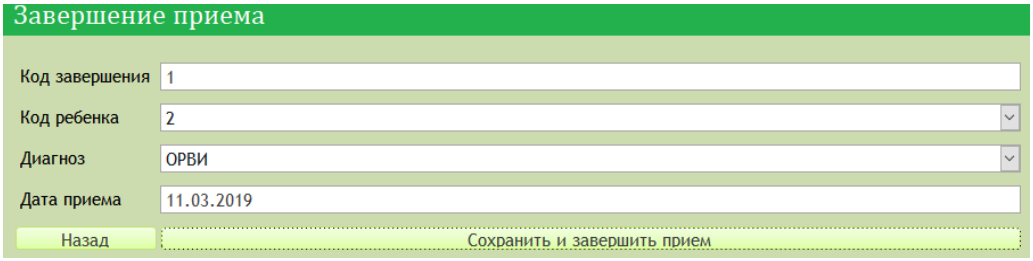


Рисунок 10 – Интерфейс «Завершение приема»

После того, как пользователь нажмет на кнопку «Сохранить и завершить прием» на интерфейсе «Завершение приема», программа вернется к интерфейсу «Главное меню программы».

СПРИНТ ТРЕТИЙ

Третий спринт подразумевает реализацию требования 5: возможность поиска данных. На данном этапе создаются запросы для поиска нужной информации среди всей базы данных. С их помощью работа осуществляется с частью базы данных. Запрос с названием «Поиск по коду рецепта» позволяет найти конкретный рецепт из таблицы «Рецепты» (рисунок 11). Аналогичным способом был создан запрос с названием «Поиск по коду ребенка», позволяющий найти личные данные ребенка из таблицы «Личные данные ребенка».

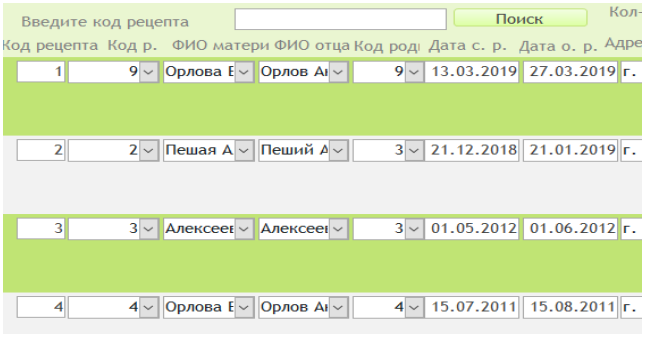


Рисунок 11 – Интерфейс «Поиск рецепта»

СПРИНТ ЧЕТВЕРТЫЙ

Четвертый спринт подразумевает реализацию требований 6 – 7: защита от несанкционированного доступа, создание простого и легкого интерфейса.

На данном этапе реализовывается возможность защиты от несанкционированного доступа: при запуске приложения программа запрашивает пароль (рисунок 12).

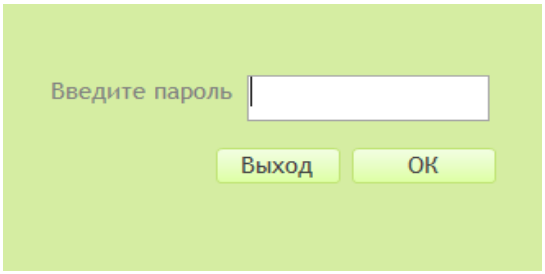


Рисунок 12 – Всплывающее окно при открытии базы данных

На рисунке 13 представлен фрагмент кода функции «Войти в программу». Так же на рисунке 14 представлена блок-схема функции «Войти в программу».

```
Option Compare Database

Private Sub Button_Click()
If (Me.password.Value = "12345") Then
            MsgBox "Верный пароль"
            Me.password.Value = ""
            DoCmd.OpenForm "Кнопочная форма"
        Else
            MsgBox "Неверный пароль"
        End If

End Sub
```

Рисунок 13 – Фрагмент кода функции «Войти в программу»

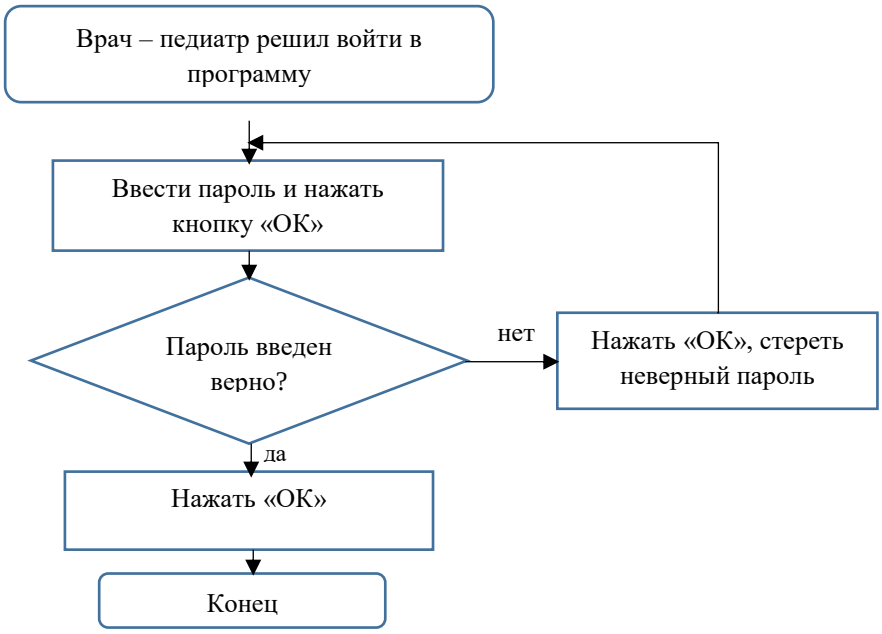


Рисунок 14 – Блок-схема функции «Войти в программу»

Четвертый спринт так же направлен на разработку интерфейса и навигацию. Конечный интерфейс разработанного приложения выглядит следующим образом (рисунок 15).

Рисунок 15 – Главное меню разрабатываемой программы

Главное меню состоит из пяти кнопок: «Принять ребенка», «Лечить ребенка», «Выписать ребенка», «Отчеты» и «Выход». Далее были добавлены кнопки навигации (почти в каждом интерфейсе есть кнопка «Назад» или «Выход», а также «Далее»). Название кнопки «Выход» говорит само за себя, данная кнопка позволяет закрыть программу. В итоге получено приложение с главным меню для врача-педиатра, содержащее 5 кнопок.

ЗАКЛЮЧЕНИЕ

В данной работе были изучены теоретические основы и ключевые особенности моделирования бизнес-процессов в различных нотациях.

Разработана медицинская информационная система для врача-педиатра. Возможности данной системы позволяют отражать данные о ребенке не только в виде традиционных разделов медицинской карты, но и представлять информацию в интересующих средах. Доступ к базе данных происходит через интерфейс клиентского программного приложения врача-педиатра.

Программное обеспечение позволит врачу-педиатру преобразовать процесс выдачи рецептов на молочную кухню. Таким образом, в рамках данной работы была изучена и применена одна из самых популярных методологий гибкой разработки Scrum. Для реализации ПО были идентифицированы способы анализа требований и выявлены пользовательские и функциональные требования. После этого был составлен план создания программного обеспечения, согласно методологии SCRUM. Всего получилось 5 спринтов. В следующим пункте создания приложения были составлены модели AS-IS и TO-BE ключевого бизнес-процесса с помощью нотаций ARIS VACD и eEPC.

Разработка приложения производилась в среде создания и обработки баз данных MS Access. Эта программа является простой и понятной в обращении; имеет возможность внесения данных; является полностью совместимой с системой Windows и обладает огромными возможностями по импорту и экспорту данных. По итогу в MS Access были реализованы требования, выявлены недочеты и исправлены ошибки.

Таким образом, цель данной работы была достигнута, все поставленные задачи – выполнены полностью. Стоит отметить, что данная работа объединяет в себе большой объем информации по различным разделам экономики и автоматизации. Анализ и применение этой информации на практике позволяет решать, как существующие, так и перспективные проблемы.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ И ИСТОЧНИКОВ

1. Варзунов А. В. Анализ и управление бизнес процессами − СПб.: Университет ИТМО, 2016. − 112 с.
2. Сазерленд Д. Scrum. Революционный метод управления проектами Scrum − М.: Манн, Иванов и Фербер, 2016. − 288 с.
3. Морозова В. И., Врублевский К. Э. Моделирование бизнес-процессов с использованием методологии ARIS/ Учебно-методическое пособие − М.: РУТ (МИИТ), 2017. − 47 с.
4. Куликов С. С. Тестирование программного обеспечения. Базовый курс/ Практическое пособие − Минск: Четыре четверти, 2015. − 294 с.
5. Леонов В. Простой и понятный самоучитель Word и Excel − М.: Эксмо, 2016. − 354 с.
6. Ульман Дж. Основы систем баз данных − М.: Финансы и статистика, 2017. − 292 с.
7. Степанов Д. Ю. Анализ, проектирование и разработка корпоративных информационных систем: уровень бизнес-процессов / М.: МИРЭА, 2017. [Интернет − ресурс] http://stepanovd.com/training_erp_1−7_ru.pdf (Дата обращения 08.05.2019)
8. Грин Д., Стеллман Э. Постигая Agile. Ценности, принципы, методологии − М.: Манн, Иванов и Фербер, 2018. − 448 с.
9. Многопрофильный медицинский и диагностический центр GMS Clinic [интернет − ресурс] http://gmsclinic.ru/specialists/pediatrician (Дата обращения 10.05.2019)
10. Смирнова Н. Н., Белозерцева В. Н., Сорока Н. Д. Педиатрия для семейного врача/ Справочник − М.: СпецЛит, 2017. − 71 с.

ПРИЛОЖЕНИЕ А

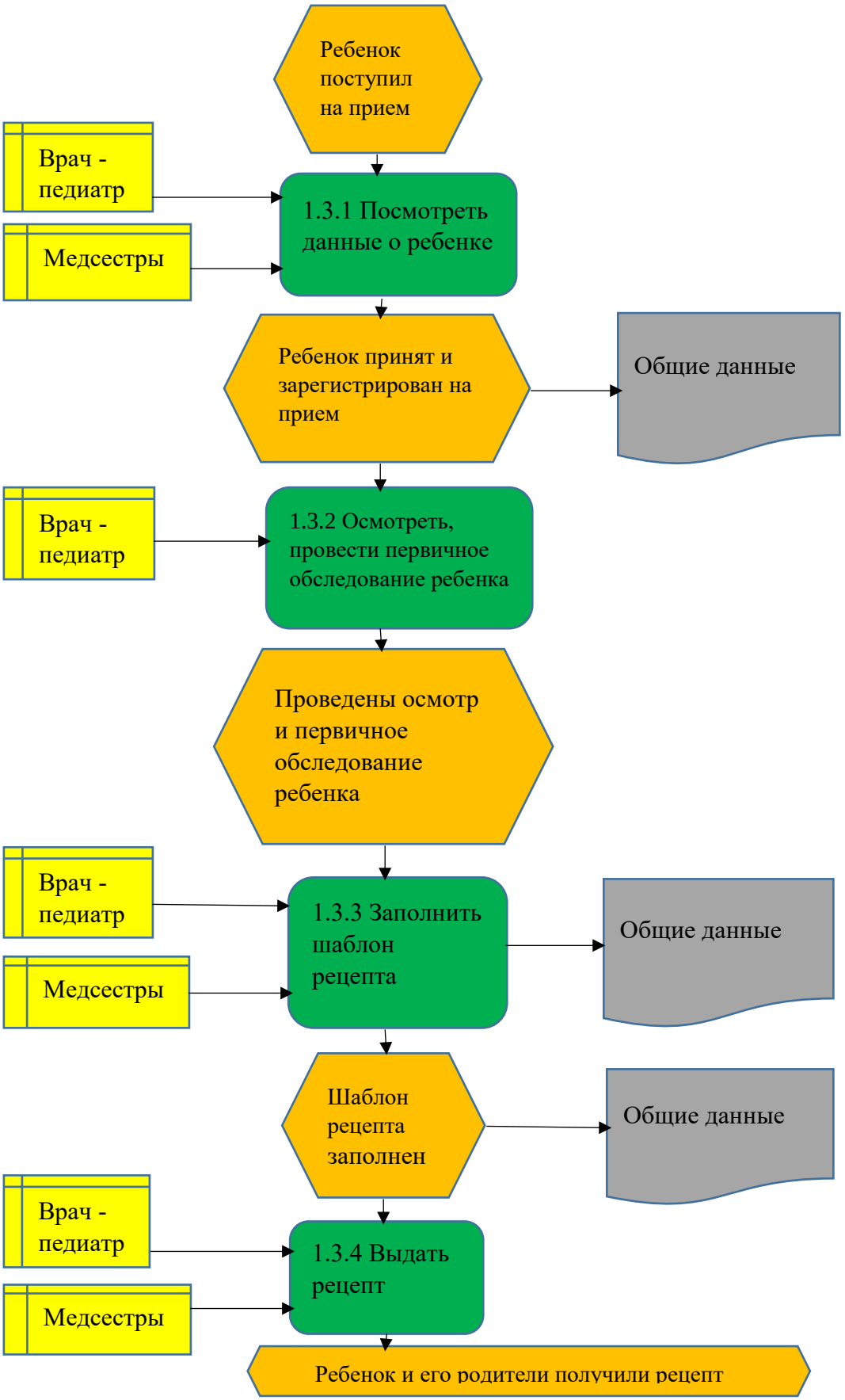


Рисунок А.1 – Второй уровень описания процесса «Выдать рецепт»

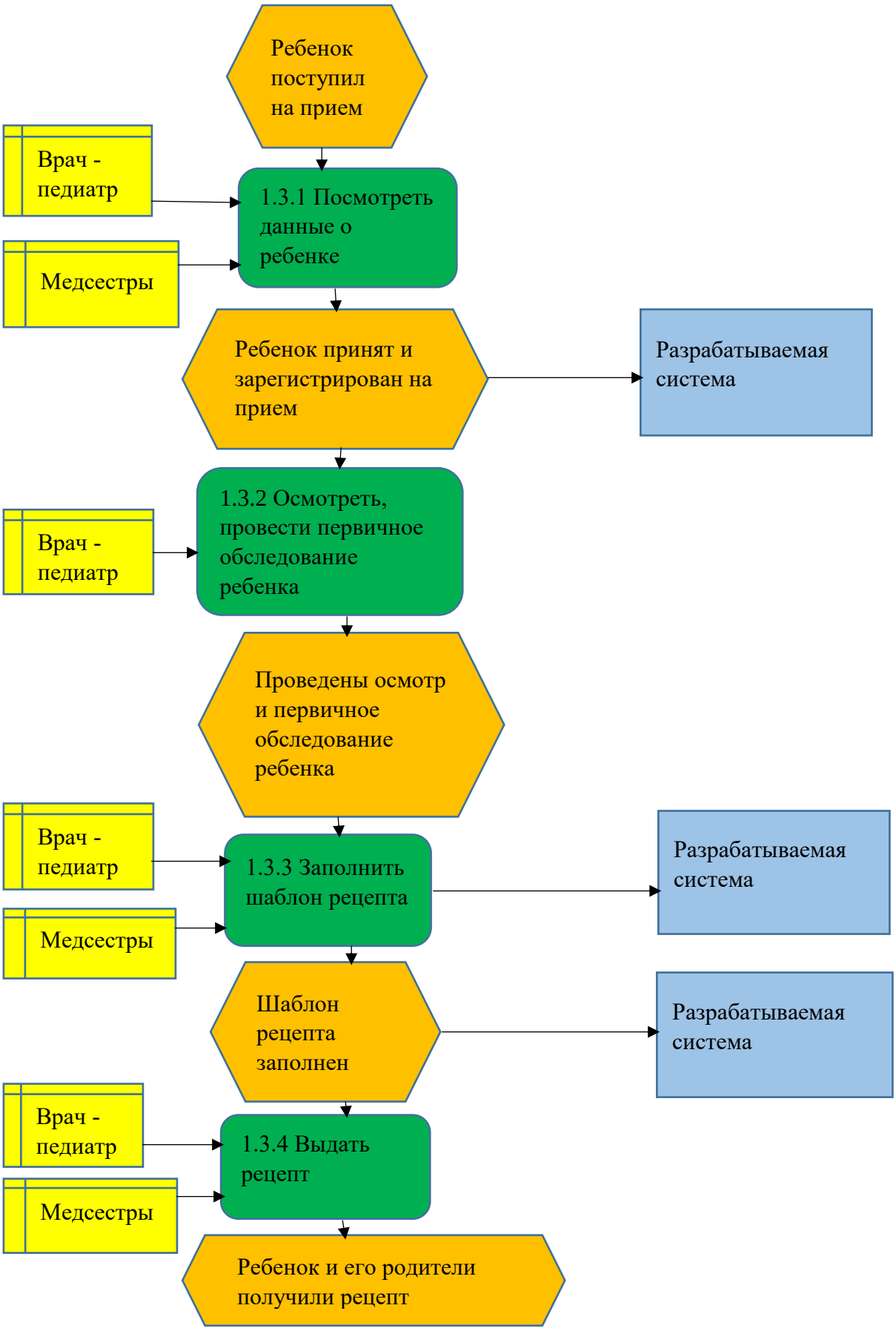


Рисунок А.2 – Второй уровень описания процесса «Выдать рецепт»

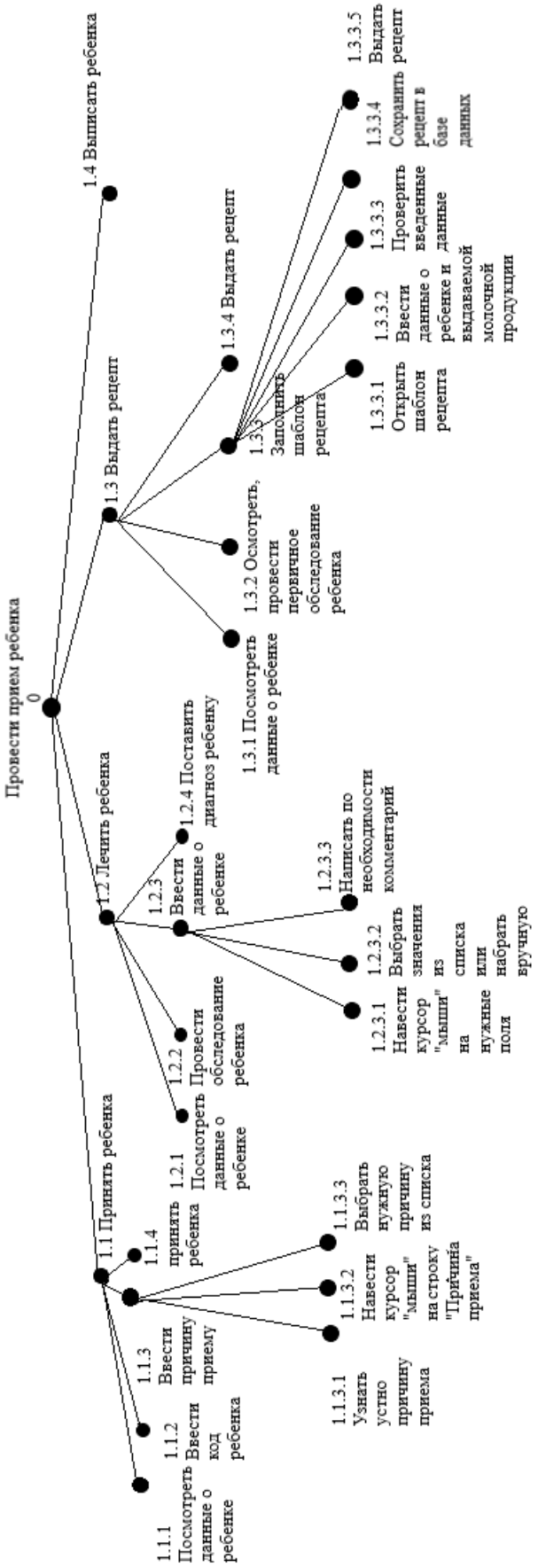


Рисунок А.3 – Карта бизнес-процессов