Методы распознавания в задачах определения лекарственных препаратов с применением итерационной модели внедрения

Колосов Иван Алексеевич

Аннотация: в статье показана разработка компьютерной программы, автоматизирующей действия врача по подбору лекарственного препарата. Статья демонстрирует использование автоматизированных методов распознавания в задачах автоматического определения лекарственных препаратов на основе данных пациента. В ходе работы были изучены действия врача, составлены требования к создаваемой системе. Разработка рабочего прототипа программы велась по итерационной модели. В качестве ключевых технологий были использованы язык программирования Python и система управления базами данных SQLite.

Введение

В настоящее время в системе здравоохранения закрепился принцип доказательной медицины, позволяющий путем исследований выявить особенности заболеваний и эффективные способы их лечения. Вместе с тем существуют проблемы данного принципа и его слабые места. Одной из них является большое время и трудозатратность проведения исследований, что влечет ряд компромиссов в виде уменьшения выборки, увеличения финансирования, проведения исследований силами специальных исследовательских лечебных заведений и др. Решение этих трудностей – лишь один из путей повышения полноты и достоверности знаний об эффективности лечения.

Другой путь, дополняющий оптимизацию медико-биологических исследований – первичный статистический анализ и использование данных, которые по тем или иным причинам не исследуются. Такими данными являются физиологические показатели пациентов, параметры симптомов новых или редких заболеваний, данные лечения сочетаний заболеваний. Своевременные обработка и использование результатов анализа этих данных должно помочь принимать решения в процессе работы как отдельного врача или больницы, так и системы здравоохранения в целом, указывая упущенные из внимания особенности и закономерности болезней и их лечения.

1. Цель работы

Цель работы состоит в использовании методов распознавания в задачах автоматизированного определения лекарственных препаратов на основе данных пациента. Для достижения цели требуется изучить:

- графические нотации проектирования процессов;
- структуру описания пользовательских программ и виды тестирования программных продуктов;

а также практически выполнить:

- идентифицировать требования к процессам обучения и тестирования классификатора по предложению лекарственных средств;
- спроектировать процессы в IDEF0 и IDEF3 для моделей AS-IS и TO-фёBE до 3-4 уровней детализации;
- реализовать и количественно оценить программное приложение для автоматизации работы больницы в среде Python.

Объектом текущего исследования является проблема использования ИТ/ИС в медучреждениях, предметом исследования – работа врача по подбору лекарственного препарата по данным пациентов. Практическая ценность исследования заключается в описании проблемы и опыта ее решения с использованием исследуемых математических методов в процессе разработки информационной системы.

2. Анализ методов распознавания образов

Методы распознавания образов – математический аппарат, предназначенный для классификации и верификации объектов реального мира из выборки по их параметрам. Класс – это множество объектов с общими свойствами (параметрами), существенными для решения конкретной задачи над объектами предметной области. При этом, объектам класса дается метка класса. Например, в гражданской авиации для приема пассажиров терминалом аэропорта требуется знать тип самолета, классифицировав его по размеру, требуемому трапу, набору оборудования, к обслуживанию которого надо быть готовым техническому персоналу.

Классификация – процесс причисления объекта к какому-либо классу по его параметрам (приписывание метки класса). Верификация – сопоставление объекта с описанием (моделью) класса. Объектами класса могут служить как отдельные объекты, которые можно сосчитать, так и дискретные отсчеты непрерывной величины (например, двумерного изображения, дробящегося на пиксели для их классификации). В лю-

бом случае, классифицируемый объект представляется вектором y = (x_0, x_1, … , x_{N-1}), где x – значение признака, а N – количество признаков (N-мерное пространство). Признак может быть любым значением: числом, значением алфавита формального языка, матрицей или иным объектом.

В задачах классификации большую роль играет сама выборка. Во-первых, она должна быть репрезентативна: количество ее членов должно быть достаточно большим, чтобы ее свойства не отличались от свойств генеральной совокупности. Во-вторых, позиция элемента в выборке может быть одним из параметров объекта. То есть, пусть у нас есть выборка из элементов y_0, y_1, … y_n, тогда y_i = (i, x x_1, x_2, …, x_N), где x ∈ (0;N-1).

Рассмотрим метод ближайшего соседа. Метод является метрическим и причисляет элемент выборки к классу, к которому принадлежит ближайший к нему элемент. Мера близости – величина, вычисляемая из параметров классифицируемого объекта и параметров классифицированных объектов выборки:

$$M = \sqrt{\sum_{i=0}^{N-1}(x_i - z_i)^2}\,, \qquad (2.1)$$

здесь M – мера расстояния, x_i – параметр классифицируемого объекта, z_i – параметр ранее классифицированного объекта, N – размерность пространства признаков (их количество).

Если была найдена минимальная M, тогда определяется, к какому классу принадлежит z и x причисляется к этому же классу. Мера ищется разными способами. Примененный нами вариант описан в разделе разработки и тестирования. Этот метод неустойчив к шумовым выбросам и наиболее применим в случае многотонного изменения признака. Для купирования этого недостатка разработаны его дополнения. Без дополнений возникает необходимость ручного отсеивания данных выборки исходя из предположения, что параметры классифицируемого объекта изменяются монотонно и делят пространство параметров на отчетливые области [1].

3. Итеративная модель разработки

Программный продукт будет разрабатываться по итерационной модели. Данная модель наилучшим способом подходит к текущей задаче, так как снижает риски, характерные для проектов по внедрению информационных систем [2-5]. Суть итерационной мотели состоит в разбиении процесса разработки продукта на отдельные ста-

дии (итерации). Размер каждой итерации выбирается таким образом, чтобы на каждой полностью завершалась часть работы, без которой дальнейшая разработка затруднена. Итерации не завершаются пока не будут выполнены все требования части продукта. Для этого сами итерации разделяют на этапы: планирование и формулировка требований, реализация, тестирование, принятие решения об успехе итерации (рис. 3.1). Опишем содержание итераций:

- сразу после начального планирования и в начале каждой итерации проводится бизнес планирование, результатом которого являются требования;
- второй шаг состоит из анализа сформулированных на предыдущем этапе требований, проектировании и реализации продукта с их учетом. В случае, если это происходит на последней итерации, полученный продукт выпускается по завершении;
- далее проводится тестирование для оценки получившегося прототипа (частей) продукта;
- и, наконец, оценка результатов тестирования для последующего планирования.

Рис. 3.1. Этапы одной итерации

4. Идентификация требований

Перед разработкой программы требуется собрать требования заказчика [6] чтобы понять, возможно ли создать программный продукт [7]. По технике приоритезации MuSCoW требования можно разделить на типы по важности:

- Must have (обязательные) – требования с наибольшим приоритетом, без выполнения которых релиз продукта невозможен.
- Should have (желательные) – высокоприоритетные требования, которые критичны для функционала;
- Could have (возможные) – требования, которые можно включить в текущий релиз, но которые не влияют существенно на его успех;
- Won't have (отсутствующие) – требования, которые не являются необходимыми в текущем релизе, но которые можно включить в следующие.

Таблица 4.1. Таблица требований

№	Пользовательское требование	Функциональное требование	Программный компонент	Приоритет требования согласно MuSCoW
1	Хранение параметров пациентов	Таблица параметров пациентов	База данных SQLite с таблицей данных	Must have (должно быть)
2	Хранение названий параметров пациентов	Таблица названий параметров пациентов	База данных SQLite с полем названий параметров	Must have (должно быть)
3	Хранение названий применимых лекарств	Таблица названий применимых лекарств	База данных SQLite с полем лекарств	Must have (должно быть)
4	Обучение классификатора на основе метода распознавания образов при помощи обучающей выборки	Алгоритм расчета параметров (весов) классификатора	Программный код на ЯП Python, выполняющий расчет параметров (весов) классификатора	Should have (желательно)

№	Пользовательское требование	Функциональное требование	Программный компонент	Приоритет требования согласно MuSCoW
5	Применение классификатора классификации записей классифицируемой выборки	Классификатор, использующий метод Распознавания образов и вычисленные раннее параметры (веса)	Программный код классификатора на ЯП Python, использующий метод распознавания образов и вычисленные раннее параметры (веса)	Should have (желательно)
6	Создание выборки для новой медицинской задачи с произвольными параметрами и препаратами	Создание таблицы с произвольны ми атрибутами параметров списком лекарств	Окно «Создать новую базу данных» одноименное окно	Could have (возможные)
7	Ввод данных в классифицируемую выборку (таблицу)	Возможность создания записей замеров параметров пациентов	Режим главного окна для ввода данных и запускающий его пункт меню	Could have (возможные)
8	Редактирование данных обучающей выборки (таблицы)	Возможность вносить изменения в поля таблицы обучающей выборки.	Режим главного окна для редактирования таблицы обучающей БД и пункт меню для его запуска	Could have (возможные)
9	Классификация записей пациентов их таблицы с целью выяснения подходящего для лечения препарата	Возможность применения классификатора к записям	Режим главного окна для классификации записей таблицы классифицируем-ой БД и пункт меню для его запуска	Could have (возможные)

№	Пользовательское требование	Функциональное требование	Программный компонент	Приоритет требования согласно MuSCoW
10	Подсказки для пользователя	Возможность получить информацию о порядке пользования программой	Подсказка в панели статуса главного окна	Won't have (отсутствующие)

5. Моделирование бизнес-процессов

Для описания процессов воспользуемся двумя нотациями проектирования: IDEF0 для высокоуровневого проектирования и IDEF3 – низкоуровневого. Эти нотации проектирования выбраны из-за своей применимости к описанию бизнес-процессов в области медицины [8]. Но для начала, разберемся, что это за нотации, для чего используются и чем отличаются друг от друга.

Нотация IDEF0 используется для создания функциональной модели, отображающей структуру и функции системы, а также потоки информации и материальных объектов, связывающие эти функции. IDEF3 предназначена для низкоуровневого моделирования. Низкоуровневое проектирование в отличие от высокоуровневого не рассматривает элементы процесса с точки зрения наличия самого факта воздействия одного элемента на другой. В нотациях нижнего уровня указываются логические связи между процессами, от состояния которых ведется манипуляция объектами [9]. Произведем декомпозицию процессов, происходящих в больнице при выборе лекарственного препарата (рис. 5.1-5.4).

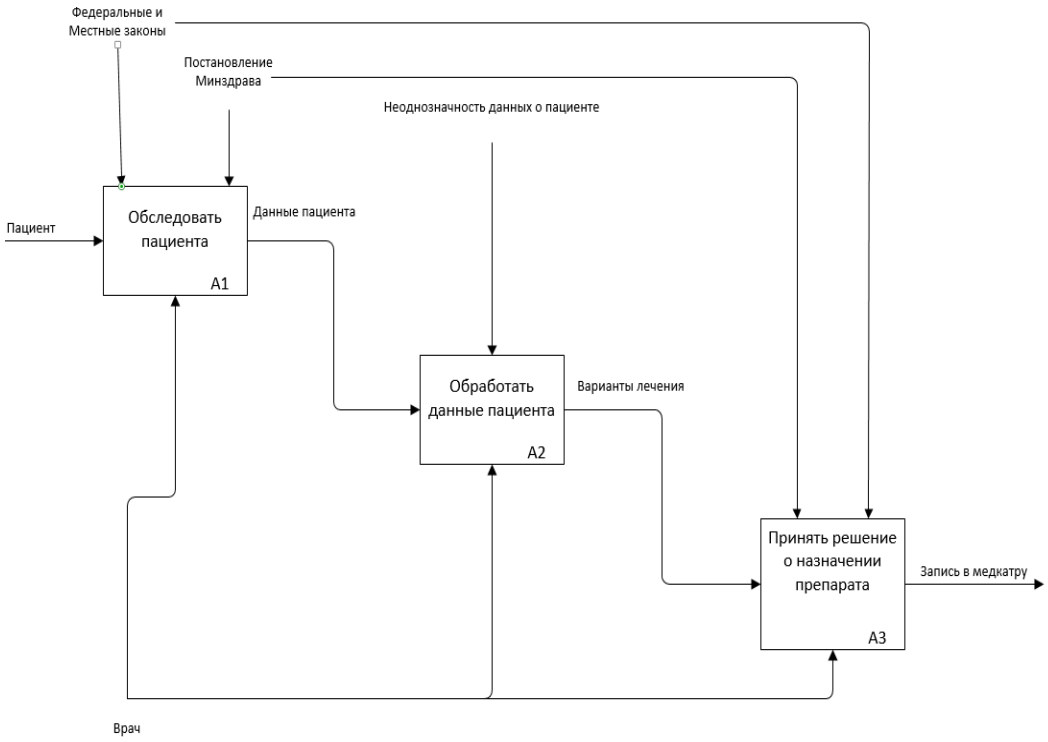


Рис. 5.1. *Начальный уровень описания процессов в модели AS-IS*

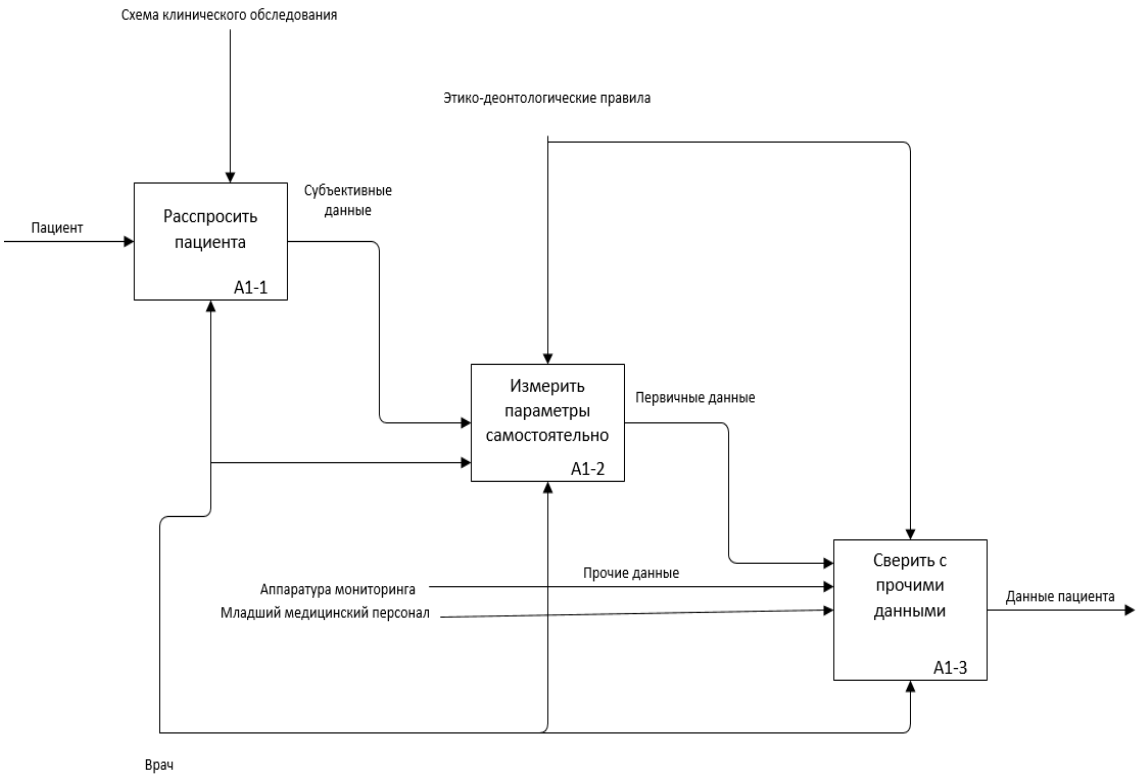


Рис. 5.2. Первый уровень описания подпроцесса A1 в модели AS-IS

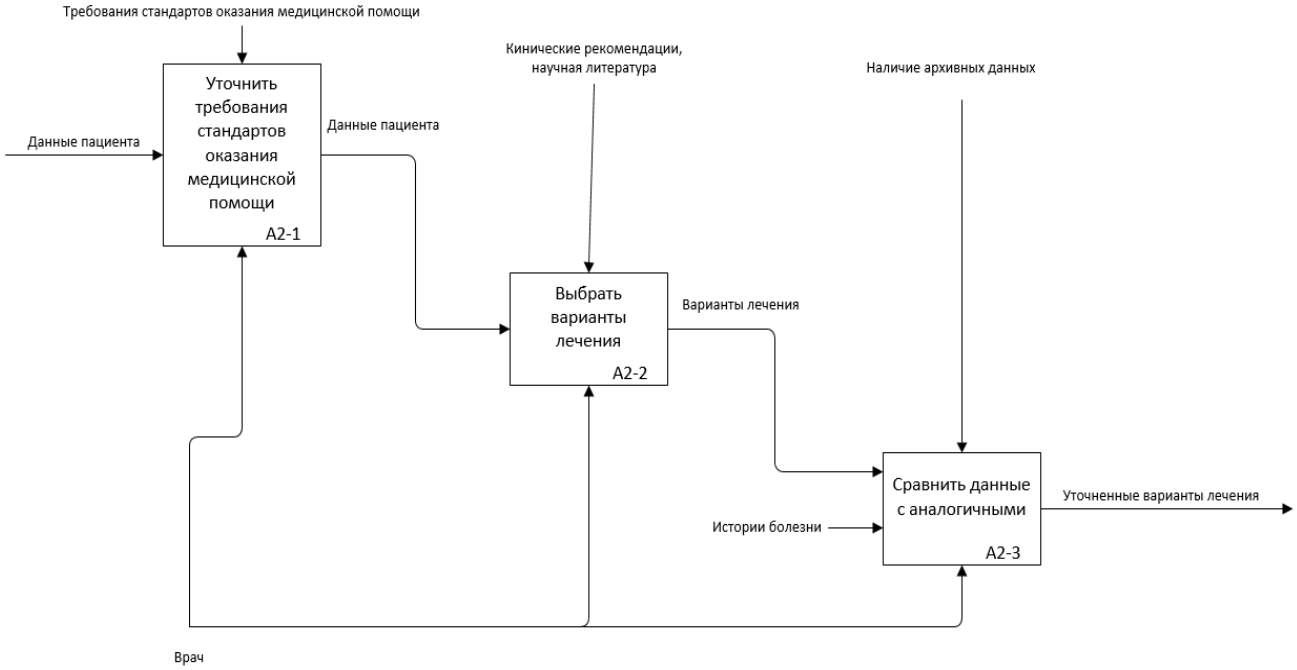


Рис. 5.3. Первый уровень описания подпроцесса A2 в модели AS-IS

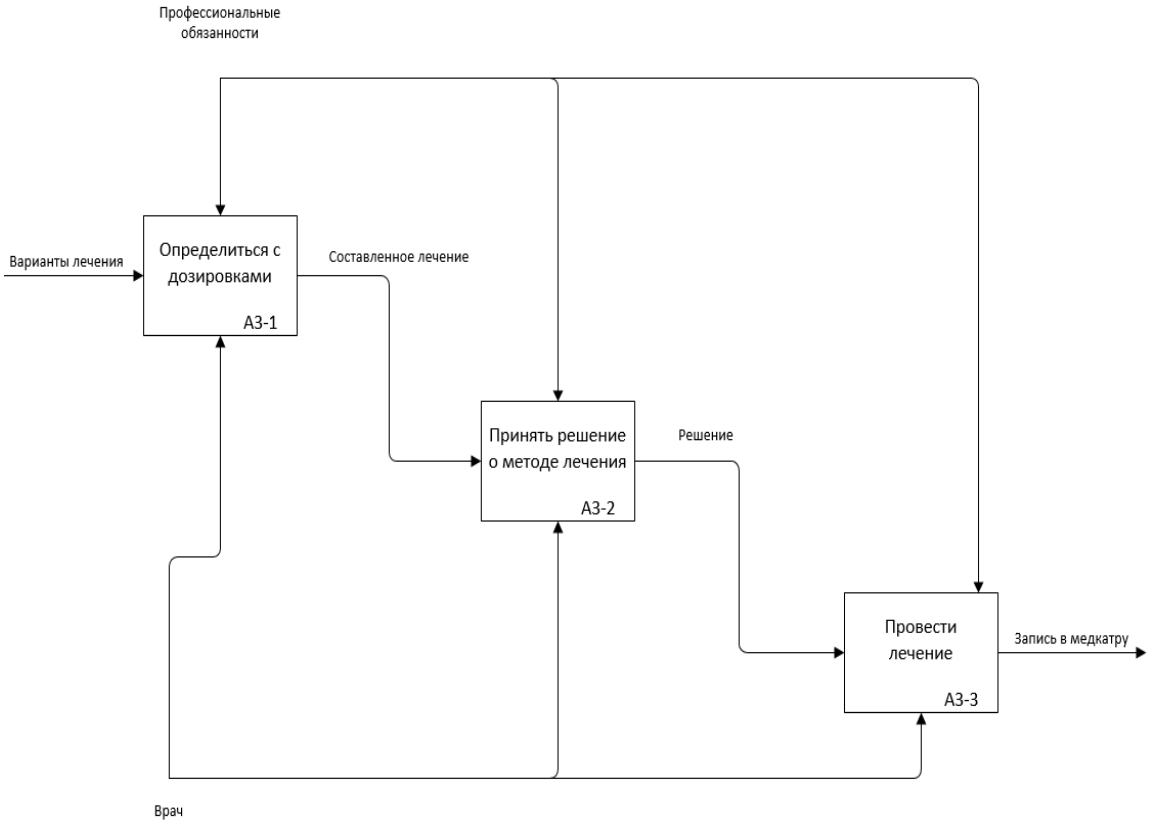


Рис. 5.4. Декомпозиция подпроцесса A3 в модели AS-IS

Подпроцесс принятия решения требует детального рассмотрения участвующих в нем источников информации, представленных в виде эмпирических данных в историях болезни и научных данных из клинических рекомендаций и прочих источников. Поэтому для его описания на низком уровне лучшим образом подходит нотация IDEF3 (рис. 5.5-5.6).

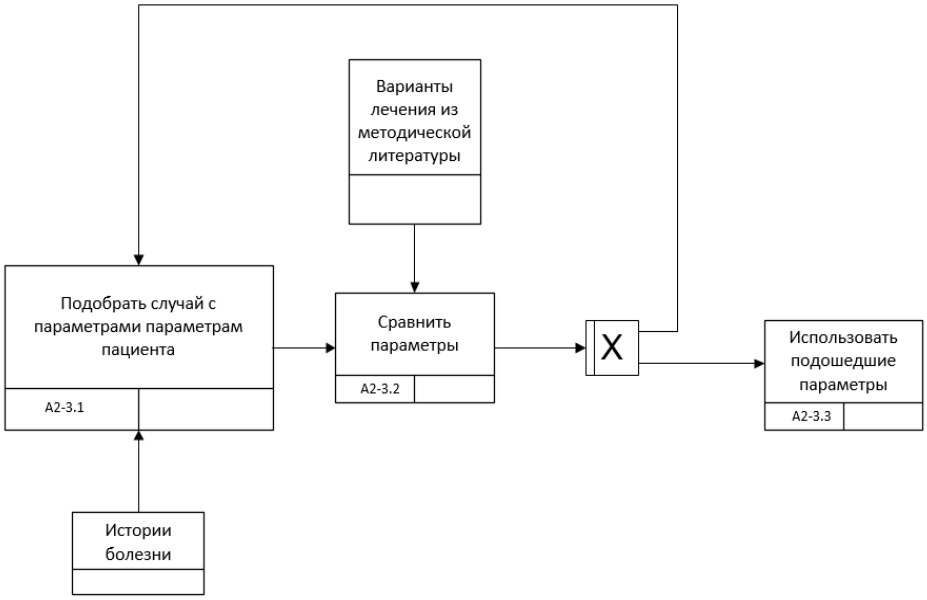


Рис. 5.5. Декомпозиция подпроцесса A2.3 в нотации IDEF3 и модели AS-IS

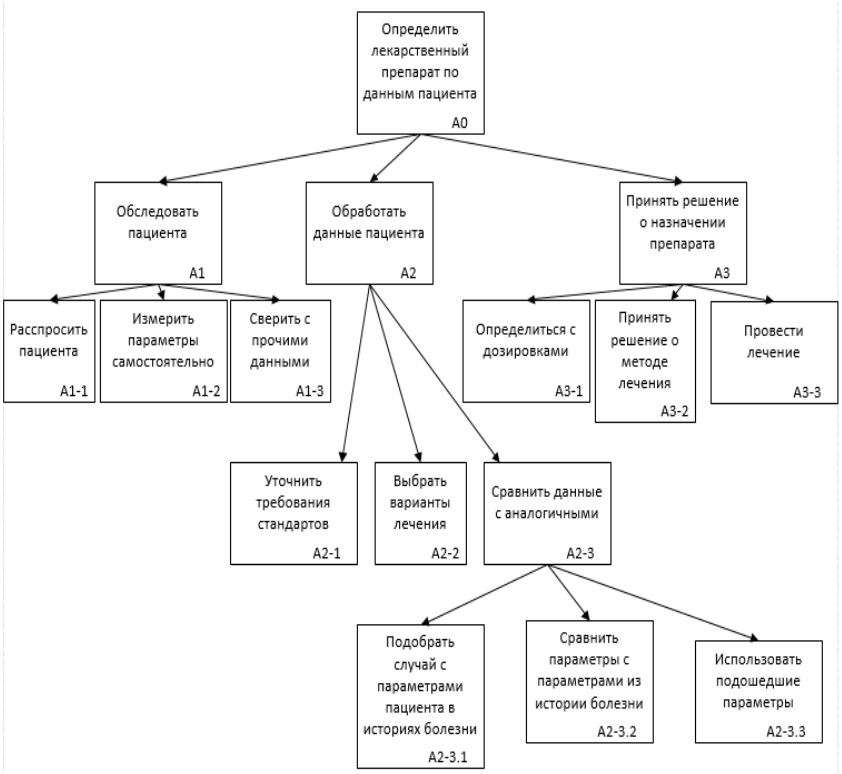


Рис. 5.6. Карта процессов в модели AS-IS

Внедряя систему выбора методики лечения, мы возлагаем работу по обработке данных пациента на нее. Процесс A2.3 опишем в нотации IDEF3 (рис. 5.7-5.9).

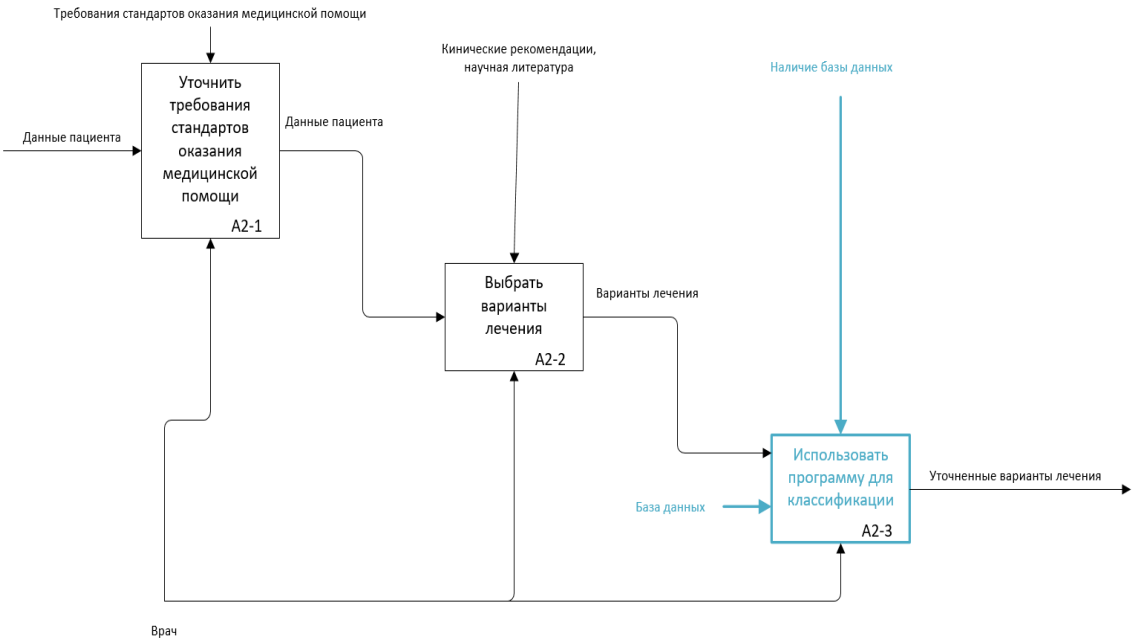


Рис. 5.7. Декомпозиция подпроцесса A2 в модели TO-BE

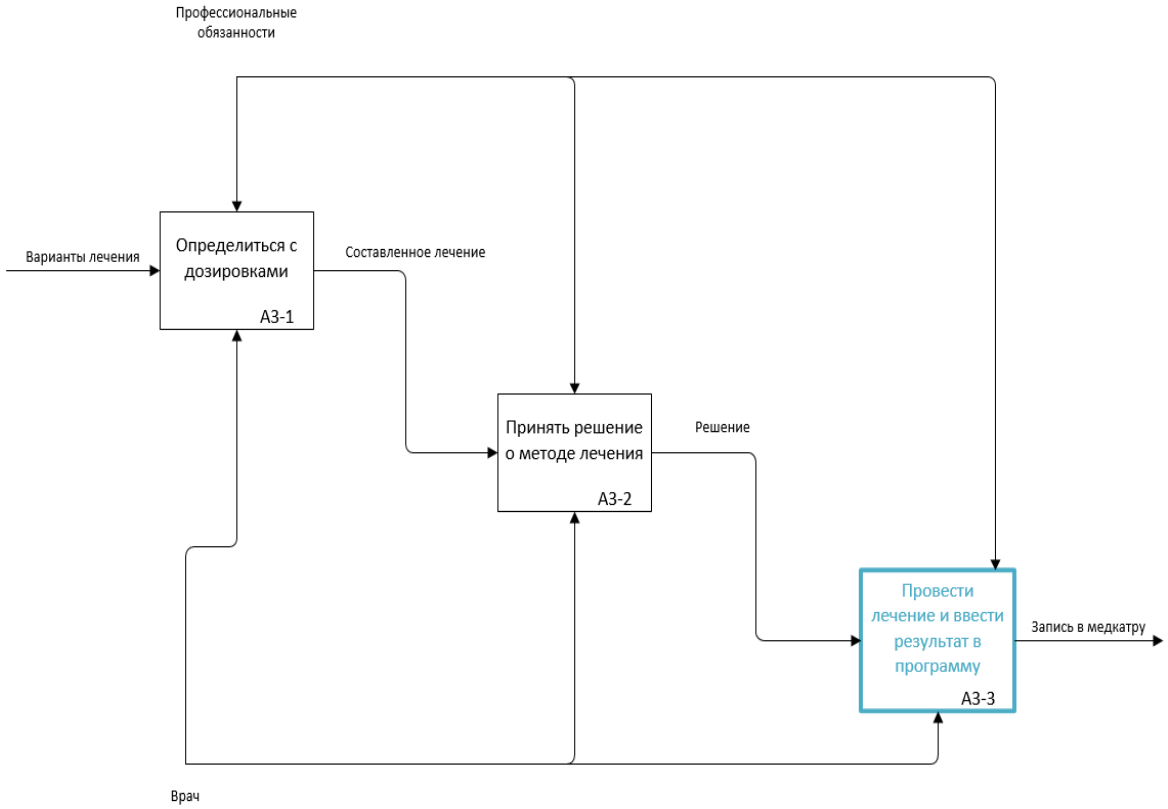


Рис. 5.8. Декомпозиция подпроцесса А3 в модели TO-BE

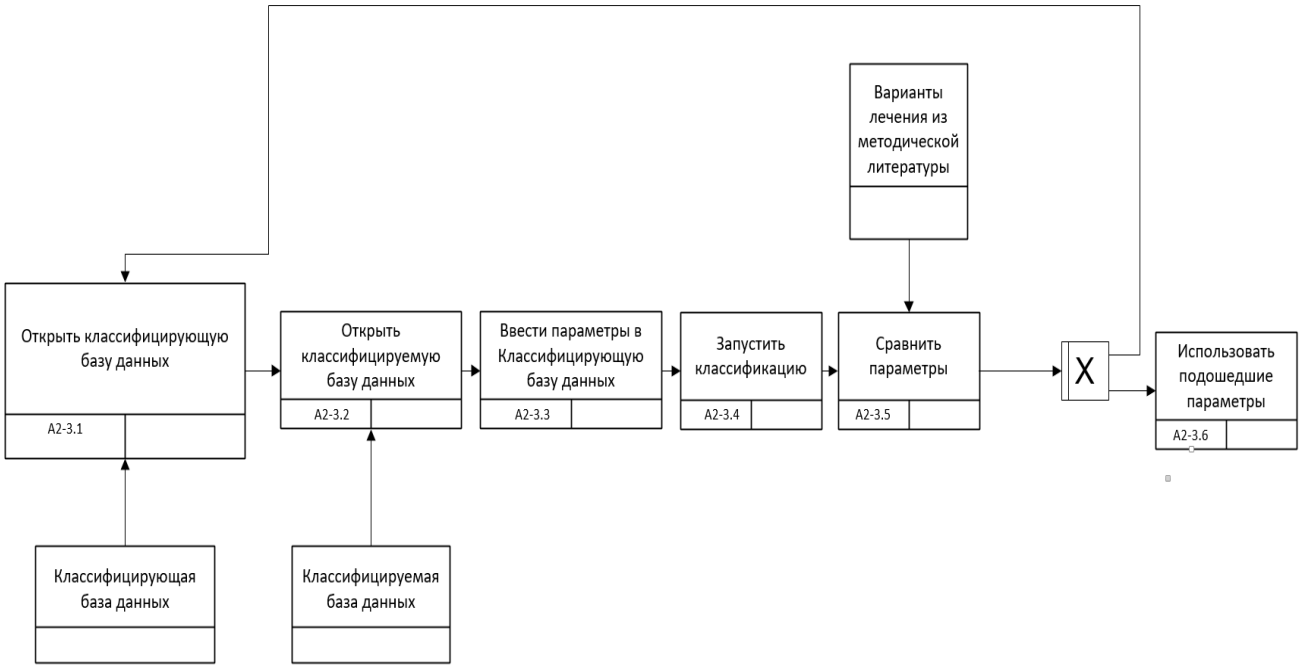


Рис. 5.9. Процесс А2.3 в нотации IDEF3 и модели TO-BE

Автоматизация подбора метода лечения основана на методе распознавания, одним из главных свойств которого является возможность обучаться на выборках. Для того чтобы задействовать эту возможность надо внести изменения в подпроцесс А3 и провести дополнительную декомпозицию для описания логики работы с программой в нотации IDEF3 (рис.5.10-5.11).

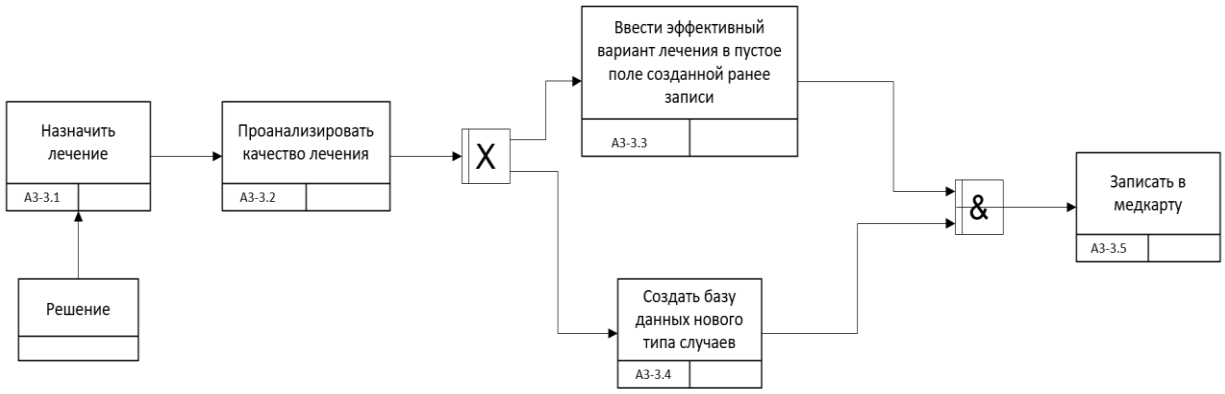


Рис. 5.10. Процесс А3-3 в нотации IDEF3 и модели TO-BE

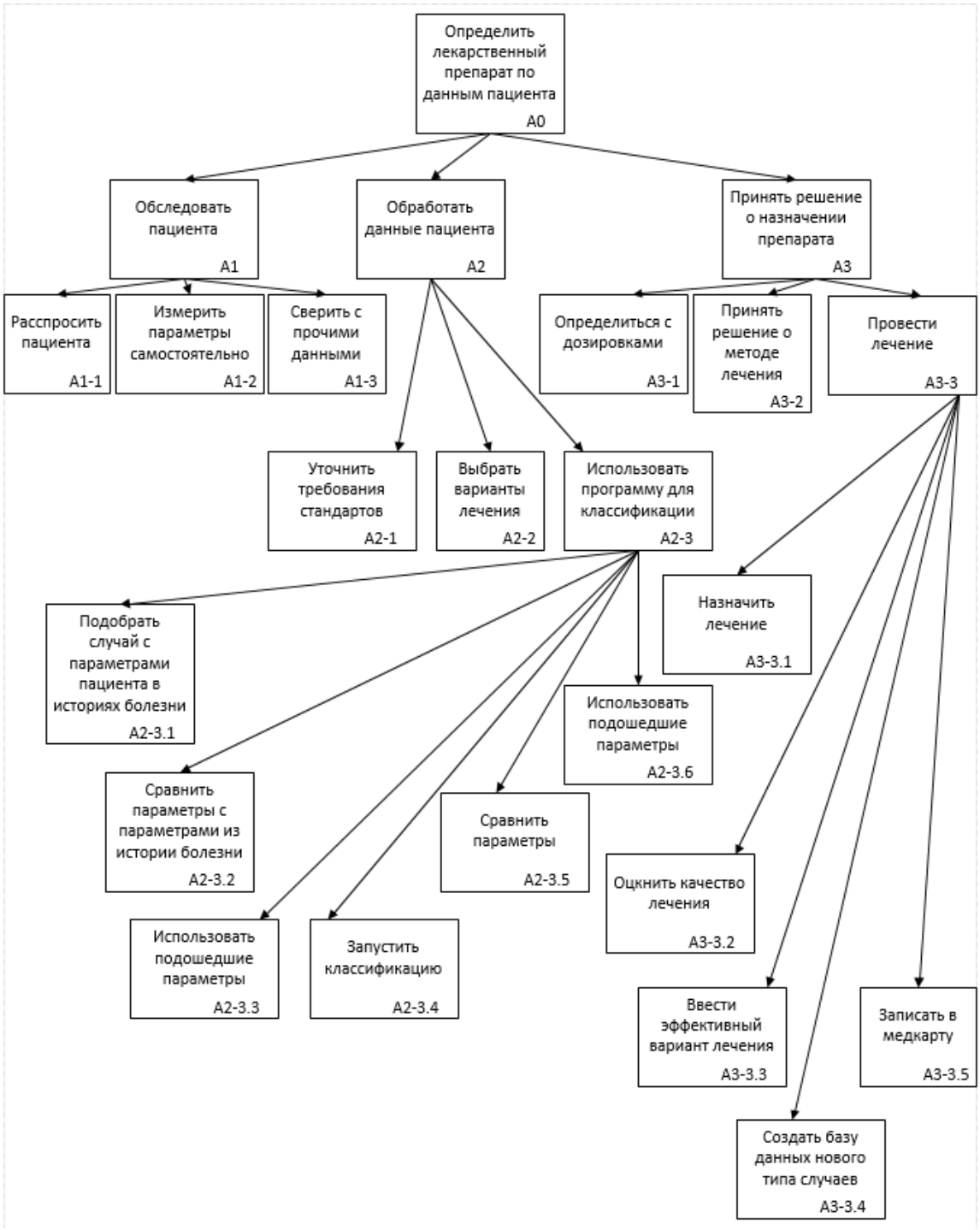


Рис. 5.11. Карта процессов в модели TO-BE

6. Первый прототип программы (приоритет Must)

На первой итерации проектируется архитектура данных (рис. 6.1). Имеем список пациентов (обучающую выборку) с указанием достоверно правильной методики лечения, исходя из историй болезни. Номер методики – класс, к которому надо причислить пациентов из второй, изучаемой выборки, для которых методика не известна или требует уточнения (рис. 6.2). Данные о разрешенных для ввода лекарствах (классах) и названиях параметров хранятся в двух текстовых полях отдельно сознанной таблицы метаданных. Для разработки типовой базы данных была применена открытая СУБД SQLiteStudio.

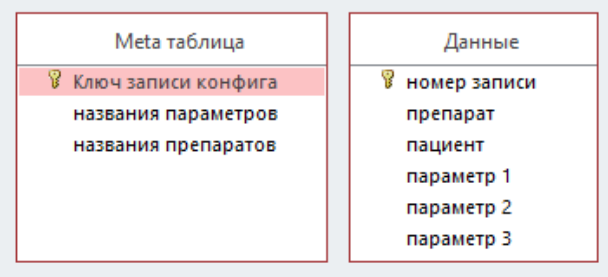


Рис. 6.1. Схема данных

sampleid	patientid	temperature	pain	cirrhosis
1	0	36.6	94.0	2.6
2	1	36.3	96.0	2.1
3	2	36.1	96.0	2.6
4	3	36.4	91.0	2.5
5	4	36.6	90.0	2.3
6	5	36.0	93.0	2.3
7	6	36.6	96.0	2.6
8	7	36.6	94.0	2.5
9	8	36.1	90.0	2.6
10	9	36.5	95.0	2.4
11	10	36.6	92.0	2.2
12	11	36.4	91.0	2.3
13	12	36.4	90.0	2.4
14	13	36.1	95.0	2.4
15	14	36.6	94.0	2.3
16	15	36.3	96.0	2.2
17	16	36.6	91.0	2.6
18	17	36.0	90.0	2.4
19	18	36.3	94.0	2.3
20	19	36.1	93.0	2.6

Рис. 6.2. Таблица данных классифицируемой базы данных (данные для проведения тестирования)

В обучающей выборке указан номер пациента, его класс и параметры, определяющие этот класс, например, для купирования боли и снижения температуры больных, страдающих циррозом печени значимыми параметрами, являются температура, субъективное ощущение боли по шкале от 1 до 100 и степень печеночной недостаточности по шкале Чайльда-Пью от 1 до 15 баллов (рис. 6.3).

sampleid	classname	patientid	temperature	pain	cirrhosis
1	Aspirin	0	36.3	96.0	2.3
2	Aspirin	1	36.1	90.0	2.6
3	Aspirin	2	36.4	95.0	2.0
4	Aspirin	3	36.6	92.0	2.4
5	Aspirin	4	36.3	91.0	2.0
6	Aspirin	5	36.5	93.0	2.0
7	Aspirin	6	36.4	96.0	2.5
8	Aspirin	7	36.5	91.0	2.1
9	Aspirin	8	36.1	95.0	2.3
10	Aspirin	9	36.1	92.0	2.2
11	Aspirin	10	36.4	90.0	2.1
12	Aspirin	11	36.1	96.0	2.0
13	Aspirin	12	36.5	96.0	2.3
14	Aspirin	13	36.2	93.0	2.0
15	Aspirin	14	36.6	92.0	2.1
16	Aspirin	15	36.6	93.0	2.2
17	Aspirin	16	36.6	90.0	2.5
18	Aspirin	17	36.0	95.0	2.5
19	Aspirin	18	36.0	93.0	2.5
20	Aspirin	19	36.4	93.0	2.3

Рис. 6.3. Таблица данных классифицирующей выборки (данные по результатам обучения)

7. Второй прототип программы (приоритет Could)

На второй итерации ведется реализация математических методов распознавания. Первая задача из списка требований – расчет весов классификатора. Обладая 3-я параметрами для каждого пациента, измеряем различия (расстояния) в каждом параметре по формуле:

$$r = |p_1 - p_2| / \max(p_1, p_2). \quad (7.1)$$

Далее реализуем классификатор, рассчитывающий веса (численно выра-женные коэффициенты значимости физиологических параметров пациента). Программа позволяет выполнять вычисление меры расстояния между элементами обучающей выборки с использованием весов (рис. 7.1).

температура

	sn1cl0	sn2cl0	sn3cl0
sn1cl0	0.0	0.0	0.002762
sn2cl0	0.0	0.0	0.002762
sn3cl0	0.002762	0.002762	0.0

Рис. 7.1. Таблица расстояний по параметру температуры

Следуя алгоритму, вычисляем, с какой частотой (ошибкой) для каждого параметра его значение является ближайшим до пациента другого класса. Исходя из этого (ошибки), считаем вес каждого параметра W в классификаторе так, что сумма весов равна единице. Веса – основные параметры классификатора, задаются формулой:

$$W = (1 - E_i)/\sum_{j=1}^{k} E_j \ . \qquad (7.2)$$

Весовые коэффициенты, найденные на этапе обучения даны на рис. 7.2.

```
Ошибки для каждого типа параметров
[0.35000000000000003, 0.26499999999999996, 0.22]
Вес для каждого типа параметров
0.300230946882217
0.33949191685912244
0.36027713625866054
Процент ошибки классификатора = 7%
```

Рис. 7.2. Вывод программой весовых коэффициентов классификатора

Ошибка классификации как на этапе обучения определяется по формуле ниже:

$$E = \sum_{i=1}^{k} W_i r_i \ . \qquad (7.3)$$

Расчет меры различия ведется с использованием весовых коэффициентов, найденных на этапе обучения. Принятие решения о принадлежности элемента из тестовой выборки эталонному элементу классификатора осуществляется на основе минимальной меры различия, что является отличительной чертой метода ближайшего соседа:

$$R = \sum_{i=1}^{k} \frac{r_i}{W_i} = min \ . \qquad (7.4)$$

Пример классификации элемента тестовой выборки дан на рисунке ниже (рис. 7.3). Для пациента с температурой 36.4 градуса, уровня более 91% и степени печеночной недостаточности 2.1 было определено лекарство аспирин, ранее предложенное для другого пациента со схожими симптомами.

sampleid	patientid	temperature	pain	cirrhosis
1	0	36.4	92.0	2.1
2	1	36.6	93.0	2.0
3	2	36.0	95.0	2.4
4	3	36.2	90.0	2.4
5	4	36.5	96.0	2.3
6	5	36.1	92.0	2.3
7	6	36.5	92.0	2.3
8	7	36.4	91.0	2.1
9	8	36.0	91.0	2.2
10	9	id похожей записи6		2.1
11	10	название класса похожего пациентаAspirin		2.6
12	11	5		2.3
13	12	temperature36.5		2.2
14	13	pain91.0		2.2
15	14	cirrhosis2.2		2.3
16	15			2.0
17	16			2.0
18	17	36.1	93.0	2.5
19	18	36.1	94.0	2.2
20	19	36.1	93.0	2.6

Рис. 7.3. Результат классификации записи исследуемой выборки

8. Третий прототип программы (приоритет Could)

Третий этап разработки включает реализацию пользовательских форм. На этапе планирования было решено реализовать режимы главного меню, такие как (рис. 8.1):

- ввод данных в таблицу данных классифицируемой выборки;
- классификация выборки, с выводом результата классификации и возможностью добавления достоверных записей в классифицирующую выборку с моментальным обновлением весов классификатора;
- полнофункциональное редактирование таблицы данных обучающей выборки;
- создание пользовательских баз данных с определенным набором параметров и препаратов.

Разработка велась с использованием Python 3 и библиотеки Tkinter для реализации кроссплатформенных приложений (рис. 8.2-8.6).

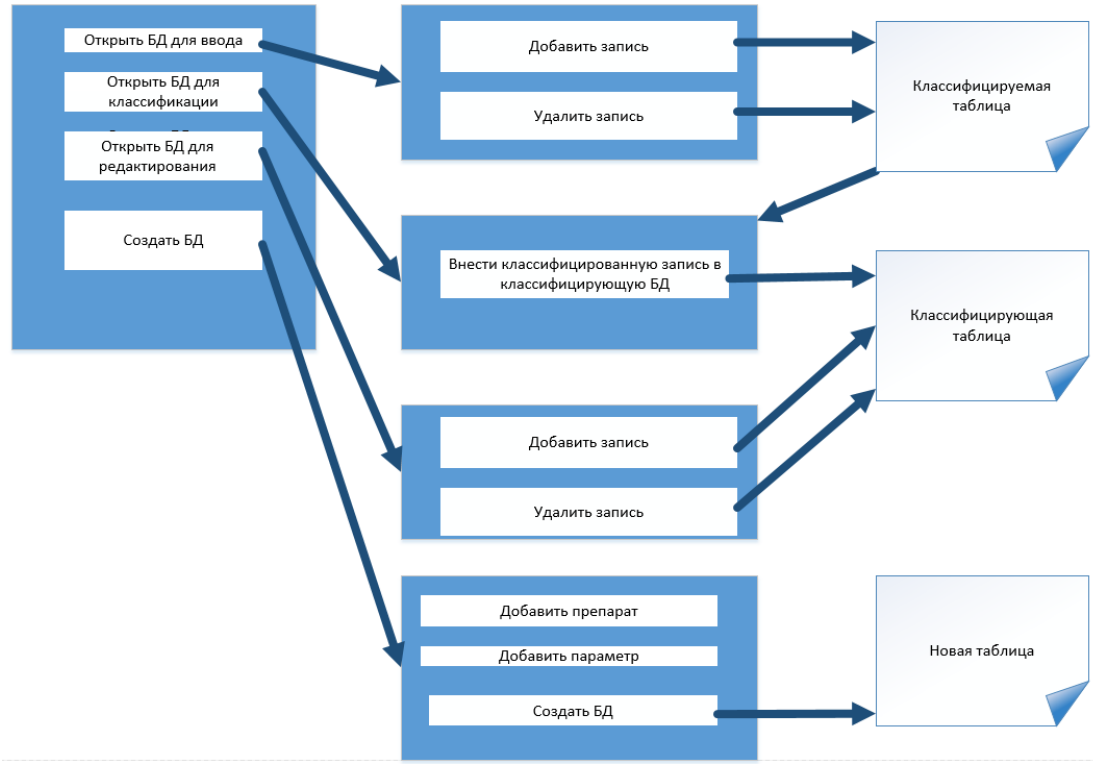


Рис. 8.1. Схема программы

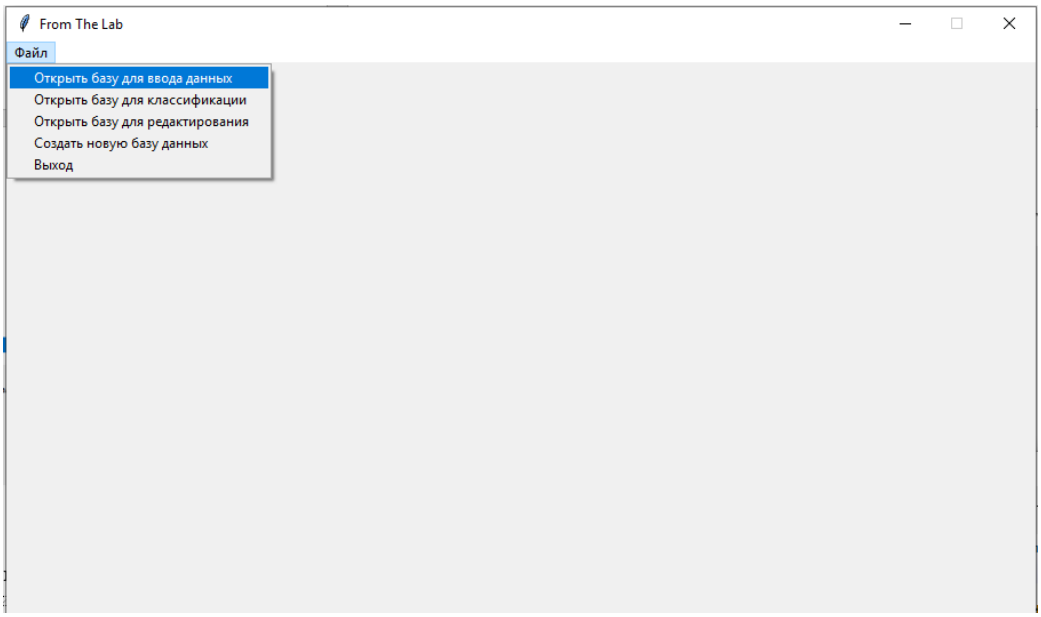


Рис. 8.2. Меню «Файл» для запуска режимов работы с БД

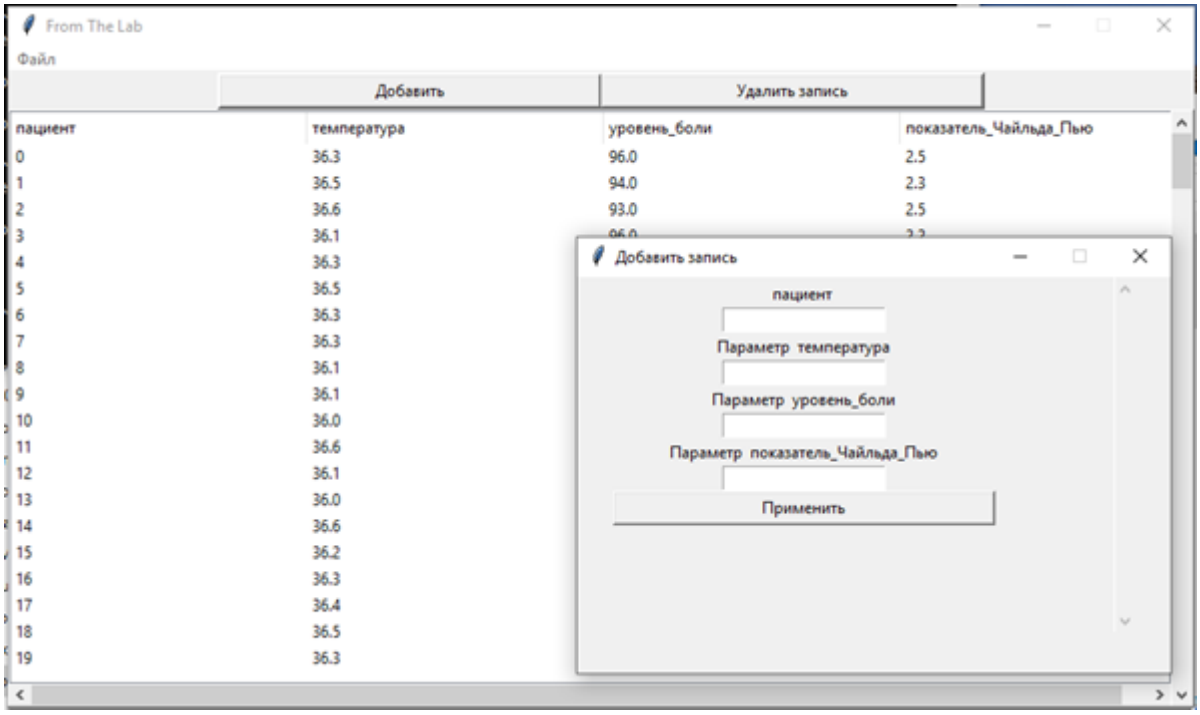


Рис. 8.3. Режим ввода с формой для создания записи с параметрами пациента

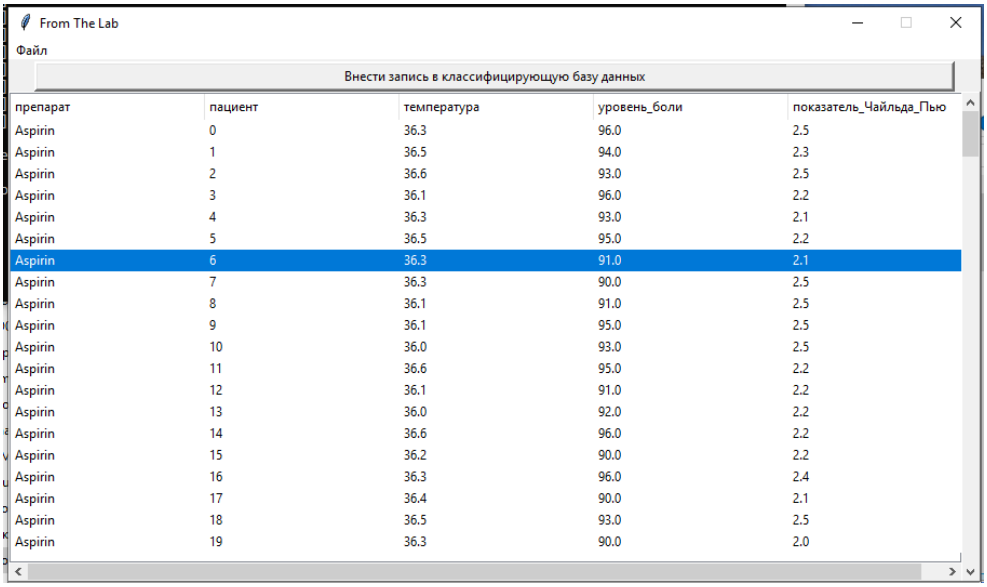
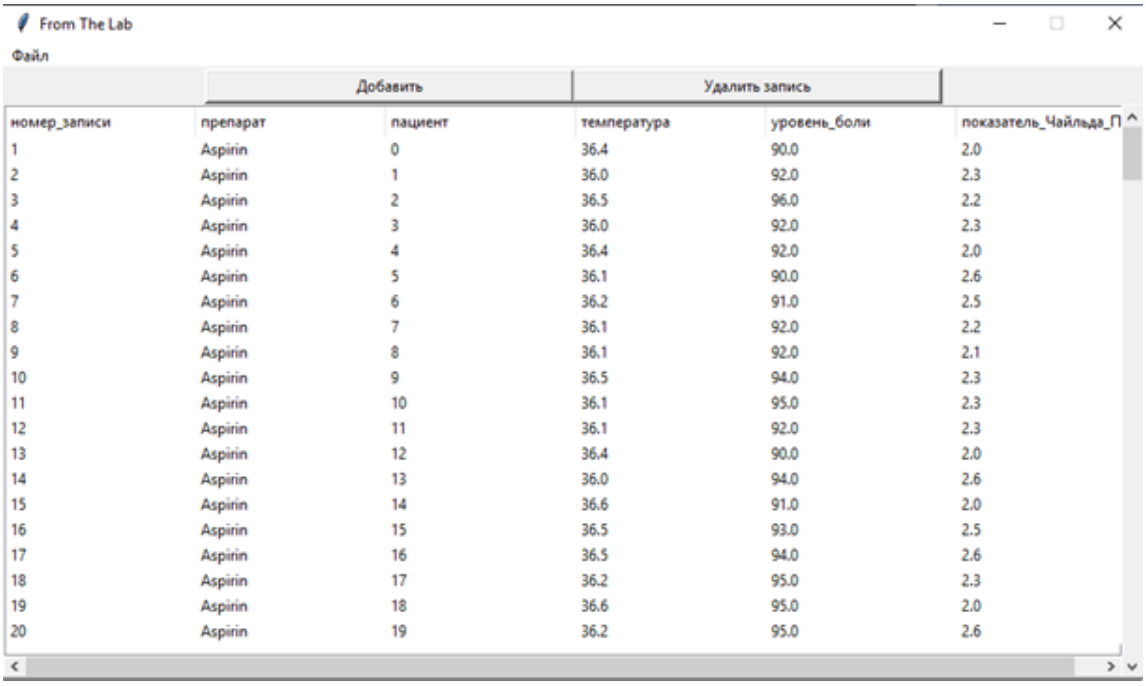


Рис. 8.4. Режим классификации с открытой БД, для которой автоматически вычислены предполагаемые препараты



номер_записи	препарат	пациент	температура	уровень_боли	показатель_Чайльда_П
1	Aspirin	0	36.4	90.0	2.0
2	Aspirin	1	36.0	92.0	2.3
3	Aspirin	2	36.5	96.0	2.2
4	Aspirin	3	36.0	92.0	2.3
5	Aspirin	4	36.4	92.0	2.0
6	Aspirin	5	36.1	90.0	2.6
7	Aspirin	6	36.2	91.0	2.5
8	Aspirin	7	36.1	92.0	2.2
9	Aspirin	8	36.1	92.0	2.1
10	Aspirin	9	36.5	94.0	2.3
11	Aspirin	10	36.1	95.0	2.3
12	Aspirin	11	36.1	92.0	2.3
13	Aspirin	12	36.4	90.0	2.0
14	Aspirin	13	36.0	94.0	2.6
15	Aspirin	14	36.6	91.0	2.0
16	Aspirin	15	36.5	93.0	2.5
17	Aspirin	16	36.5	94.0	2.6
18	Aspirin	17	36.2	95.0	2.3
19	Aspirin	18	36.6	95.0	2.0
20	Aspirin	19	36.2	95.0	2.6

Рис. 8.5. Режим редактирования БД

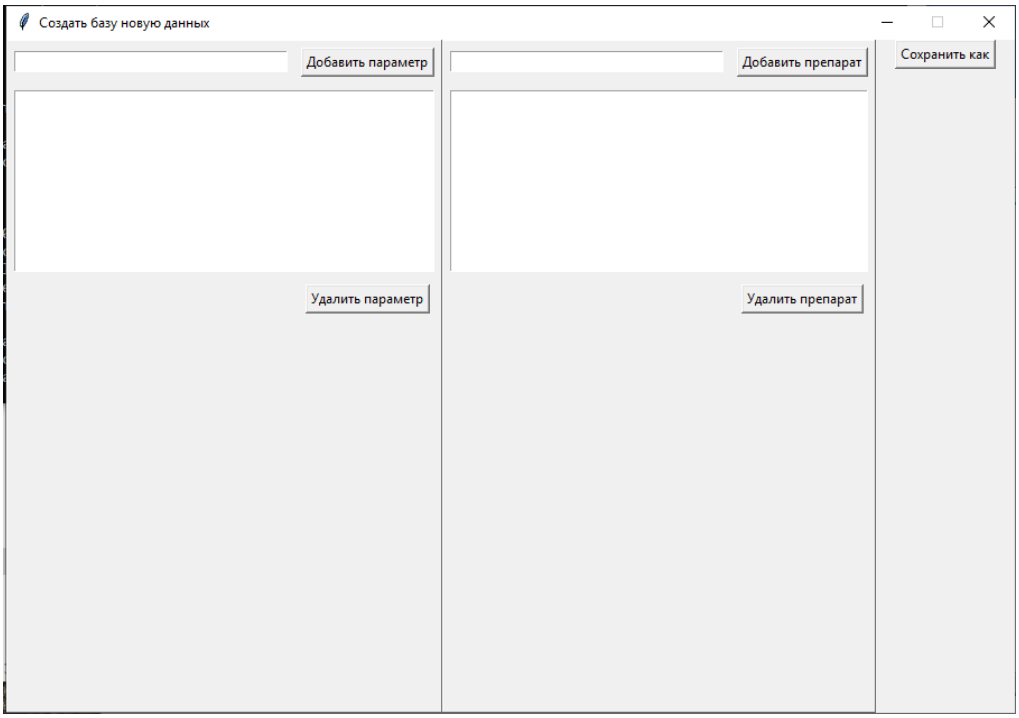


Рис. 8.6. Режим создания новой БД с пользовательскими названиями препаратов и параметров

9. Четвертый прототип программы (приоритет Would)

На четвертой итерации была разработана подсказка в панели состояния программного продукта. Что позволило повысить понятность интерфейса без размещения на начальном экране кнопок открытия файлов (рис. 9.1). На этой итерации было воплощено последнее требование, поэтому цикл разработки программы завершается.

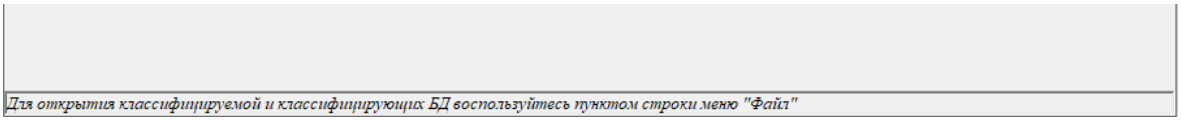


Рис. 9.1. Строка подсказки в панели состояния

Заключение

В работе проделано моделирование процессов в нотации IDEF0 на верхнем уровне, на нижнем уровне моделирование процессов велось с применением нотации IDEF3. Построена карта процессов до внедрения системы и предполагаемая в случае ее разработки. Проведен анализ требований по созданию медицинской информационной системы. По результатам приоритизации был составлен план разработки согласно итерационной модели.

Создан прототип программы, реализующей требования за 4 итерации: проектирование данных, проектирование методов и алгоритмов, проектирование интерфейсов, создание подсказок для пользователя. Проведен тест классификации исследуемой выборки улучшенным методом ближайшего соседа с использованием записей классифицирующей выборки.

Литература

1. Швец М.Ю. Монотонные классификаторы для задач медицинской диагностики. Бакалаврская диссертация / МФТИ.-М., 2015.
2. Федорова Г.Н. Информационные системы: учебник для студ. учреждений сред. проф. образования // Издательский центр «Академия», 2013 – 208 с.
3. Невлюдов И. Ш., Евсеев В. В., Бортникова В. О. Модели жизненного цикла программного обеспечения при разработке корпоративных информационных систем технологической подготовки производства. – 2011.
4. Анисимов В. В. Проектирование информационных систем. Конспект лекций. М.: Дальневосточныйгосударственный университет путей сообщения.

URL:https://sites.google.com/site/anisimovkhv/publication/umr/pris (дата обращения 09.12.2019).

5. ГОСТ Р ИСО/МЭК 12207-2010. Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств.
6. Карпенко С.Н., Вершинина Е.В., Гонченко М.С. Обзор моделей жизненного цикла разработки программного обеспечения// Математические и программные технологии для современных компьютерных систем (Информационные технологии), 2017 – 69 с.
7. Лешек А. Мацяшек Анализ требований и проектирование систем. // Издательский дом «Вильямс», 2005 – 432 с.
8. Свободная энциклопедия Википедия, статья «Анализ требований».URL:https://ru.wikipedia.org/wiki/Анализ_требований (дата обращения: 22.05.2020).
9. Ротер М., Шук Дж. Учитесь видеть бизнес-процессы. Построение карт потоков создания ценности / Ротер М. – М.: Альпина Паблишер, 2017. – 144 с.

Выходные данные статьи

Колосов И.А. Методы распознавания в задачах определения лекарственных препаратов с применением итерационной модели внедрения // Корпоративные информационные системы. – 2021. – №2 (14) – С. 36-58. – URL: https://corpinfosys.ru/archive/issue-14/126-2021-14-recognition.

Об авторе

Колосов Иван Алексеевич – студент 4-го курса кафедры оптических и биотехнических систем и технологий физико-технологического института РТУ МИРЭА. Тема выпускной квалификационной работы бакалавра «Использование методов распознавания в задачах автоматизированного определения лекарственных препаратов на основе данных пациента». Электронная почта: mail@corpinfosys.ru.