|  |  |
| --- | --- |
|  | |
| МИНОБРНАУКИ РОССИИ | |
| Федеральное государственное бюджетное образовательное учреждениевысшего образования"МИРЭА - Российский технологический университет"РТУ МИРЭА | |
| Институт информационных технологий (ИТ) | |
| Кафедра вычислительной техники | |

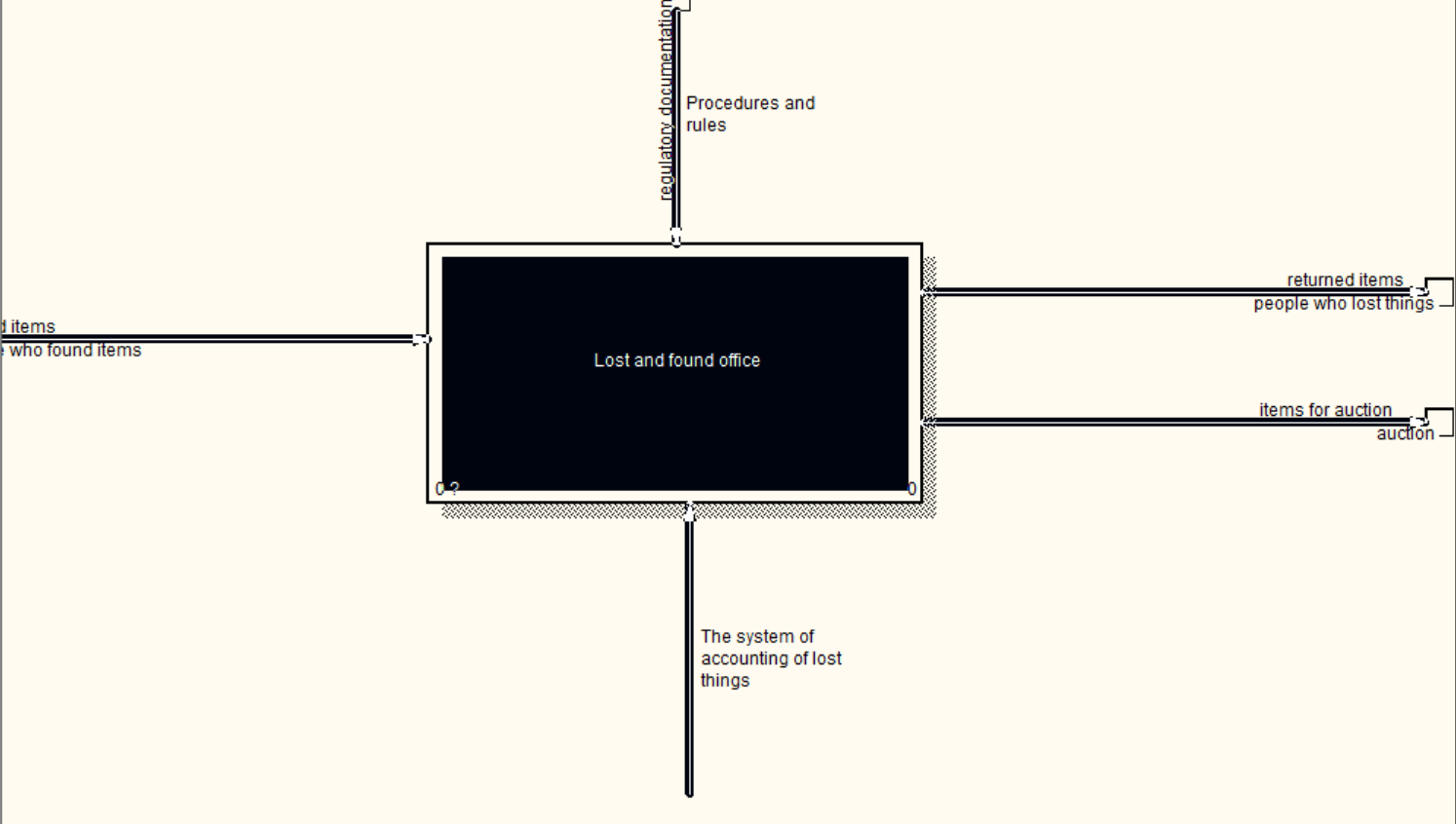
|  |  |
| --- | --- |
| **Отчет по практической работе. Тема:**  **«Модели IDEF0 и DFD»** | |
| **по дисциплине** | |
| **«Системы обработки и хранения данных»** | |
|  | |
| Выполнила студентка группы ИКБО-04-19 |  |
| Принял |  |

|  |  |  |
| --- | --- | --- |
| Практические работы выполнены | «14» марта 2021 г. |  |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_2020\_ г. |  |

**IDEF0**

Модель IDEF0 по предметной области “Бюро находок”.

**Предметная область “Бюро находок” –** бюро занимается нахождением, регистрацией, хранением и возвратом утерянных вещей.



IDEF0. Контекстная диаграмма

Основной блок – Деятельность бюро находок

Входными данными является:

**Найденная вещь**

Выходными данными является:

**Возвращенная владельцу вещь**

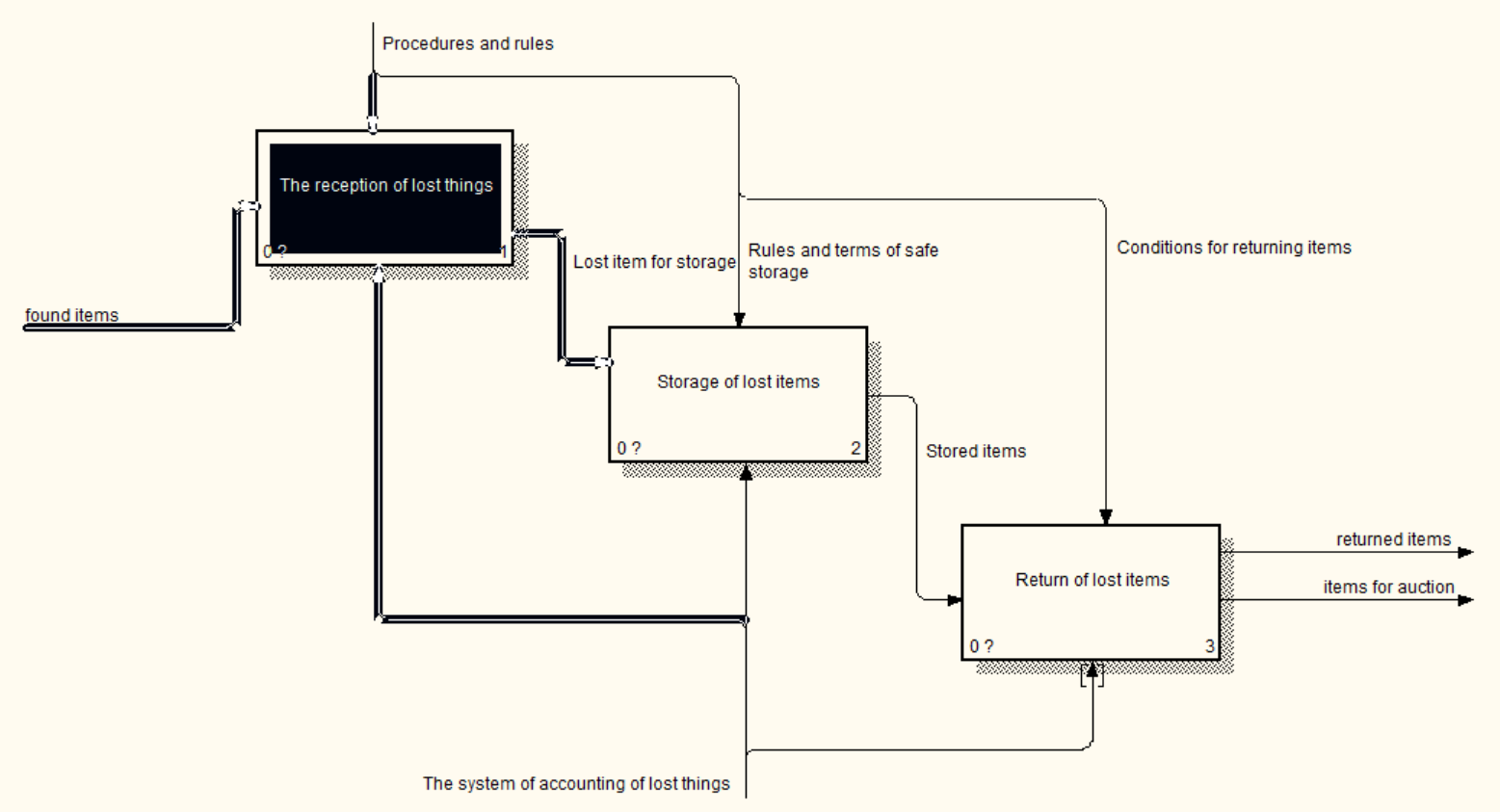
**Вещь, переданная на аукцион**

Механизмом системы является:

**Система контроля утерянных вещей**

Управляющим блоком является:

**Законы и нормы работы бюро находок**

****

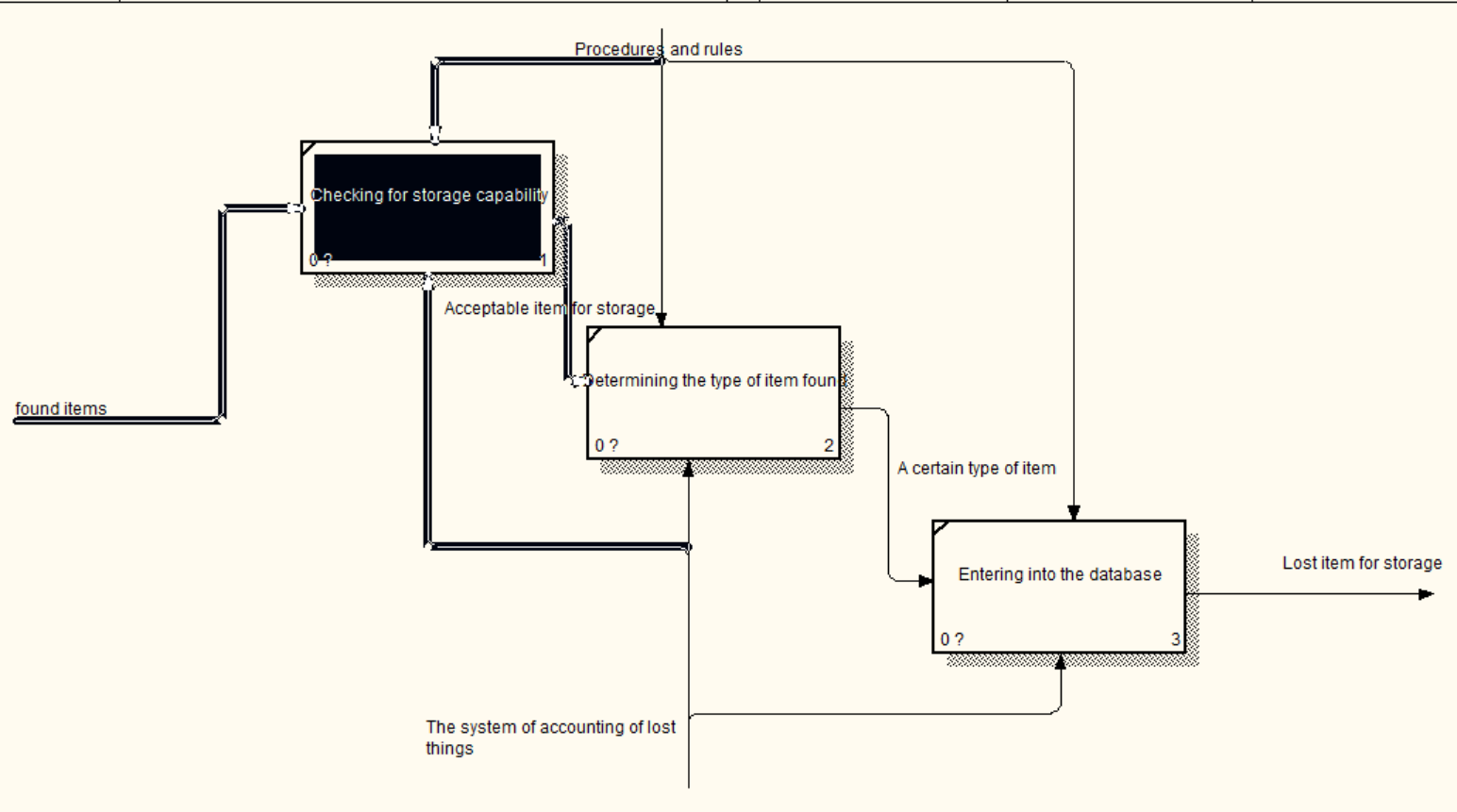
Декомпозиция контекстной диаграммы

Мы декомпозировали общий блок «Деятельность бюро находок» на связанные между собой элементы. В нашем случае делится на 3 основных этапа:

**Регистрация найденной вещи**

**Хранение найденной вещи**

**Возврат найденной вещи**

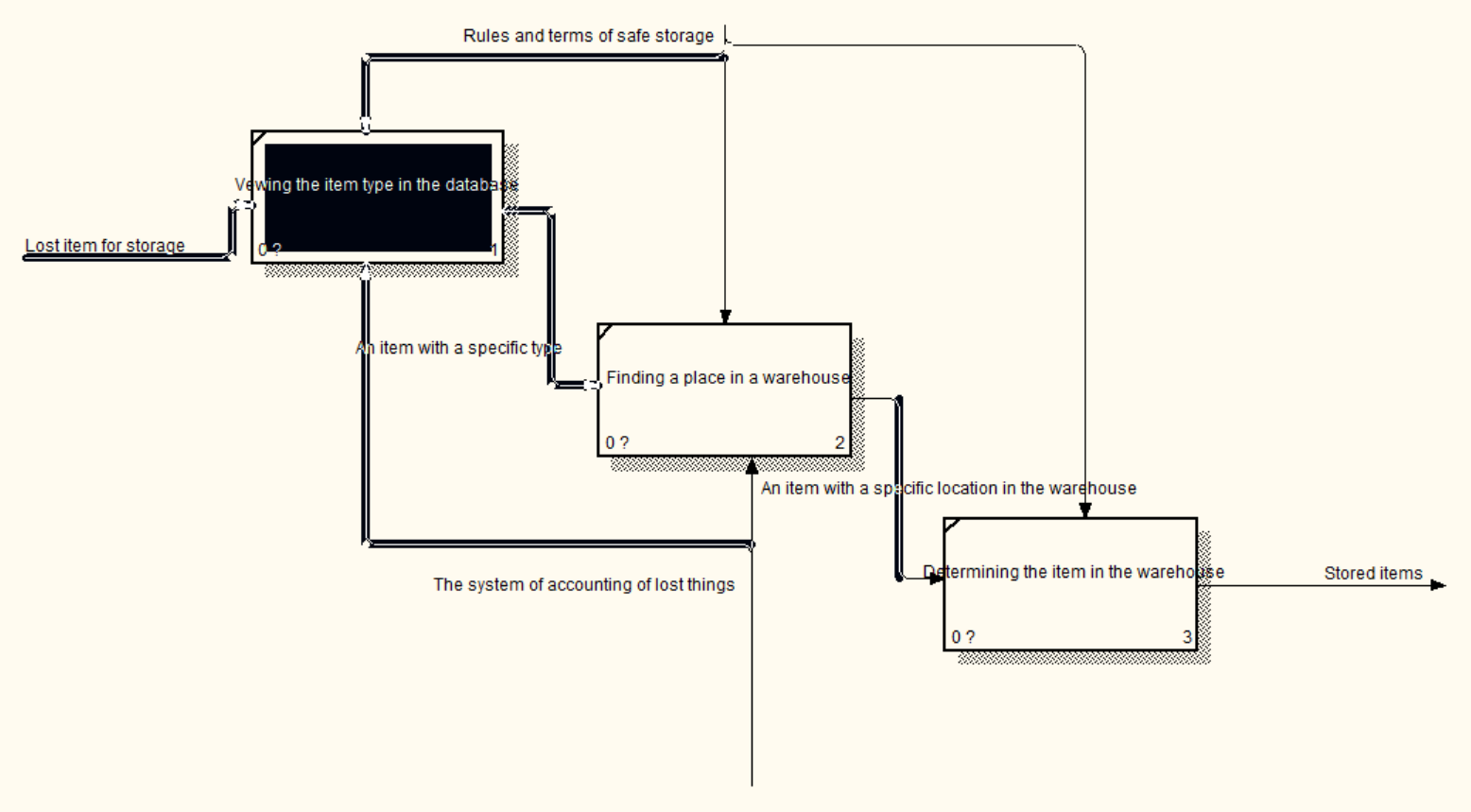
****

Блок “Регистрация потерянной вещи” мы декомпозируем еще на 3 этапа:

**Проверка на пригодность к хранению**

**Определение типа найденной вещи**

**Занесение в базу данных**

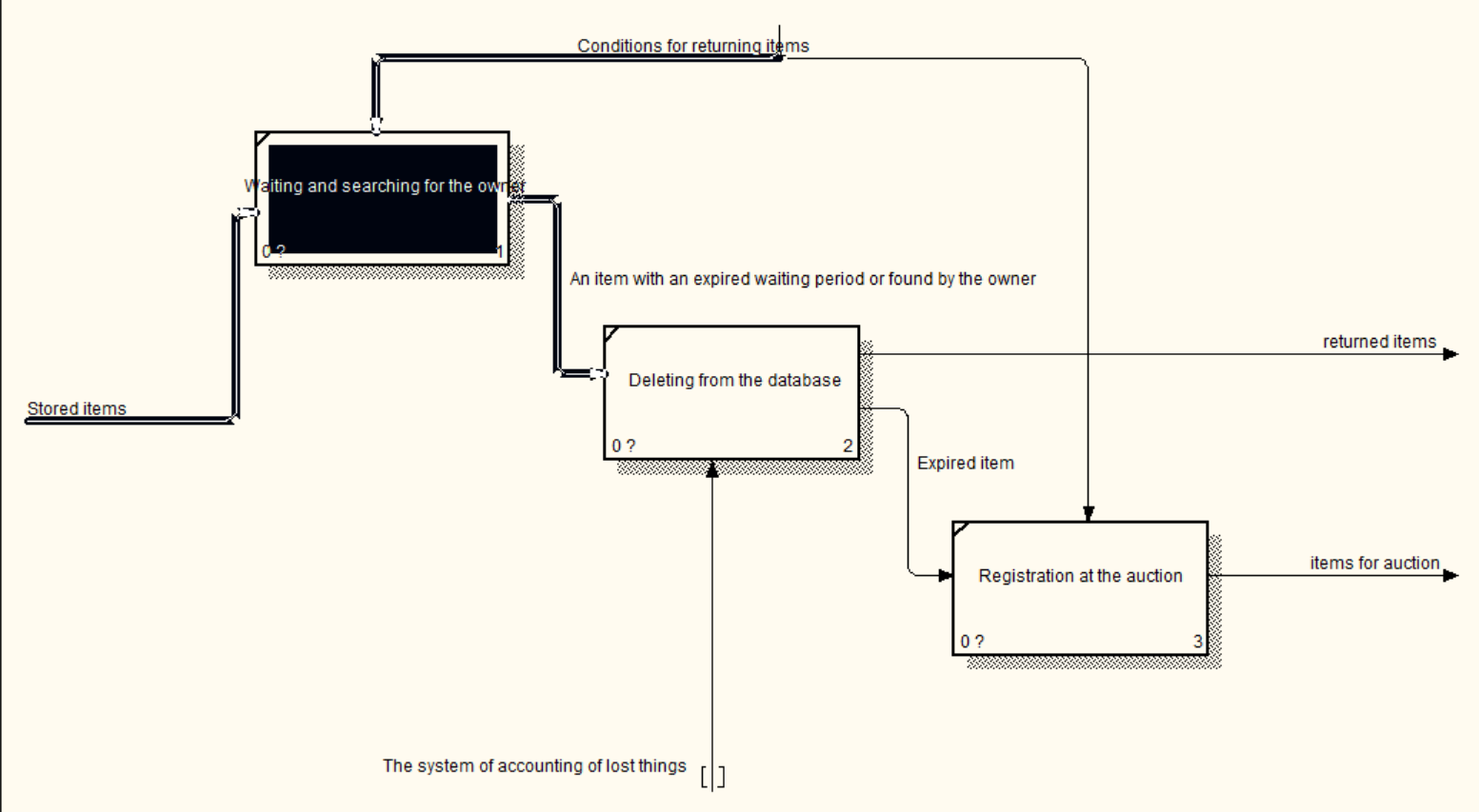


Блок “Хранение вещей” мы декомпозируем еще на 3 этапа:

**Просмотр типа найденной вещи в базе данных**

**Нахождение места на складе**

**Расположение найденной вещи на складе**

****

Блок “Возврат утерянных вещей” мы декомпозируем еще на 3 этапа:

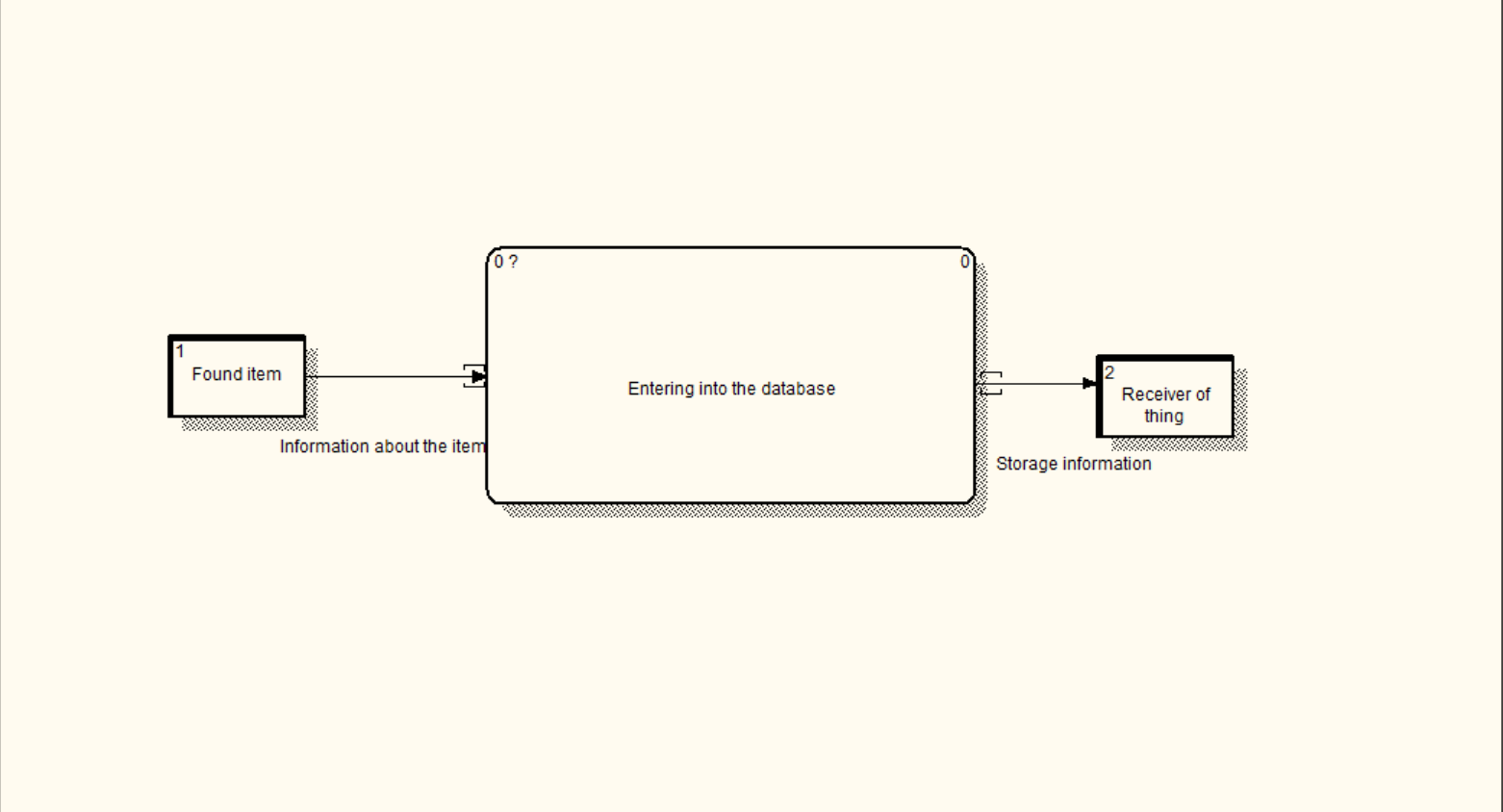
**Поиск владельца**

**Удаление вещи из базы данных**

**Выставление на аукцион**

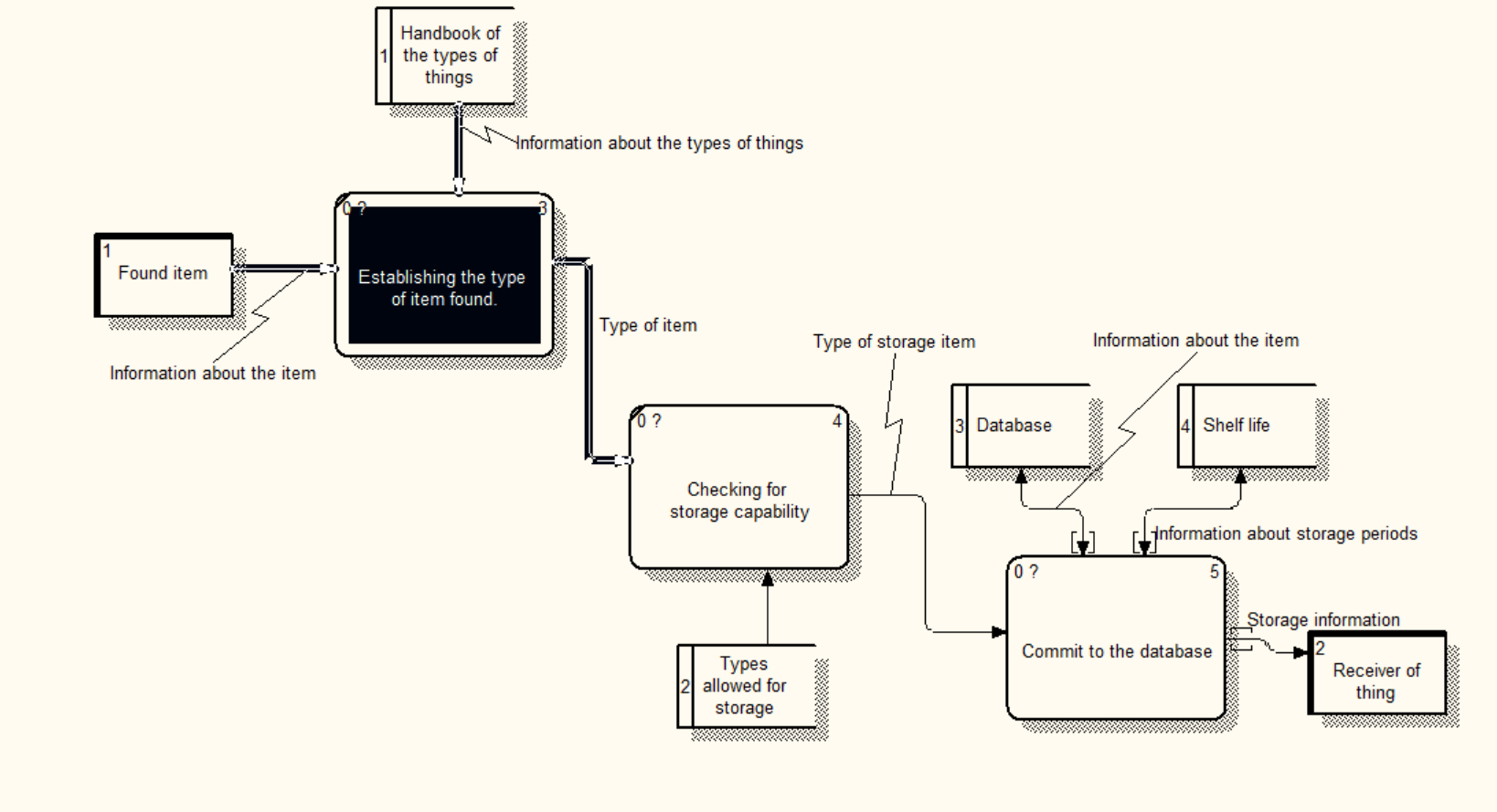
**DFD**

Модель DFD по предметной области “Бюро находок”



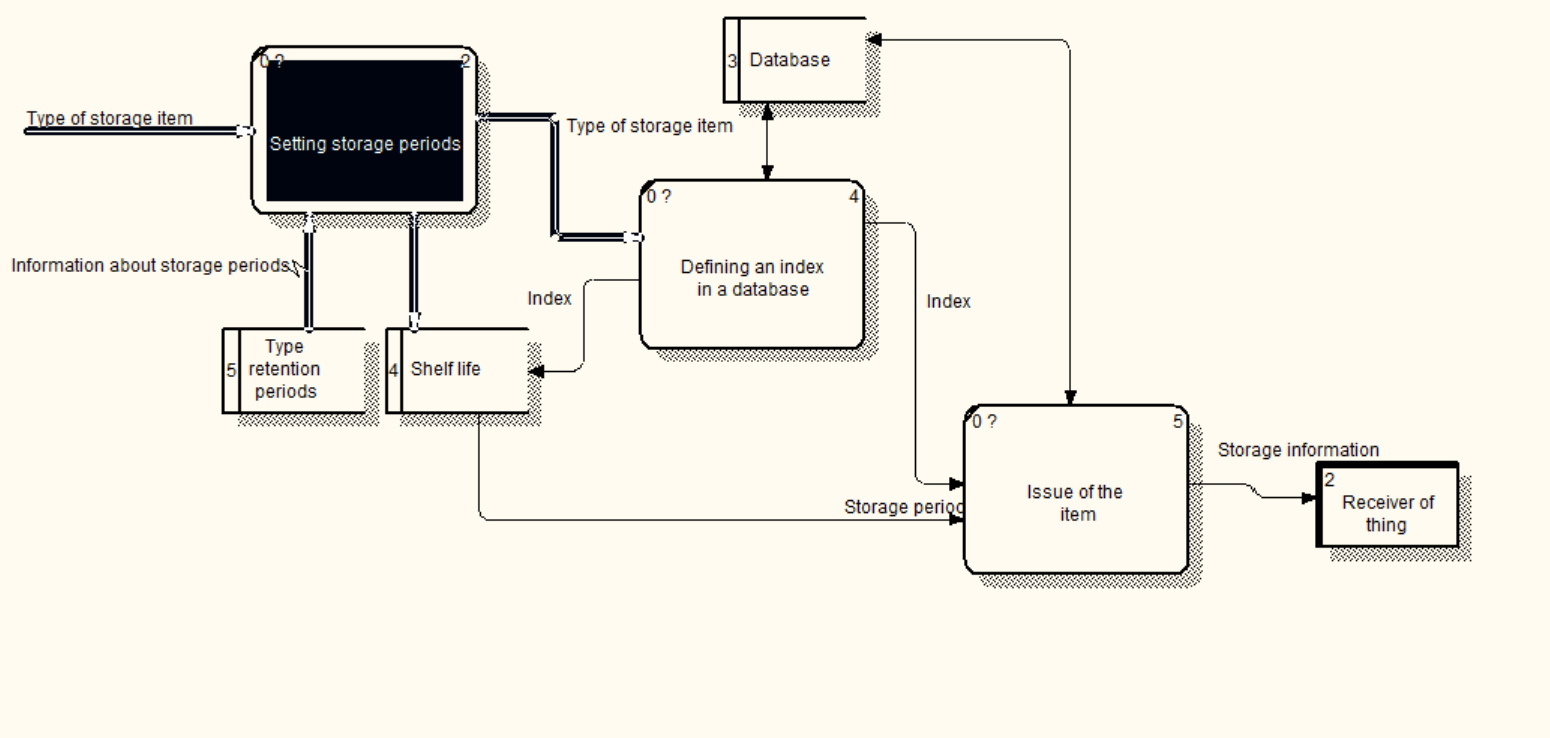
Контекстная диаграмма DFD

Модель показывает информационные потоки в системе, их взаимодействие и соотношение. Основным процессом системы является **“Взаимодействие с базой данных”**, который преобразует информационный поток **“Информация о вещи”** от сущности **“Найденная вещь”** в информационный поток **“Информация о хранении”** для сущности **“Приемник вещи”**



Декомпозируем контекстную диаграммы на еще три процесса:

1. **Установление типа найденной вещи.** Принимает информационный поток **“Информация о вещи”** от сущности **“Найденная вещь”** и информационный поток **“Информация и типах вещей”** от хранилища **“Справочник о типах вещей”** и преобразует их в информационный поток **“Тип вещи”** для передачи в процесс **“Проверка на возможность хранения”**
2. **Проверка на возможность хранения.** Принимает информационный поток от процесса **“ Установление типа найденной вещи”** и информационный поток от хранилища **“Список типов, допустимых для хранения”** и преобразует их в информационный поток **“Тип вещи для хранения”** для процесса **“Занесение в базу данных”**
3. **Занесение в базу данных.** Процесс производит обмен информационными потоками с такими хранилищами как **“База данных”** и **“Сроки хранения”.** Происходит передача информационного потока **“Информация о хранении”** в сущность **“Приемник вещи”**

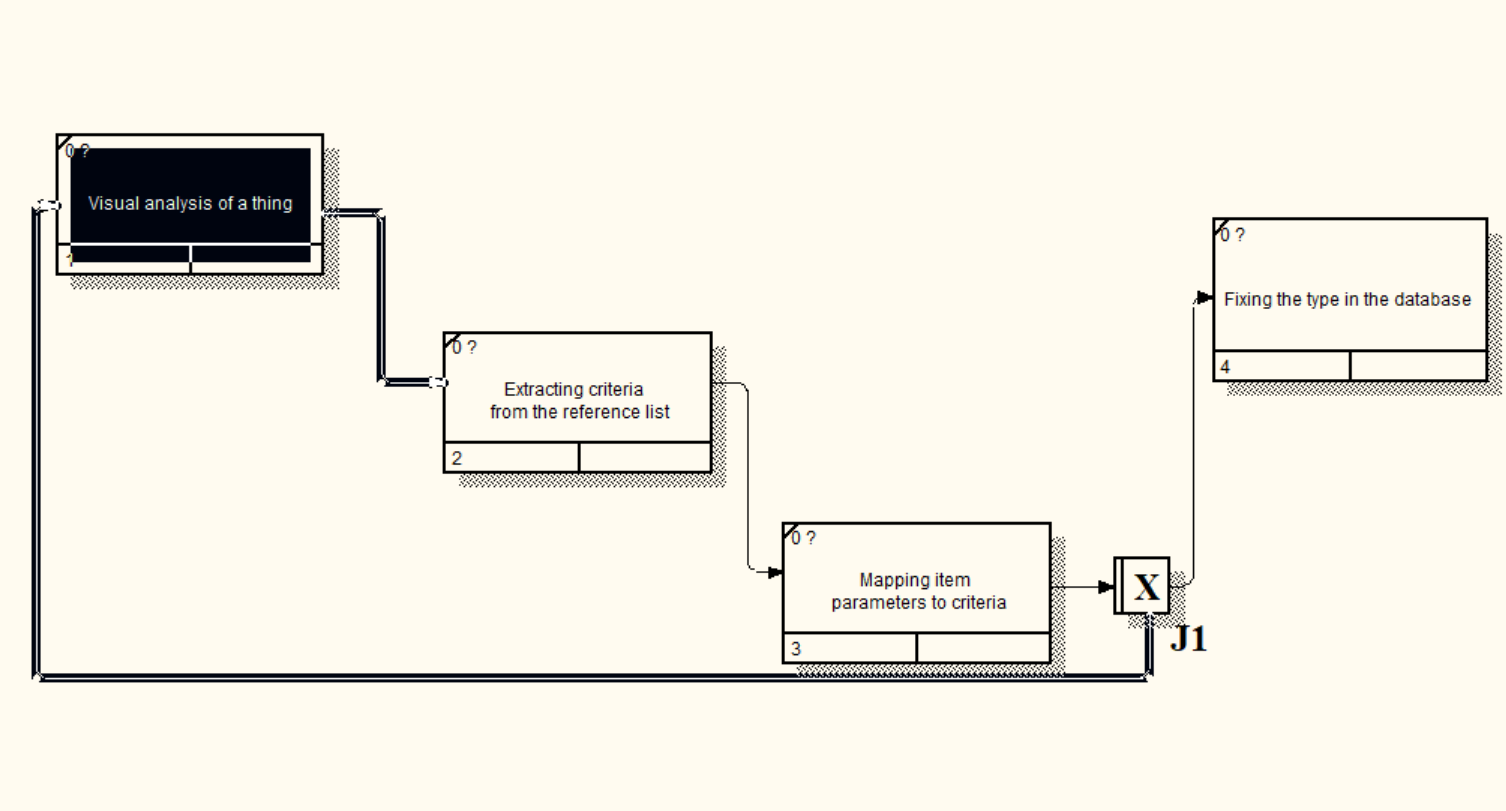


Декомпозируем процесс **“Занесение в базу данных”** еще на три блока:

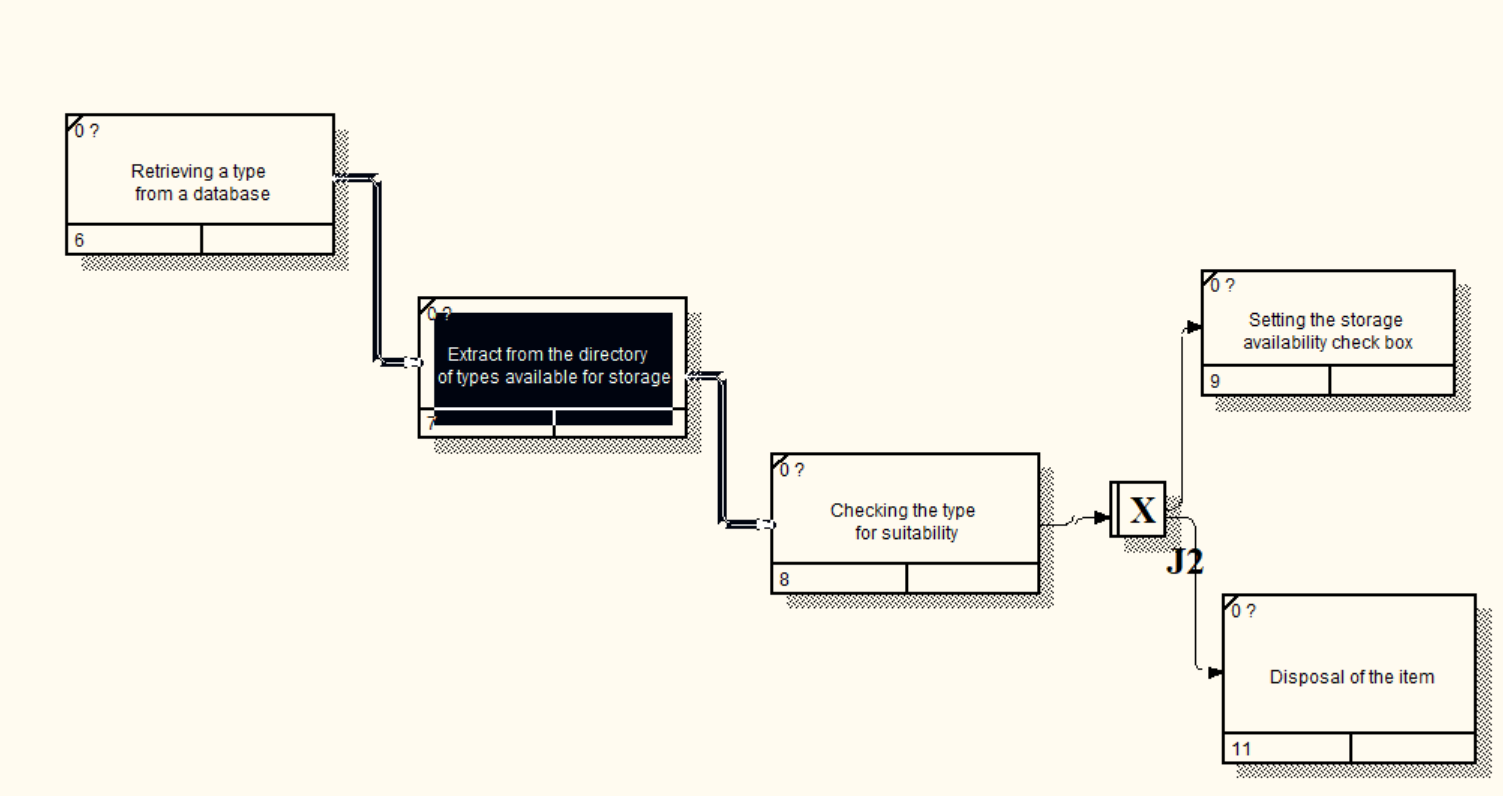
1. **Установка периодов хранения.** Принимает информационный поток **“Тип вещи для хранения”** от внешней сущности и информационный поток **“Информация о периодах хранения”** от хранилища **“Сроки хранения для различных типов”**. Информационный поток **“Информация о периодах хранения”** передается в хранилище **“Сроки хранения”**. Информационный поток **“Тип вещи для хранения”** передается в процесс **“Определение индекса в базе данных”**
2. **Определение индекса в базе данных.** Процесс обменивается информационными потоками с хранилищем **“База данных”**, а также передает информационный поток **“Индекс”** в хранилище **“Сроки хранения”** и процессу **“Выдача вещи”.**
3. **Выдача вещи.** Процесс обменивается информационными потока с хранилищем **“База данных”** и получается информационный поток **“Период хранения”** от хранилища **“Сроки хранения”**. Передает информационный поток **“Информация о хранении”** сущности **“Приемник вещи”**

**IDEF3**

*Модель IDEF3 по предметной области « Бюро находок».*

**

Методология IDEF3 позволяет декомпозировать работу многократно, т. е. работа может иметь множество дочерних работ. Возможность множественной декомпозиции отражается в нумерации работ: номер работы состоит из номера родительской работы, номера декомпозиции и номера работы на текущей диаграмме.

**

**Диаграмма прецендентов**

Use case diagram (диаграммы прецедентов) - этот вид диаграмм

позволяет создать список операций, которые выполняет система. Каждая

такая диаграмма – это описание сценария поведения, которому следуют

действующие лица (Actors).

Данный тип диаграмм используется при описании бизнес процессов

предметной области, определении требований к будущей программной

системе. Отражает объекты как системы, так и предметной области и задачи,

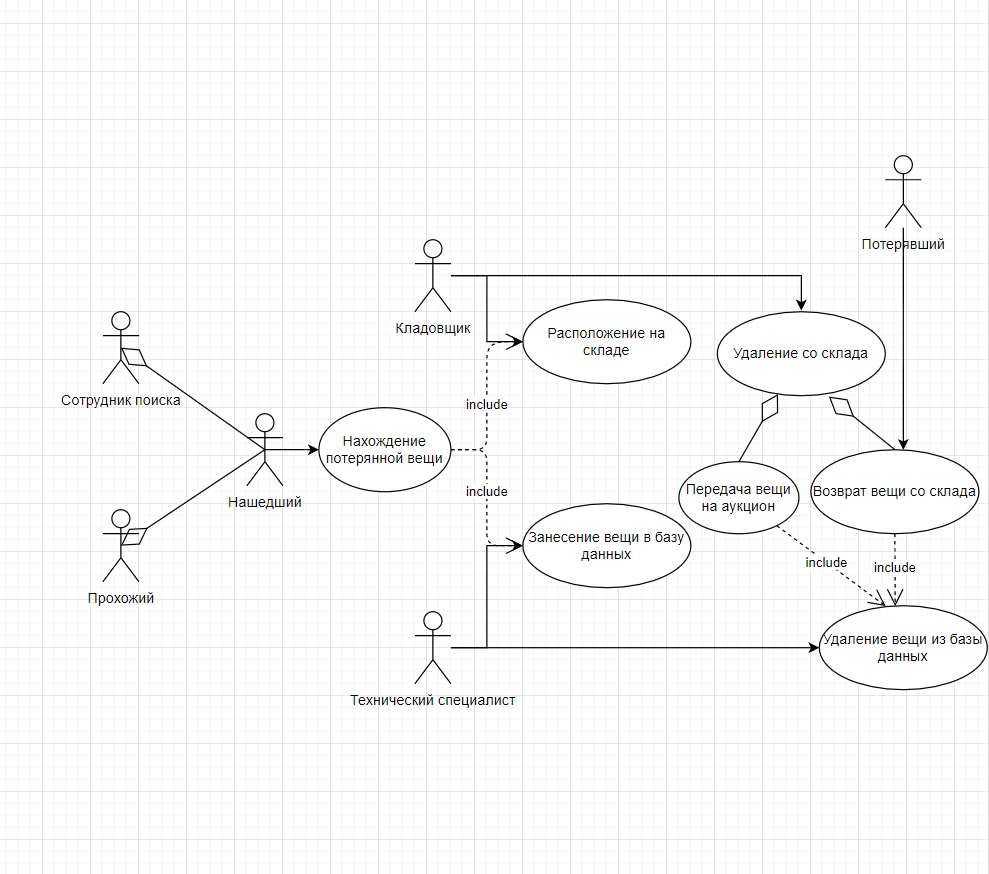
ими выполняемые. Окончательный вид диаграммы показан на рисунке 13 На

диаграмме находятся Use Case (Вариант использования), Actor

(Действующее лицо). Так же были добавлены ассоциации и описания ко

всем элементам диаграммы.

На данной диаграмме представлен процесс работы бюро находок. Сотрудник поиска или случайный прохожий может принести найденную вещь в бюро. Кладовщик расположит вещь на складе, а технический специалист занесет все данные в базу данных. Если потерявший придет забирать вещь, или если по истечению определенного срока вещь будет передана на аукцион, то технический специалист проследит, чтобы информация о вещи была удалена из базы данных.



**Диаграмма последовательности**

Sequence diagram (диаграммы последовательностей действий). Данный

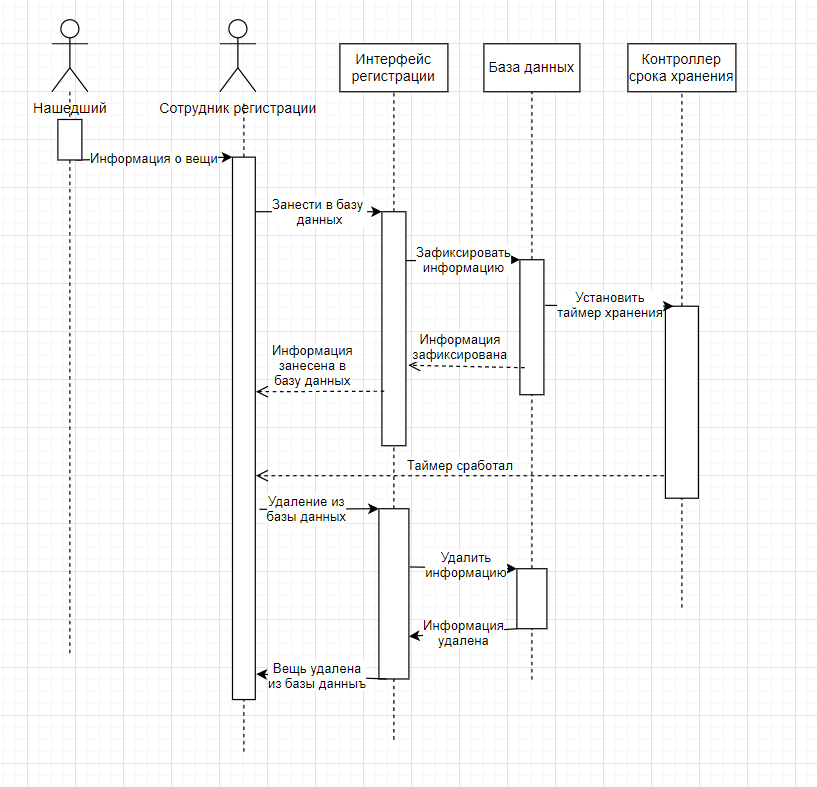
тип диаграмм позволяет отразить последовательность передачи сообщений

между объектами. Этот тип диаграммы не акцентирует внимание на

конкретном взаимодействии, главный акцент уделяется последовательности

приема/передачи сообщений.

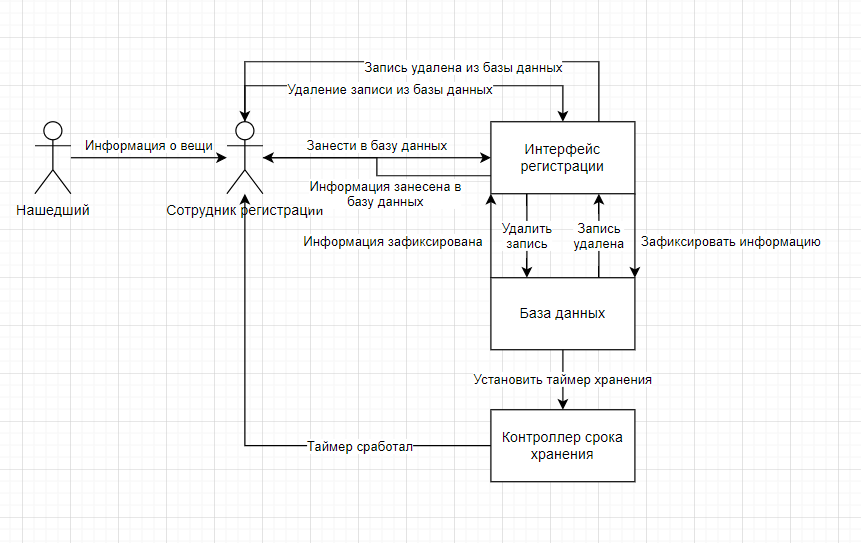
На диаграмме показан процесс занесения информации о найденной вещи в базу данных. Нашедший передает ее сотруднику регистрации, который через соответствующий интерфейс взаимодействует с базой данных и с контроллером таймера хранения. По срабатыванию таймера происходит очистка базы данных от информации об объекте, так как он передается на аукцион



**Кооперативная диаграмма**

Collaboration diagram (диаграммы сотрудничества). Этот тип диаграмм позволяет описать взаимодействия последовательности передачи сообщений. На этом типе диаграмм в компактном виде отражаются все принимаемые и передаваемые сообщения конкретного объекта и типы этих сообщений. По причине того, что диаграммы Sequence и Collaboration являются разными взглядами на одни и те же процессы, Rational Rose позволяет создавать из Sequence диаграммы диаграмму Collaboration и наоборот, а также производит автоматическую синхронизацию этих диаграмм.

На диаграмме показан процесс занесения информации о найденной вещи в базу данных. Нашедший передает ее сотруднику регистрации, который через соответствующий интерфейс взаимодействует с базой данных и с контроллером таймера хранения. По срабатыванию таймера происходит очистка базы данных от информации об объекте, так как он передается на аукцион



**Диаграмма состояний**

Каждый объект системы, обладающий определенным поведением,

может находится в определенных состояниях, переходить из состояния в

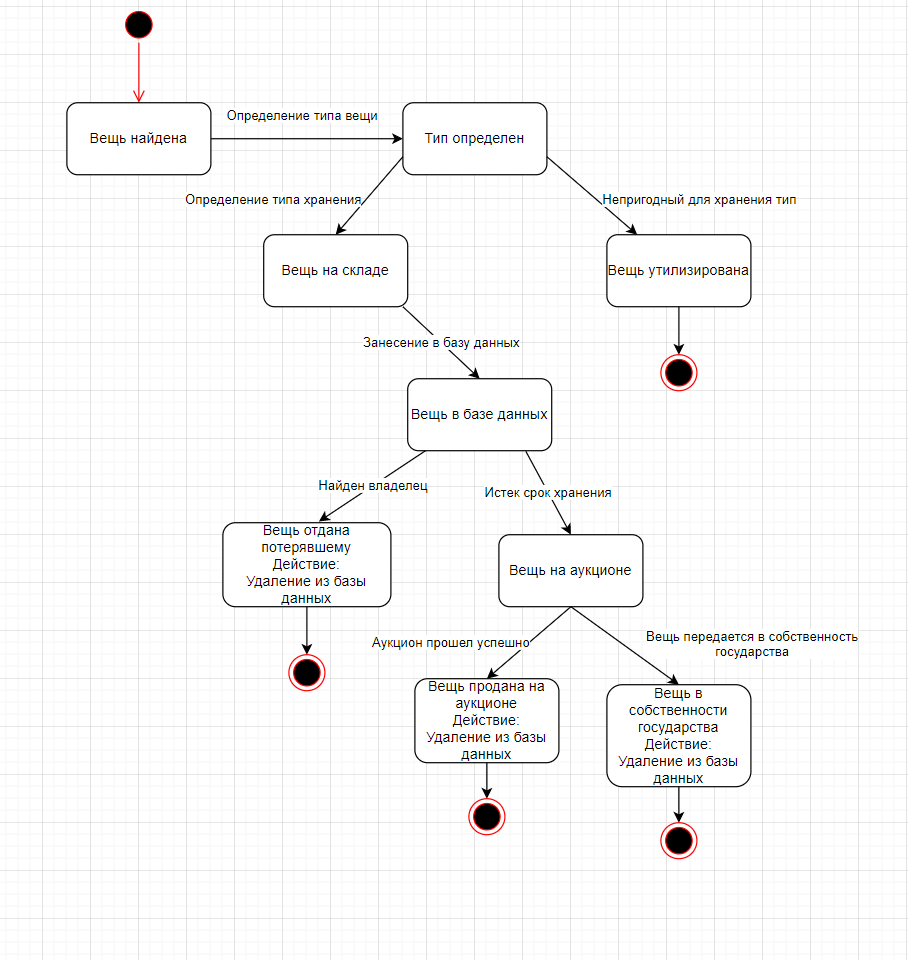
состояние, совершая определенные действия в процессе реализации сценария

поведения объекта. Поведение большинства объектов реальных систем

можно представить с точки зрения теории конечных автоматов, то есть

поведение объекта отражается в его состояниях, и данный тип диаграмм

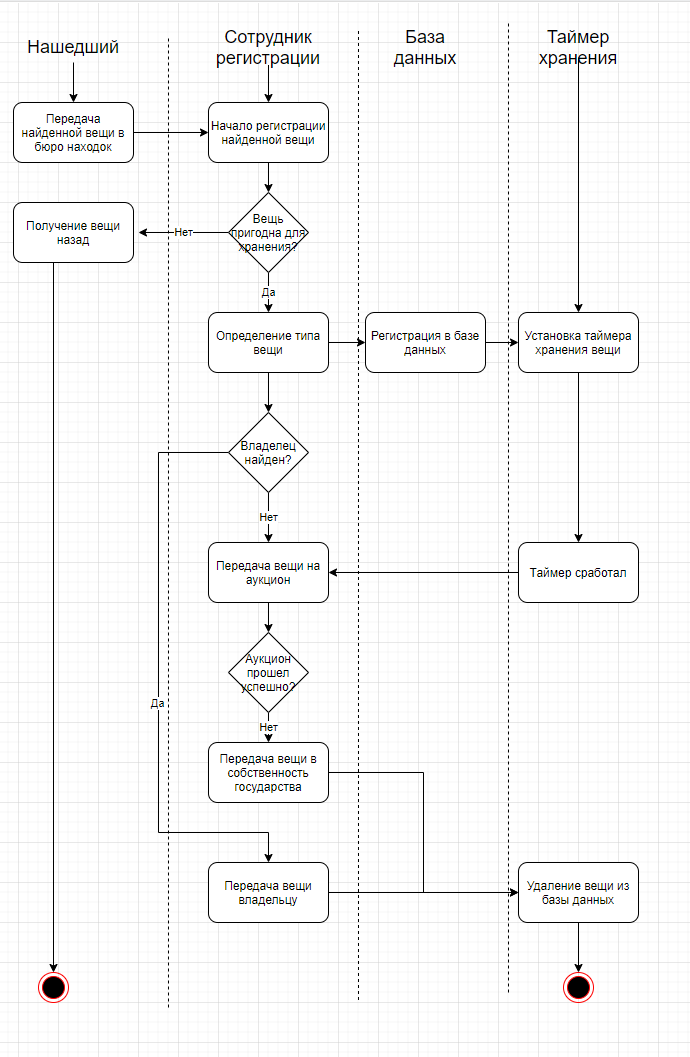
позволяет отразить это графически.

****

**Диаграмма активности**

Activity diagram (диаграммы активности). Это дальнейшее развитие диаграммы состояний. Фактически данный тип диаграмм может использоваться и для отражения состояний моделируемого объекта, однако, основное назначение Activity diagram в том, чтобы отражать бизнес-процессы объекта. Этот тип диаграмм позволяет показать не только последовательность процессов, но и ветвление и даже синхронизацию

процессов

****

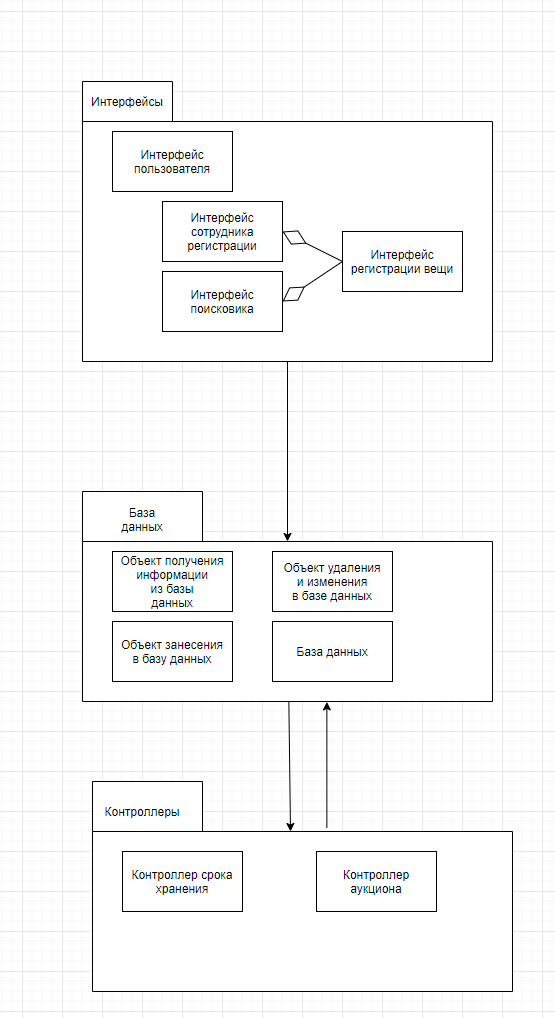
**Пакет и классы**

Class diagram (диаграммы классов). Этот тип диаграмм позволяет

создавать логическое представление системы, на основе которого создается

исходный код описанных классов

Значки диаграммы позволяют отображать сложную иерархию систем, взаимосвязи классов (Classes) и интерфейсов (Interfaces). Данный тип диаграмм противоположен по содержанию диаграмме Collaboration, на котором отображаются объекты системы. В данной работе были созданы пакеты, и объединены в них классы.

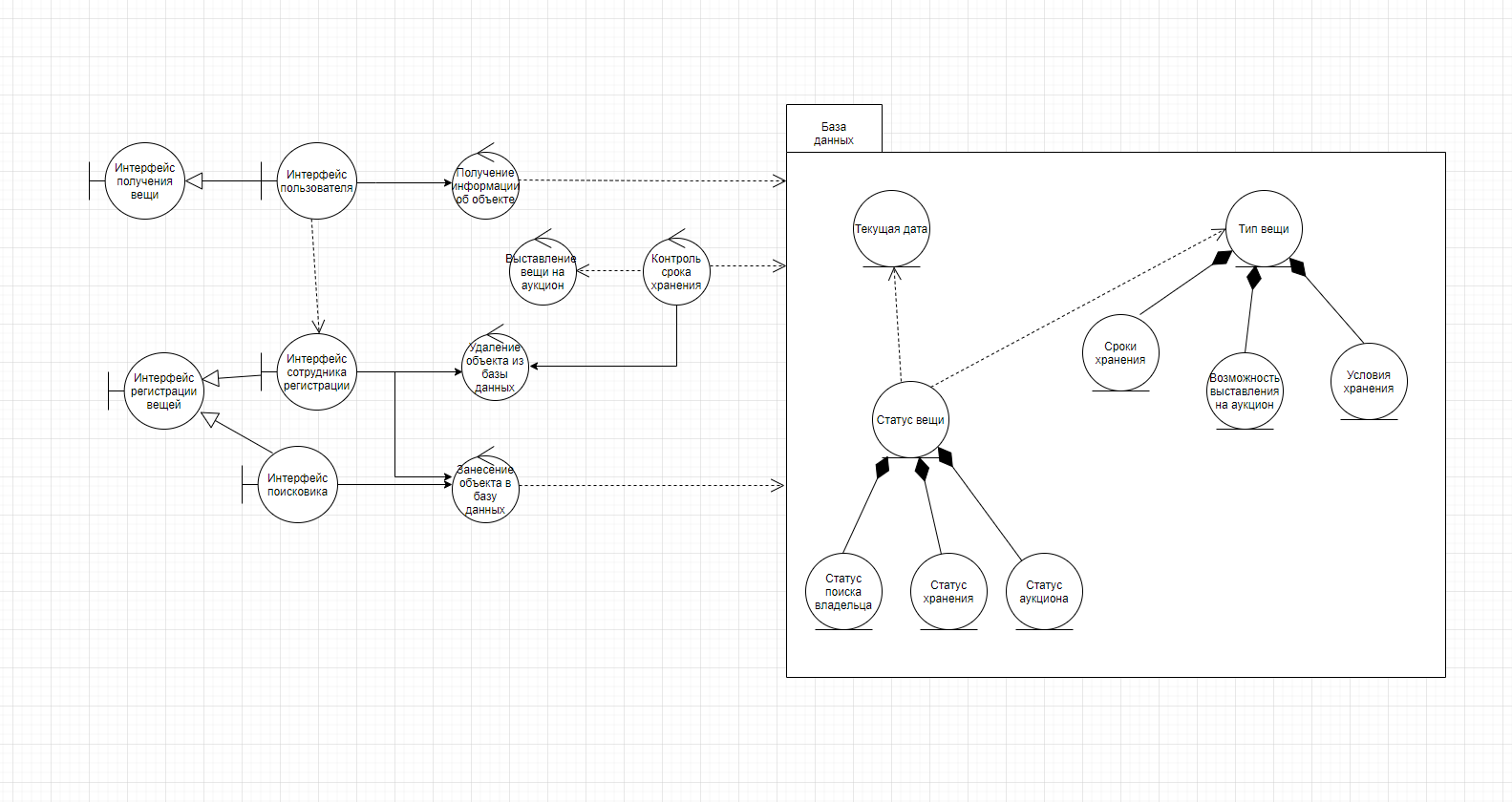
****

**Диаграмма классов**

На данном этапе создается уже готовая диаграмма классов, а именно

добавляются новые атрибуты и операции к классу. Происходит подробное

описание операций и атрибутов, а именно указываются их типы.



**Диаграмма компонентов**

Component diagram (диаграммы компонентов). Этот тип диаграмм

предназначен для распределения классов и объектов по компонентам при

физическом проектировании системы. Часто данный тип диаграмм называют

диаграммами модулей. Диаграмма компонентов представлена на рисунке

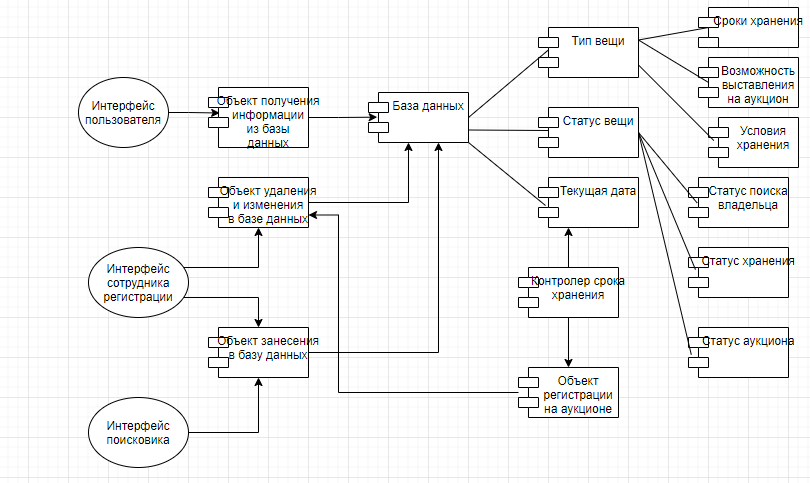
Диаграмма компонентов, в отличие от ранее рассмотренных диаграмм,

описывает особенности физического представления системы. Пунктирные

стрелки, соединяющие модули, показывают отношения взаимозависимости.

Основными графическими элементами диаграммы компонентов являются

компоненты, интерфейсы и зависимости между ними.

****

**Диаграмма развертывания**

Диаграмма развёртывания в UML моделирует физическое развертывание артефактов на узлах. Например, чтобы описать веб-сайт, диаграмма развертывания должна показывать, какие аппаратные компоненты («узлы») существуют (например, веб-сервер, сервер базы данных, сервер приложения), какие программные компоненты («артефакты») работают на каждом узле (например, веб-приложение, база данных), и как различные части этого комплекса соединяются друг с другом (например, JDBC, REST, RMI).

Узлы представляются как прямоугольные параллелепипеды с артефактами, расположенными в них, изображёнными в виде прямоугольников. Узлы могут иметь подузлы, которые представляются как вложенные прямоугольные параллелепипеды. Один узел диаграммы развертывания может концептуально представлять множество физических узлов, таких как кластер серверов баз данных.

