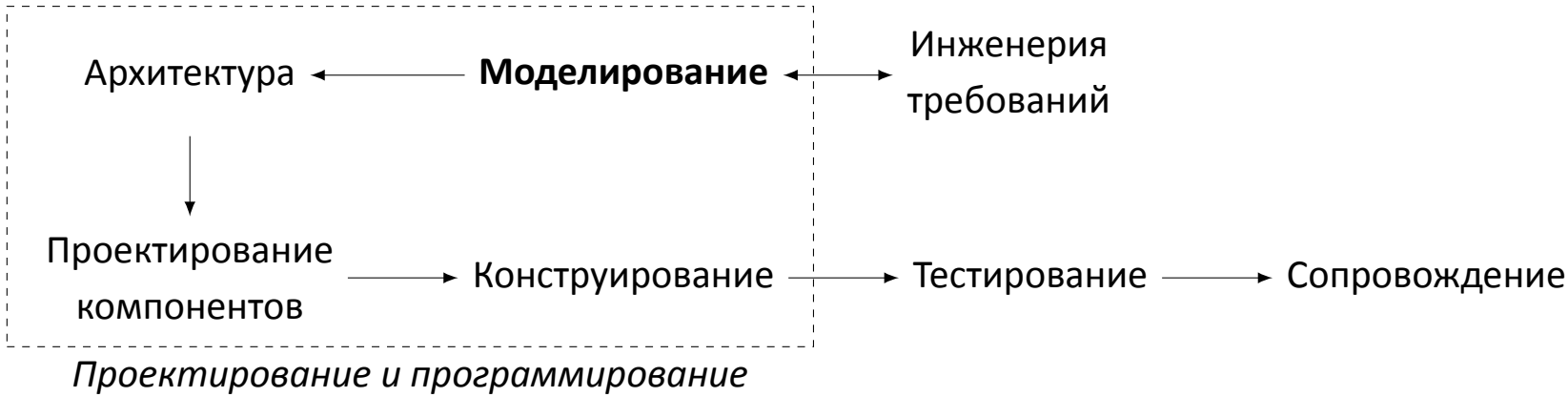
Моделирование программных систем

Алексей Островский

Физико-технический учебно-научный центр НАН Украины

07 ноября 2014 г.

Моделирование в разработке ПО



Определение

Системное моделирование — процесс создания абстрактных моделей программной системы, отображающих различные ее аспекты.

Цели моделирования

Процесс	Цель моделирования
Инженерия требований	объяснение предложенных требований заинтересованным сторонам.
Проектирование	создание общей структуры системы архитекторами; планирование и документирование общих указаний по имплементации.
Программирование	частичная или полная имплементация системы с помощью генераторов кода.
Сопровождение	объяснение структуры системы для команды сопровождения; базис для внесения изменений в систему.

Представления системы

Определение

Представление — абстрактная модель системы, выделяющая ее характеристики, соответствующие определенному аспекту ее функционирования.

Основные представления:

- контекстное представление — модель окружения, в котором выполняется система;
- взаимодействия — связи системы с окружением, а также элементов в системе;
- структурное представление — организация системы и данных для обработки;
- поведение — модель реагирования системы в ответ на внешние события.

Язык моделирования UML

Определение

Унифицированный язык моделирования (англ. *unified modeling language, UML*) — язык маркировки общего назначения, целью которого является стандартизация графического представления архитектуры и дизайна ПС.

1996 г. — UML 1.0 (Grady Booch, Ivar Jacobson, James Rumbaugh).

2000 г. — стандарт ISO.

2005 г. — UML 2.0 (новые виды диаграмм, расширение семантики языка).

Инструменты UML:

- Eclipse Modeling Tools;
- Papyrus;
- Rational Software Architect/Modeler, …

Основные диаграммы UML

- **Диаграмма деятельности** (англ. *activity diagram*): составляющие деятельности по обработке данных.
- **Диаграмма вариантов использования** (англ. *use case diagram*): взаимодействие между системой и ее окружением.
- **Диаграмма последовательности** (англ. *sequence diagram*): взаимодействие системы с актерами и компонентов системы друг с другом.
- **Диаграмма классов** (англ. *class diagram*): структура классов, используемых в системе, и отношения между классами.
- **Диаграмма состояний** (англ. *state diagram*): реагирование системы на внутренние и внешние события.

Контекстные модели

Цель:

- разграничение функций системы и ее окружения;
- определение компонентов, которые надо имплементировать, и используемых интерфейсов.

Пример: контекст электронной библиотечной системы.

Модель процесса

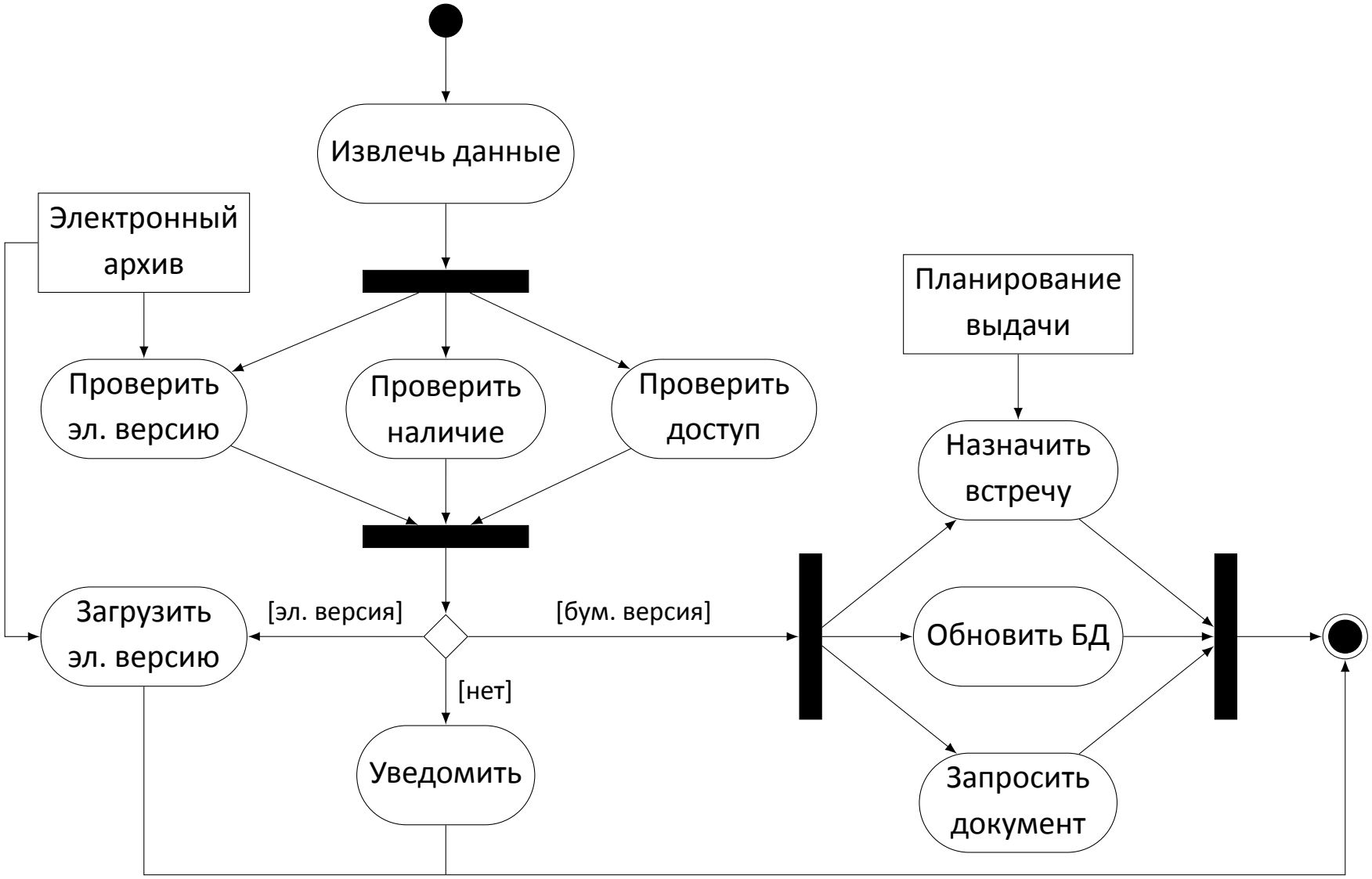


Диаграмма деятельности UML (англ. *activity diagram*) для обработки запроса на выдачу документа.

Модель процесса

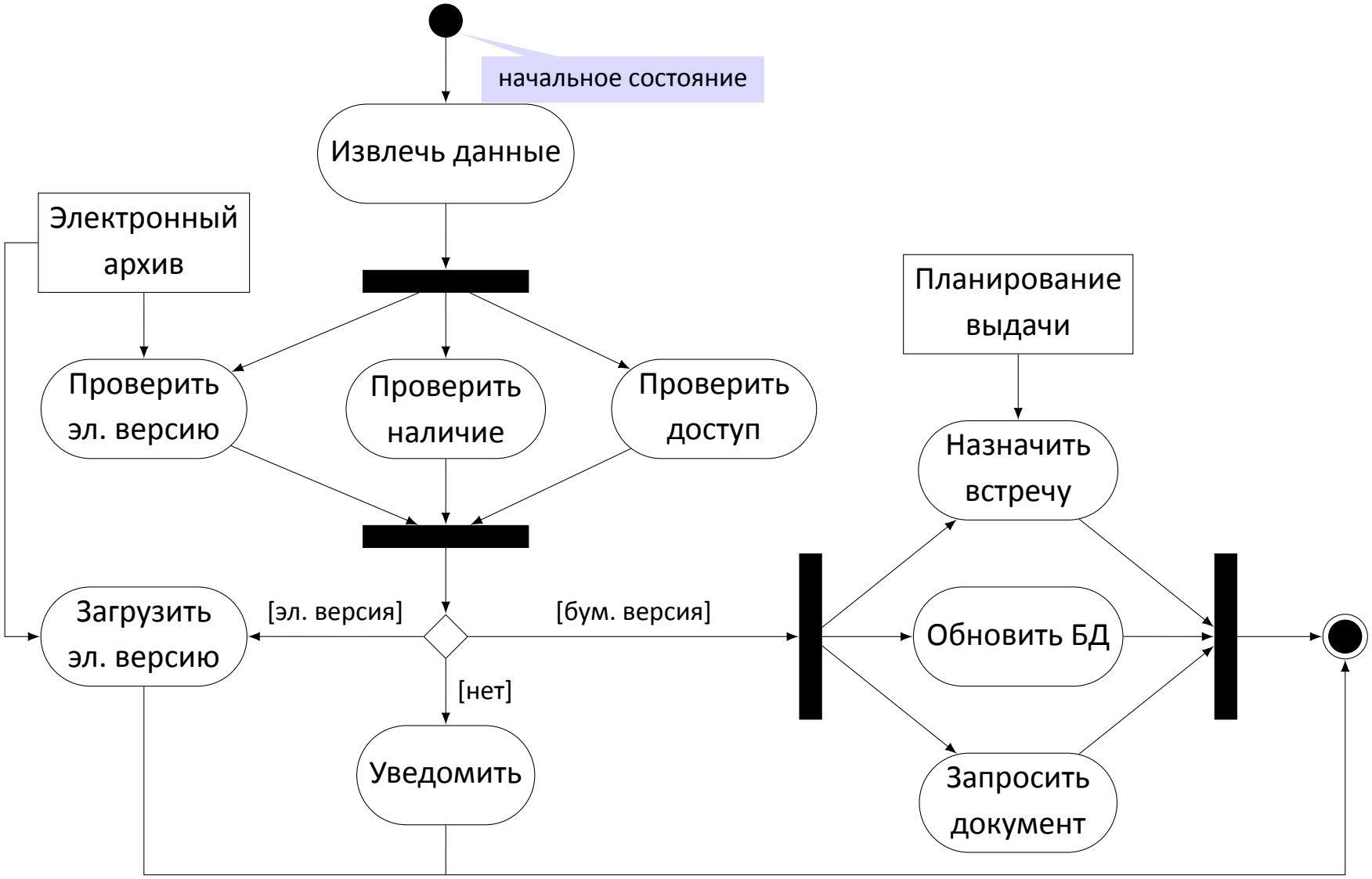


Диаграмма деятельности UML (англ. *activity diagram*) для обработки запроса на выдачу документа.

Модель процесса

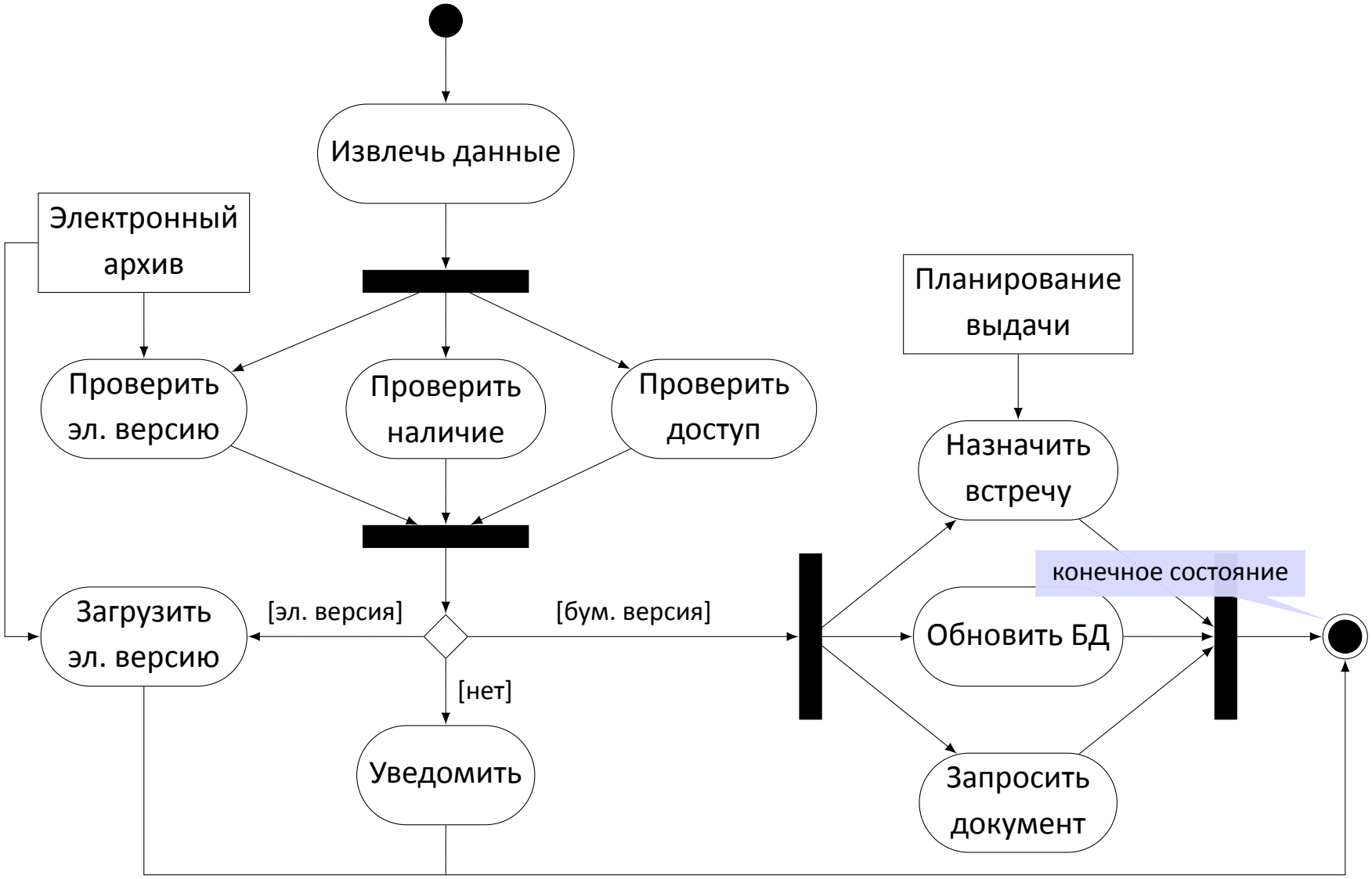


Диаграмма деятельности UML (англ. *activity diagram*) для обработки запроса на выдачу документа.

Модель процесса

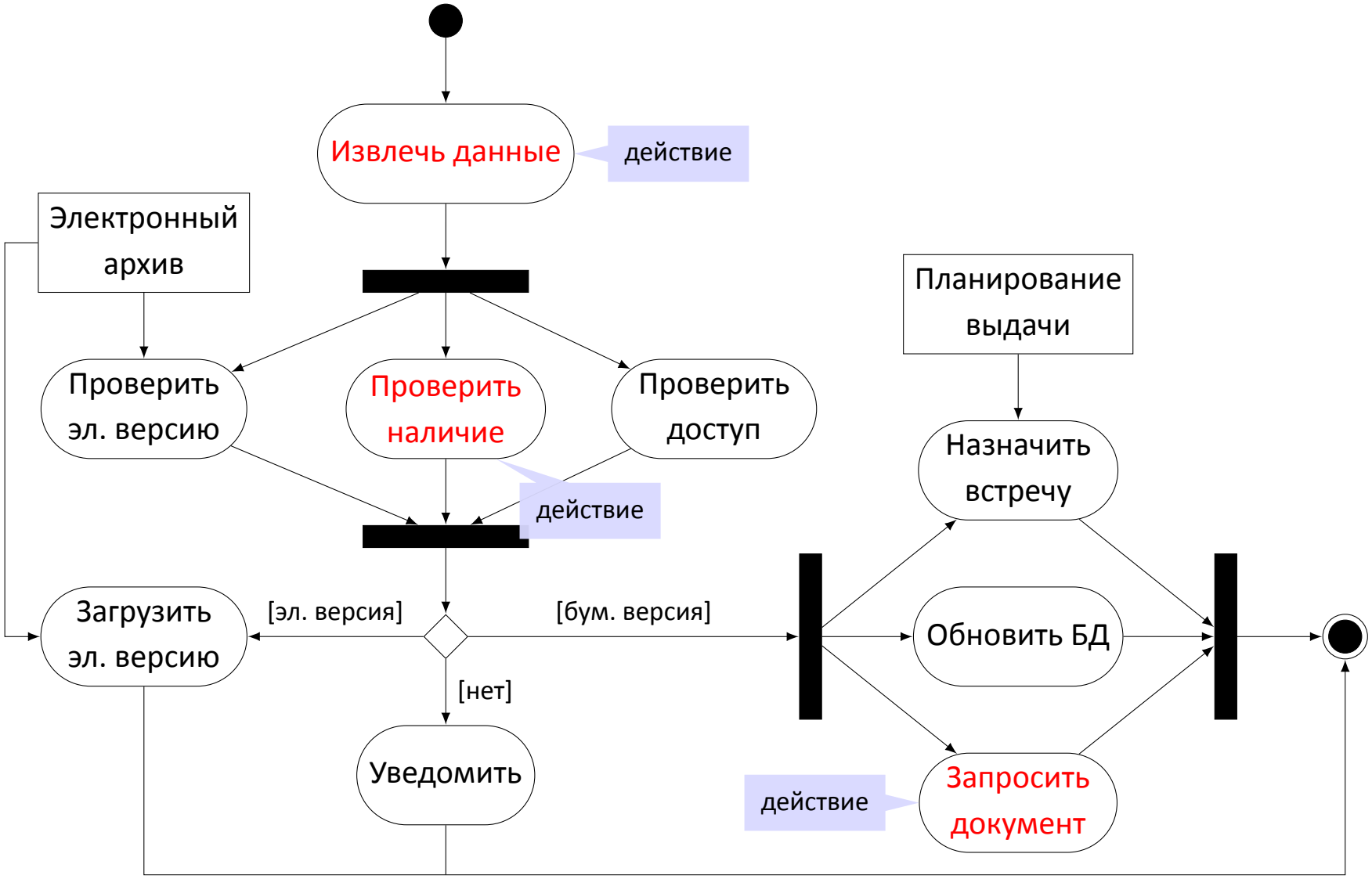


Диаграмма деятельности UML (англ. *activity diagram*) для обработки запроса на выдачу документа.

Модель процесса

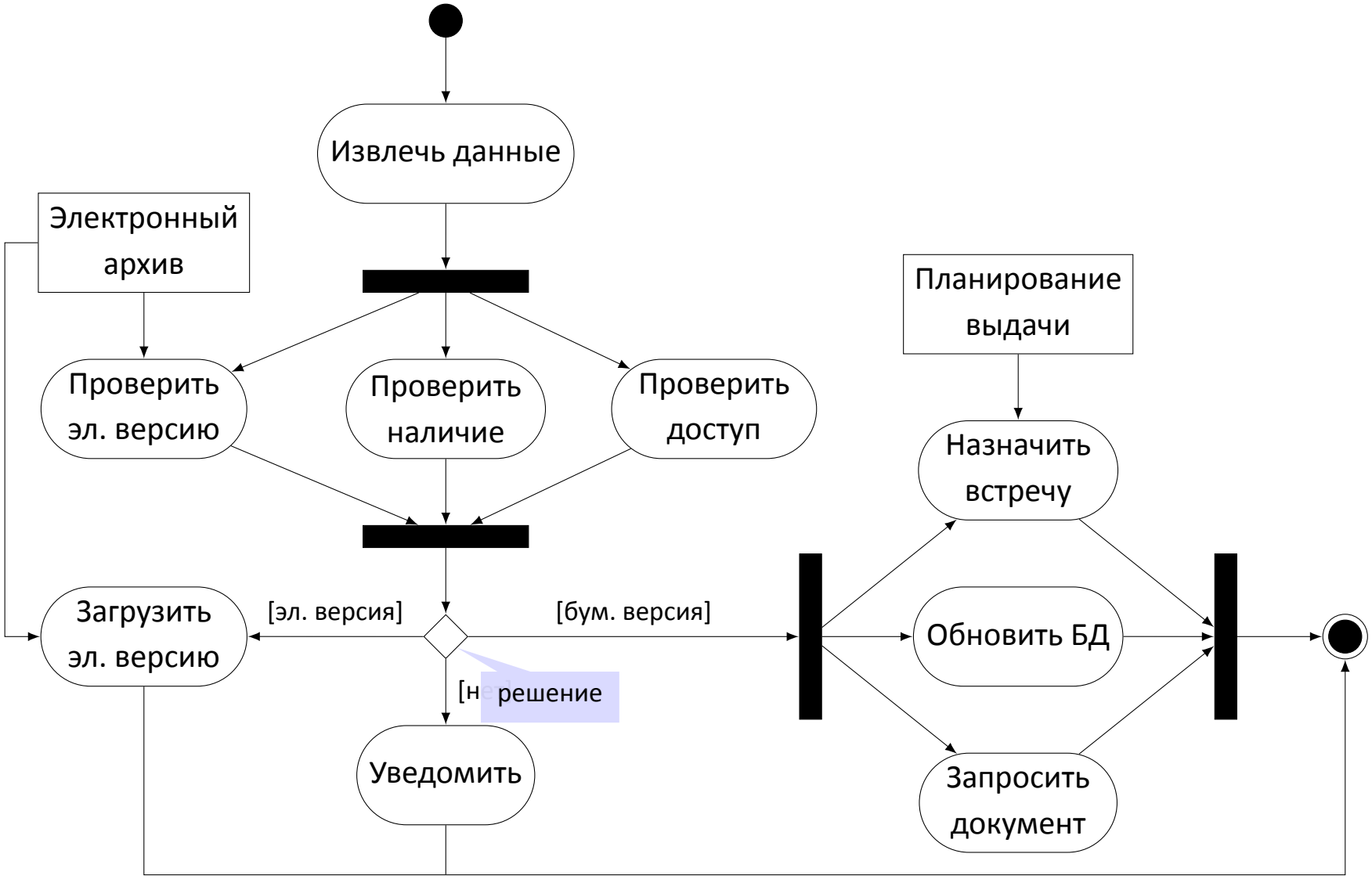


Диаграмма деятельности UML (англ. *activity diagram*) для обработки запроса на выдачу документа.

Модель процесса

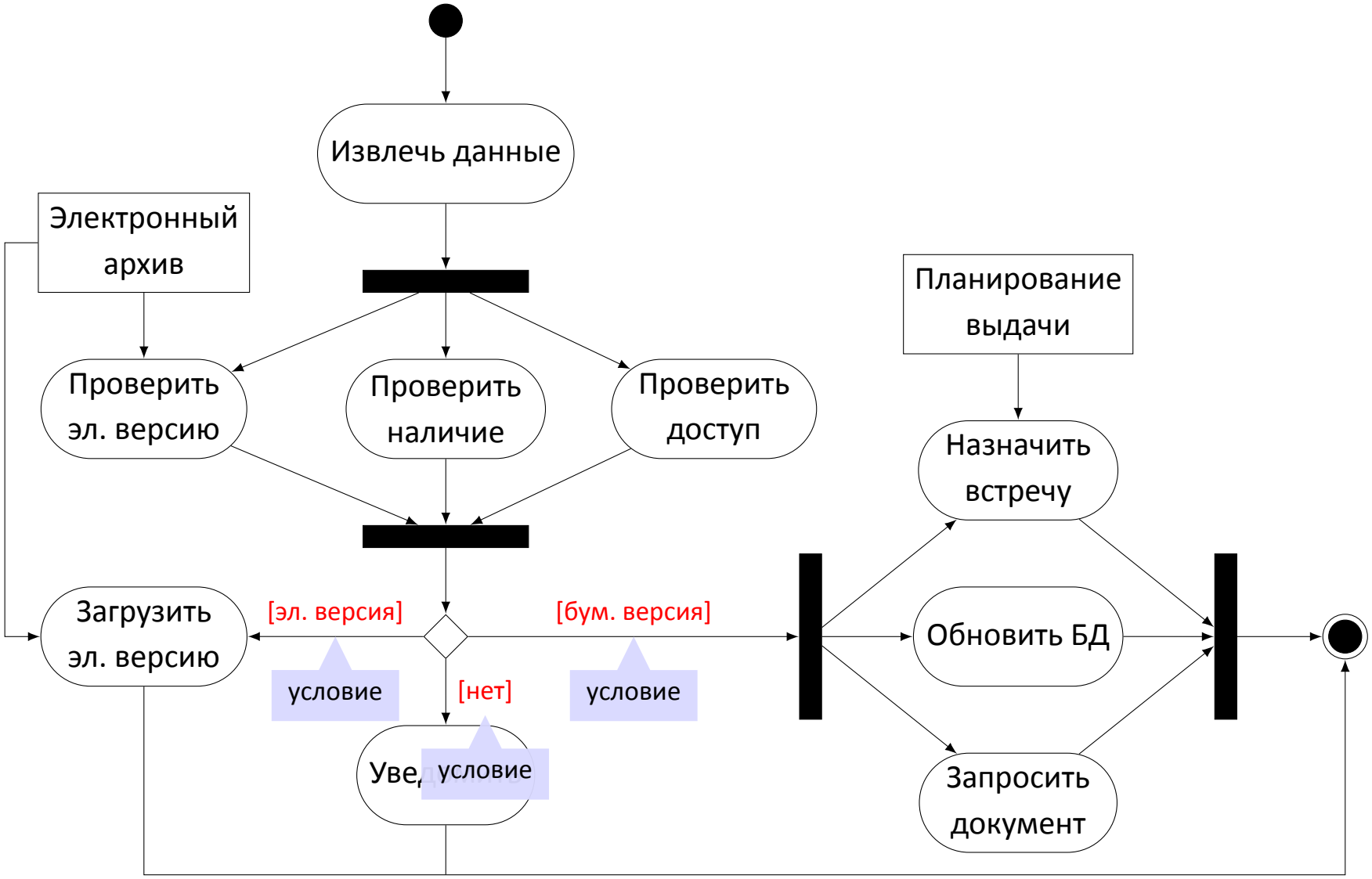


Диаграмма деятельности UML (англ. *activity diagram*) для обработки запроса на выдачу документа.

Модель процесса

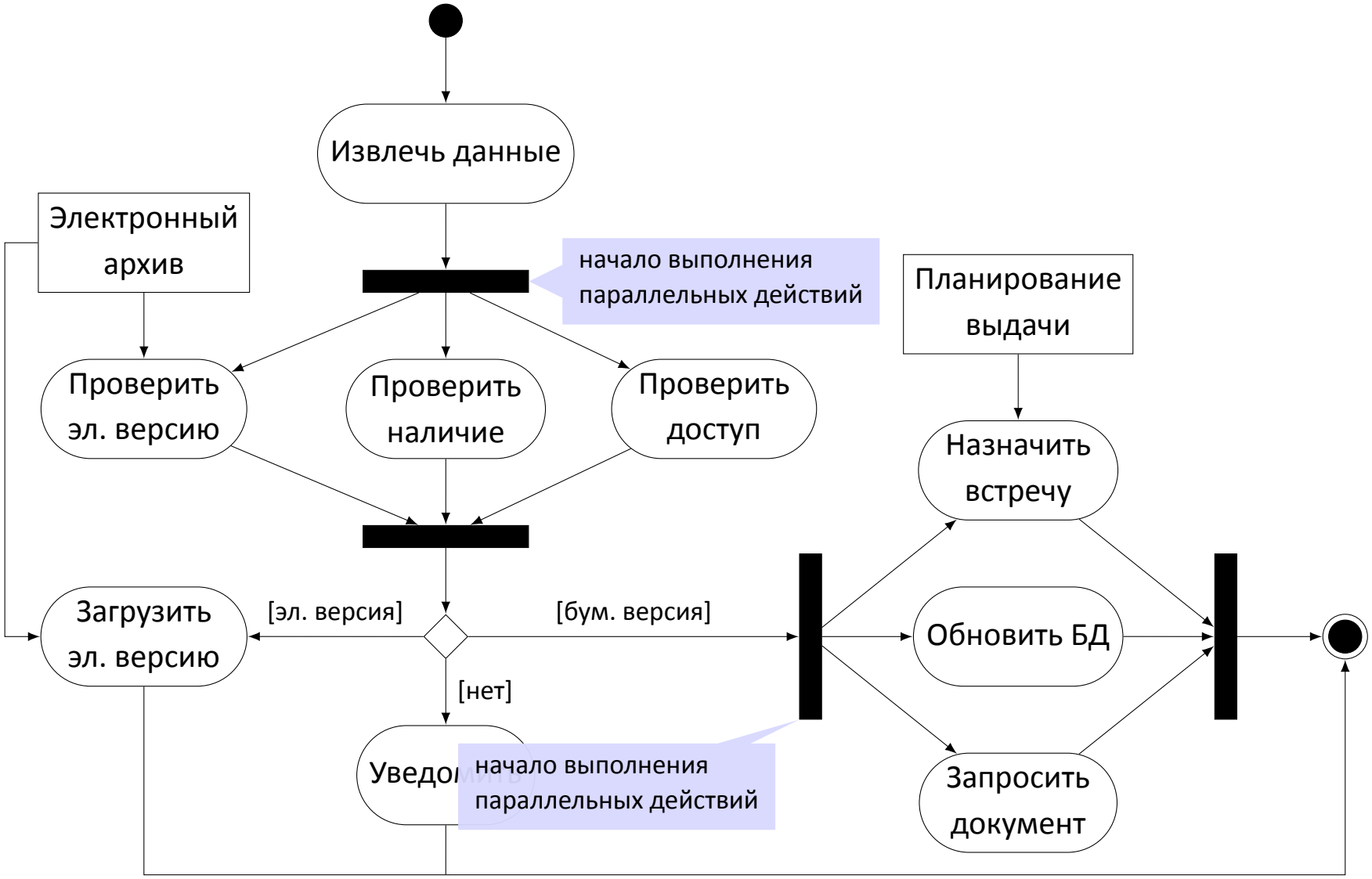


Диаграмма деятельности UML (англ. *activity diagram*) для обработки запроса на выдачу документа.

Модель процесса

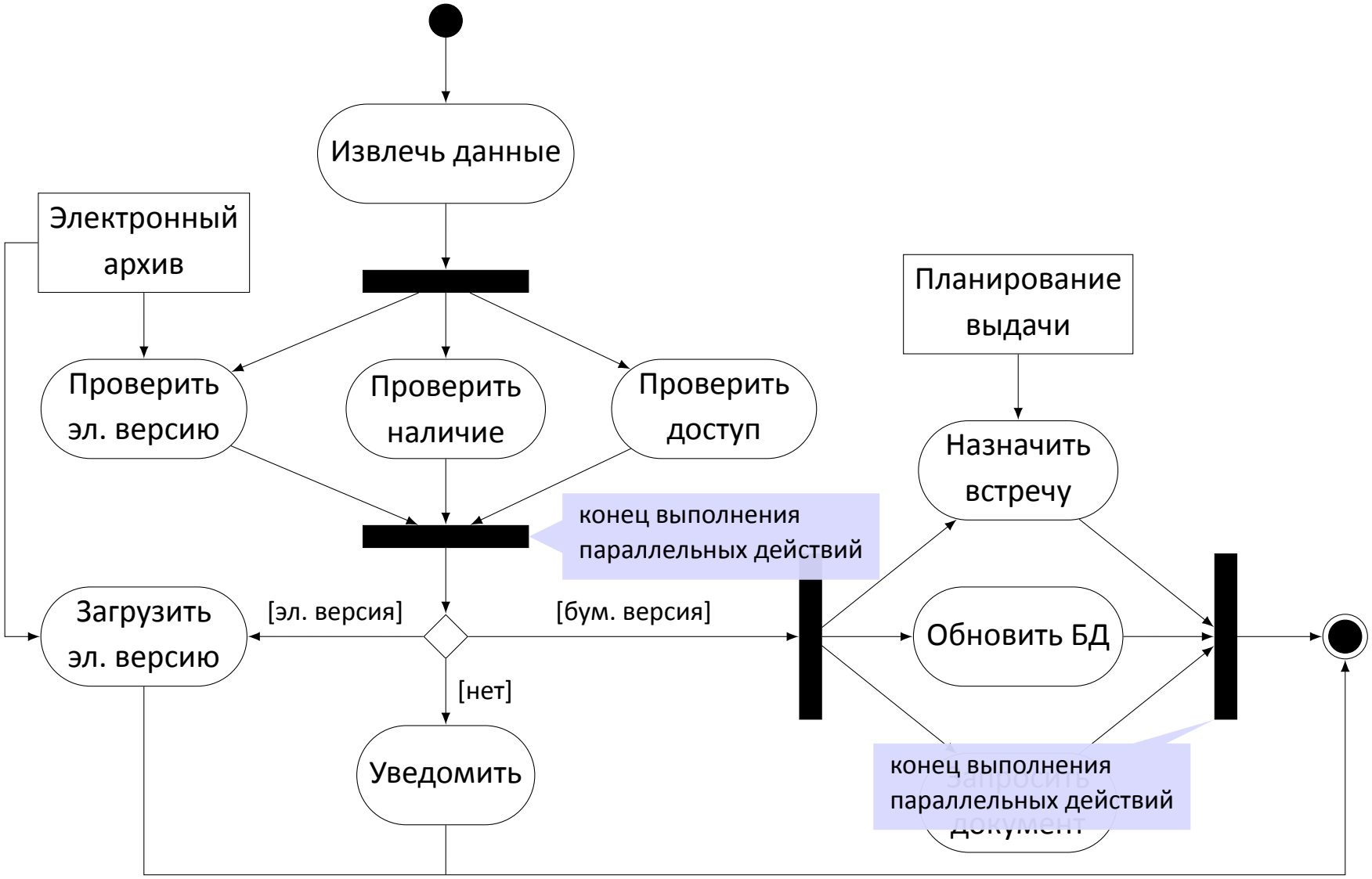


Диаграмма деятельности UML (англ. *activity diagram*) для обработки запроса на выдачу документа.

Модель процесса

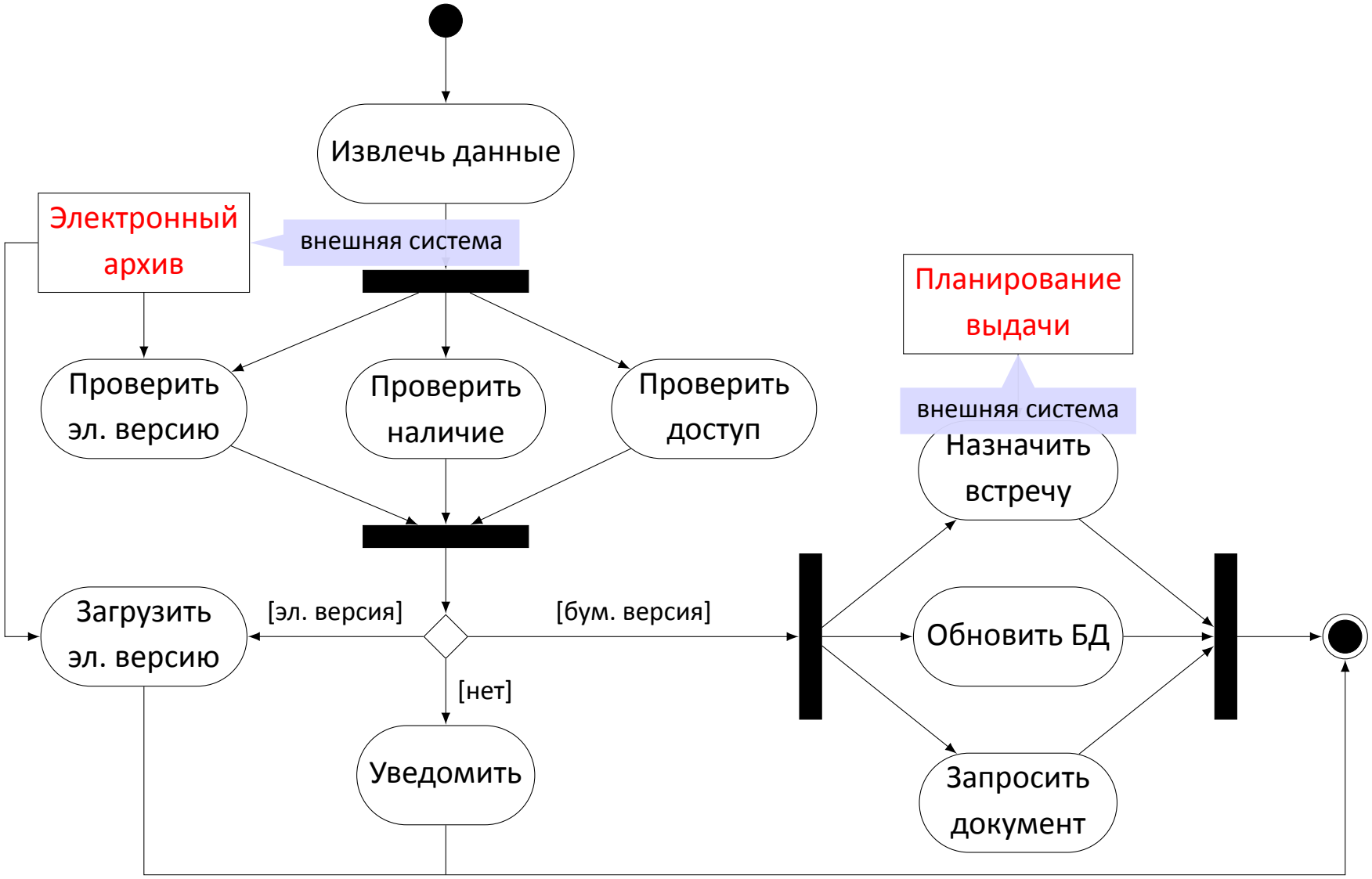


Диаграмма деятельности UML (англ. *activity diagram*) для обработки запроса на выдачу документа.

Модели взаимодействия

Цели:

- отладка взаимодействия с пользователями;
- выработка пользовательских требований;
- определение возможных «узких мест» и проблем коммуникации;
- отработка производительности (англ. *performance*) и надежности (англ. *dependability*).

Типы моделей:

- варианты использования — моделирование взаимодействия системы с актерами (пользователями или другими системами);
- диаграммы последовательностей — моделирование взаимодействия компонентов системы.

Варианты использования

Взаимодействие между читателем и библиотекарем на диаграмме вариантов использования (англ. *use case diagram*)

Актеры: читатель, библиотекарь

Описание: Библиотекарь выдает книгу на руки пользователю и вносит соответствующие данные в систему.

Данные: идентификатор книги, дата выдачи, кому выдана, на какой срок.

Побуждение: команда системы, полученная от библиотекаря.

Отклик: подтверждение внесения изменений в систему.

Доп. условия: библиотекарь должен быть авторизован в системе; читатель должен иметь разрешение на выдачу книги.

Варианты использования

Взаимодействие между читателем и библиотекарем на диаграмме вариантов использования (англ. *use case diagram*)

Актеры: читатель, библиотекарь

Описание: Библиотекарь выдает книгу на руки пользователю и вносит соответствующие данные в систему.

Данные: идентификатор книги, дата выдачи, кому выдана, на какой срок.

Побуждение: команда системы, полученная от библиотекаря.

Отклик: подтверждение внесения изменений в систему.

Доп. условия: библиотекарь должен быть авторизован в системе; читатель должен иметь разрешение на выдачу книги.

Варианты использования

Взаимодействие между читателем и библиотекарем на диаграмме вариантов использования (англ. *use case diagram*)

Актеры: читатель, библиотекарь

Описание: Библиотекарь выдает книгу на руки пользователю и вносит соответствующие данные в систему.

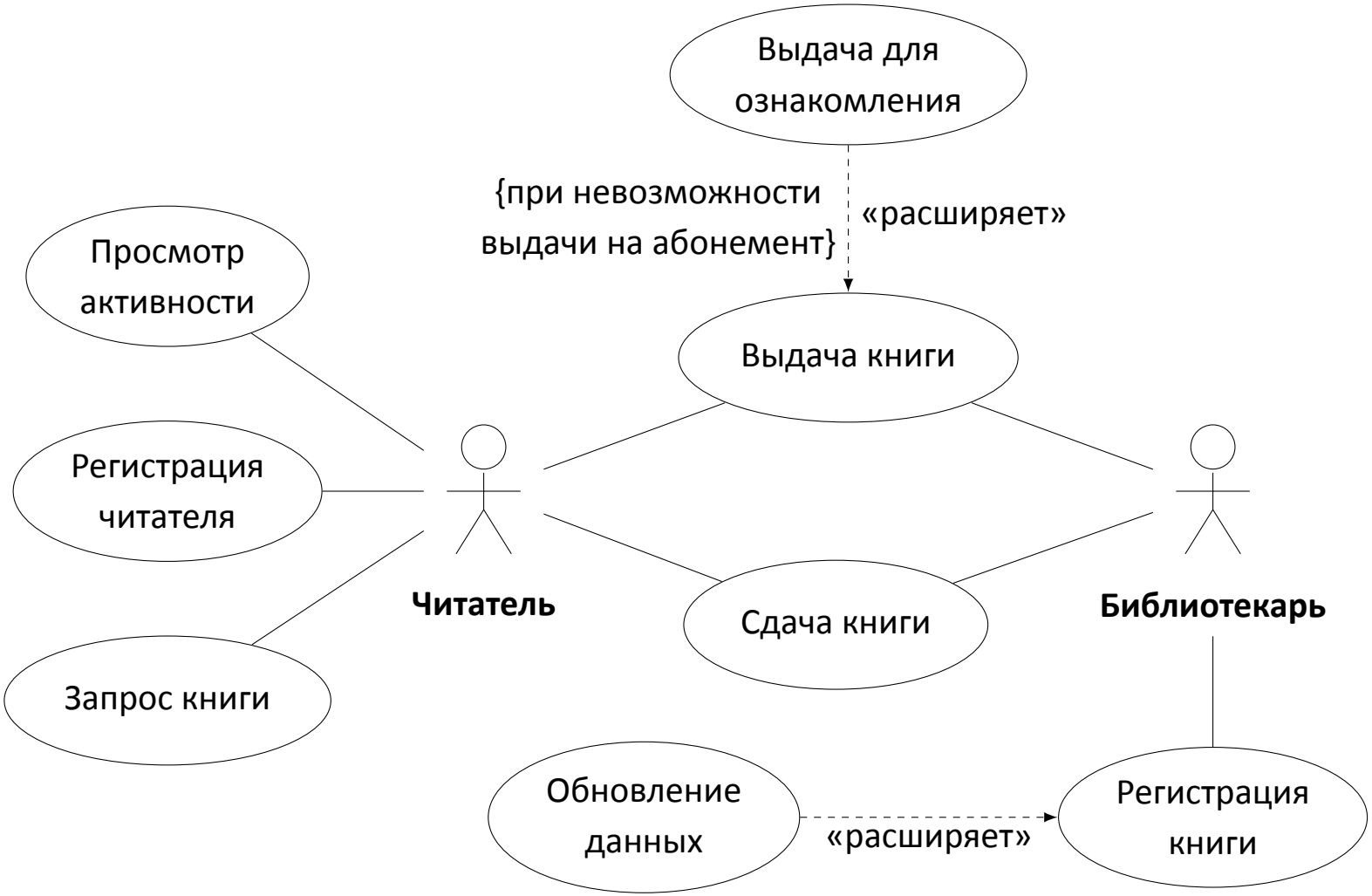
Данные: идентификатор книги, дата выдачи, кому выдана, на какой срок.

Побуждение: команда системы, полученная от библиотекаря.

Отклик: подтверждение внесения изменений в систему.

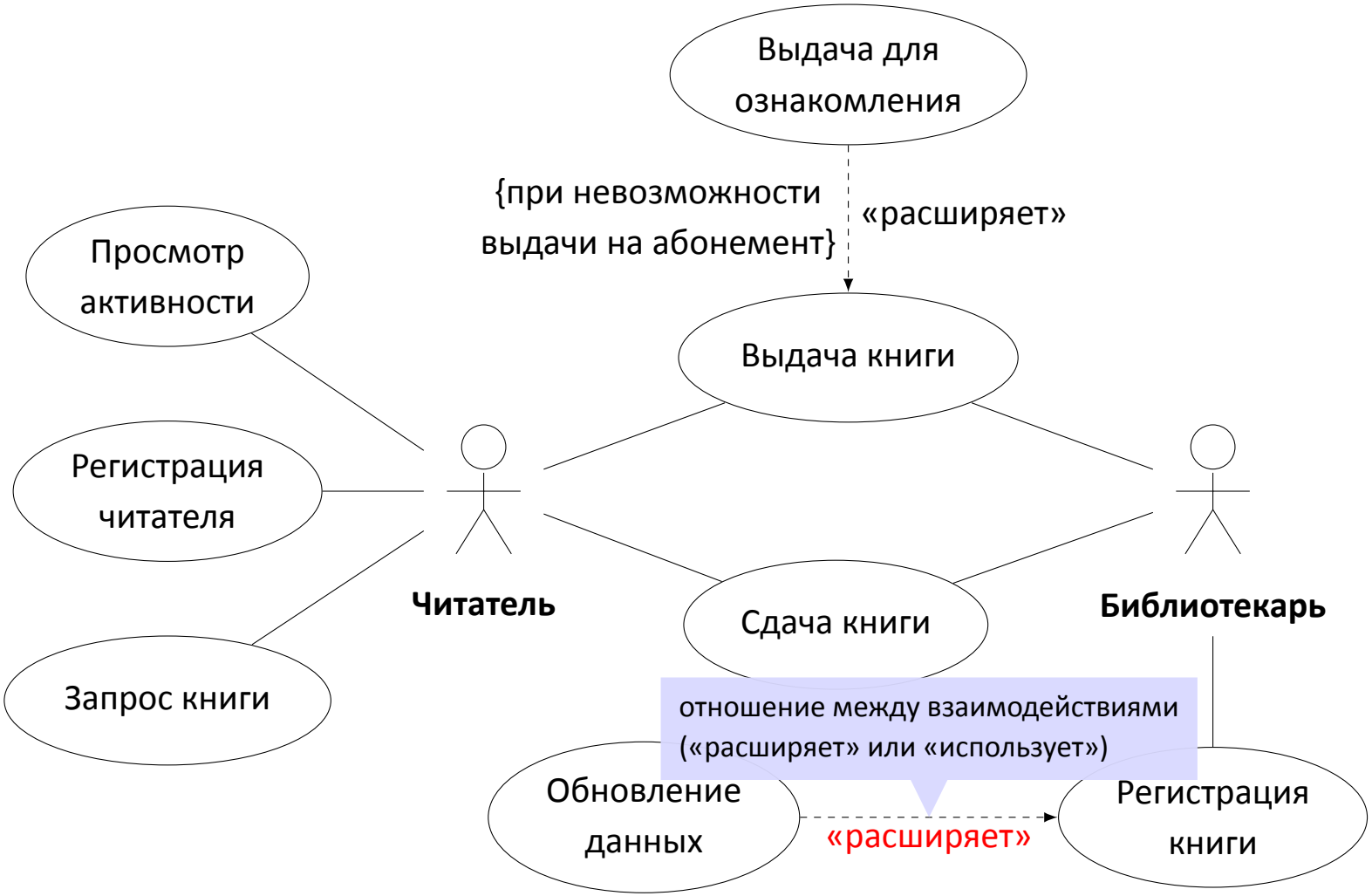
Доп. условия: библиотекарь должен быть авторизован в системе; читатель должен иметь разрешение на выдачу книги.

Варианты использования (продолжение)



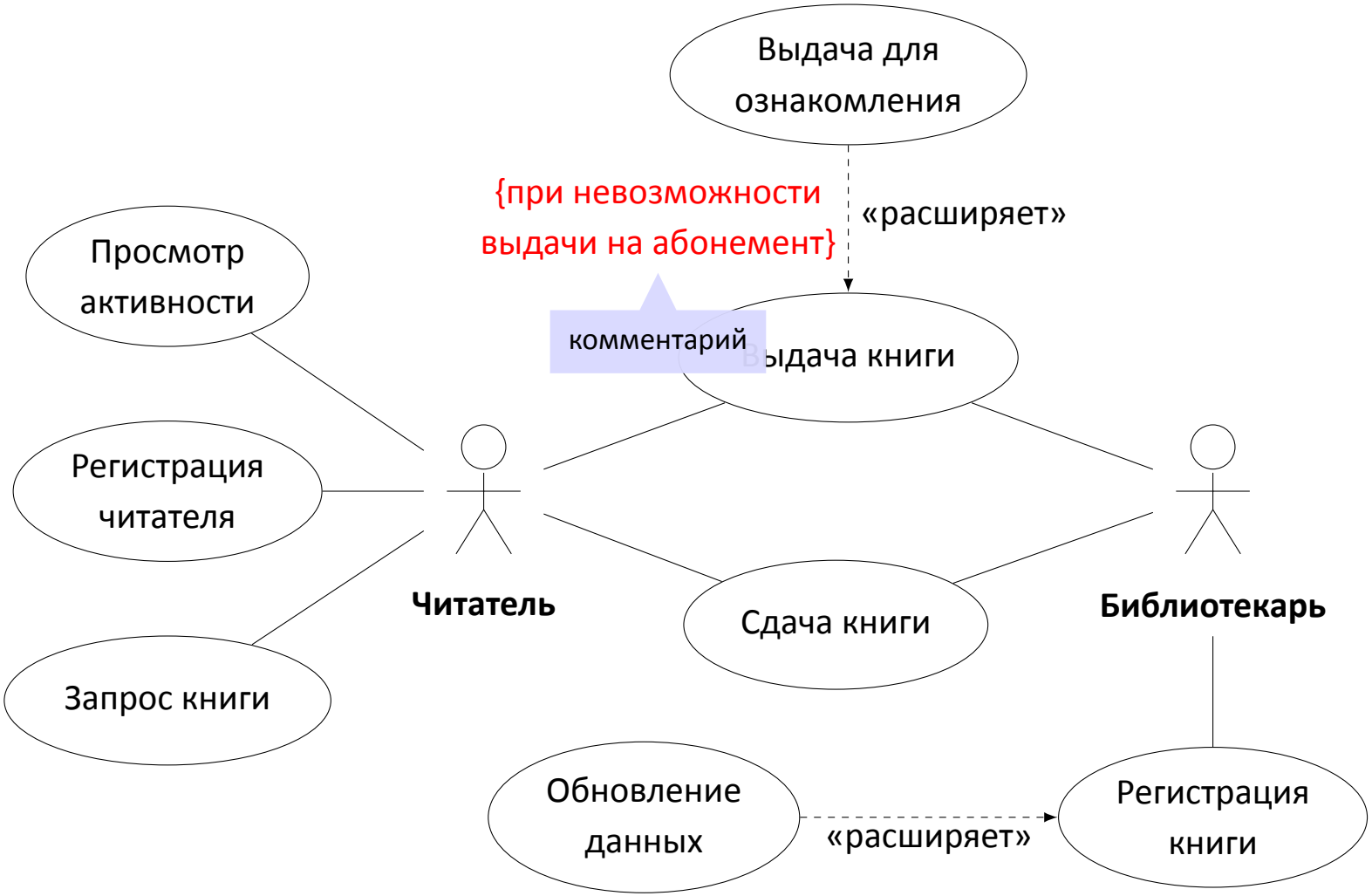
Более сложный пример, демонстрирующий связи между взаимодействиями

Варианты использования (продолжение)



Более сложный пример, демонстрирующий связи между взаимодействиями

Варианты использования (продолжение)



Более сложный пример, демонстрирующий связи между взаимодействиями

Диаграммы последовательностей

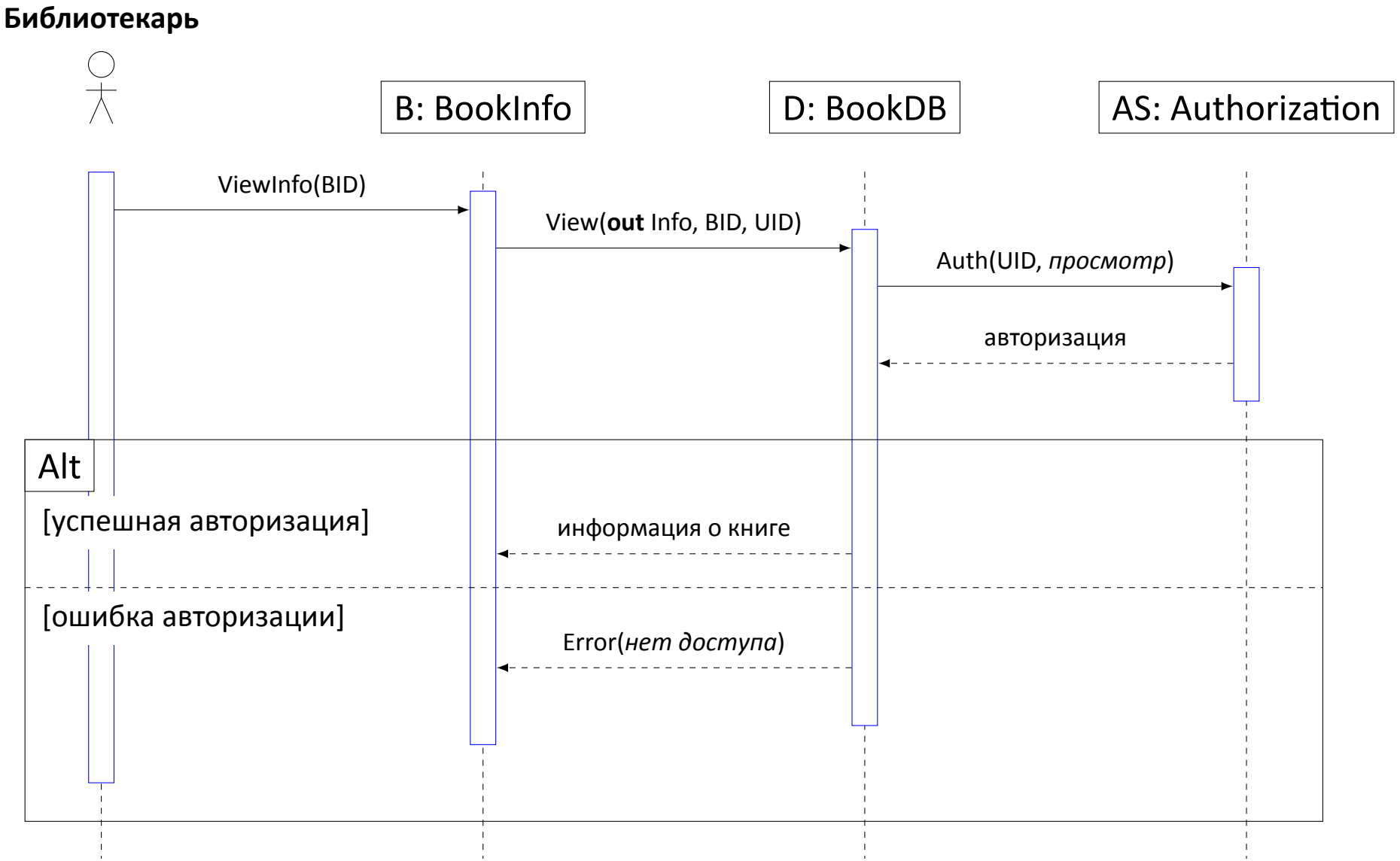


Диаграмма последовательности для просмотра информации о книге

Диаграммы последовательностей

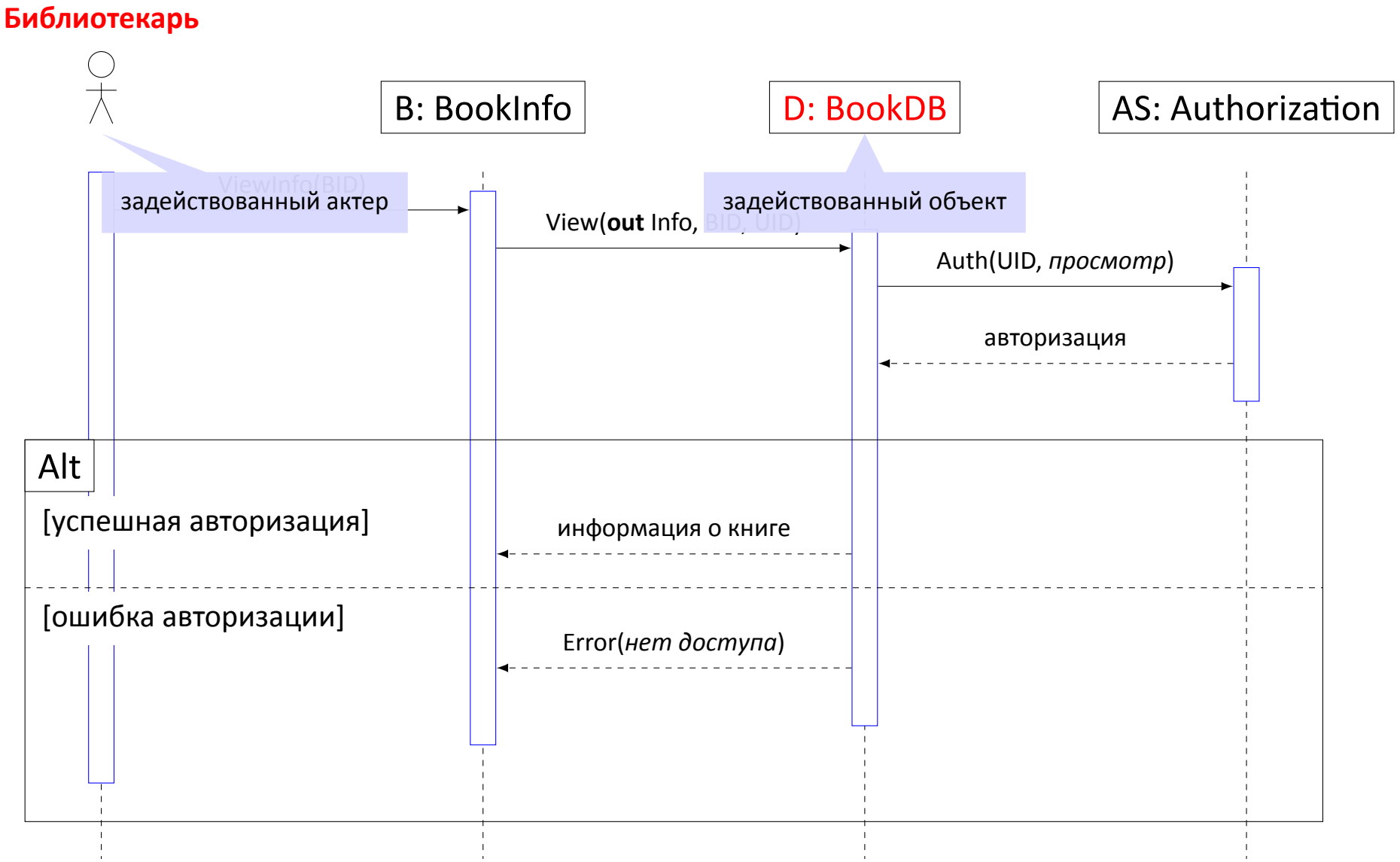


Диаграмма последовательности для просмотра информации о книге

Диаграммы последовательностей

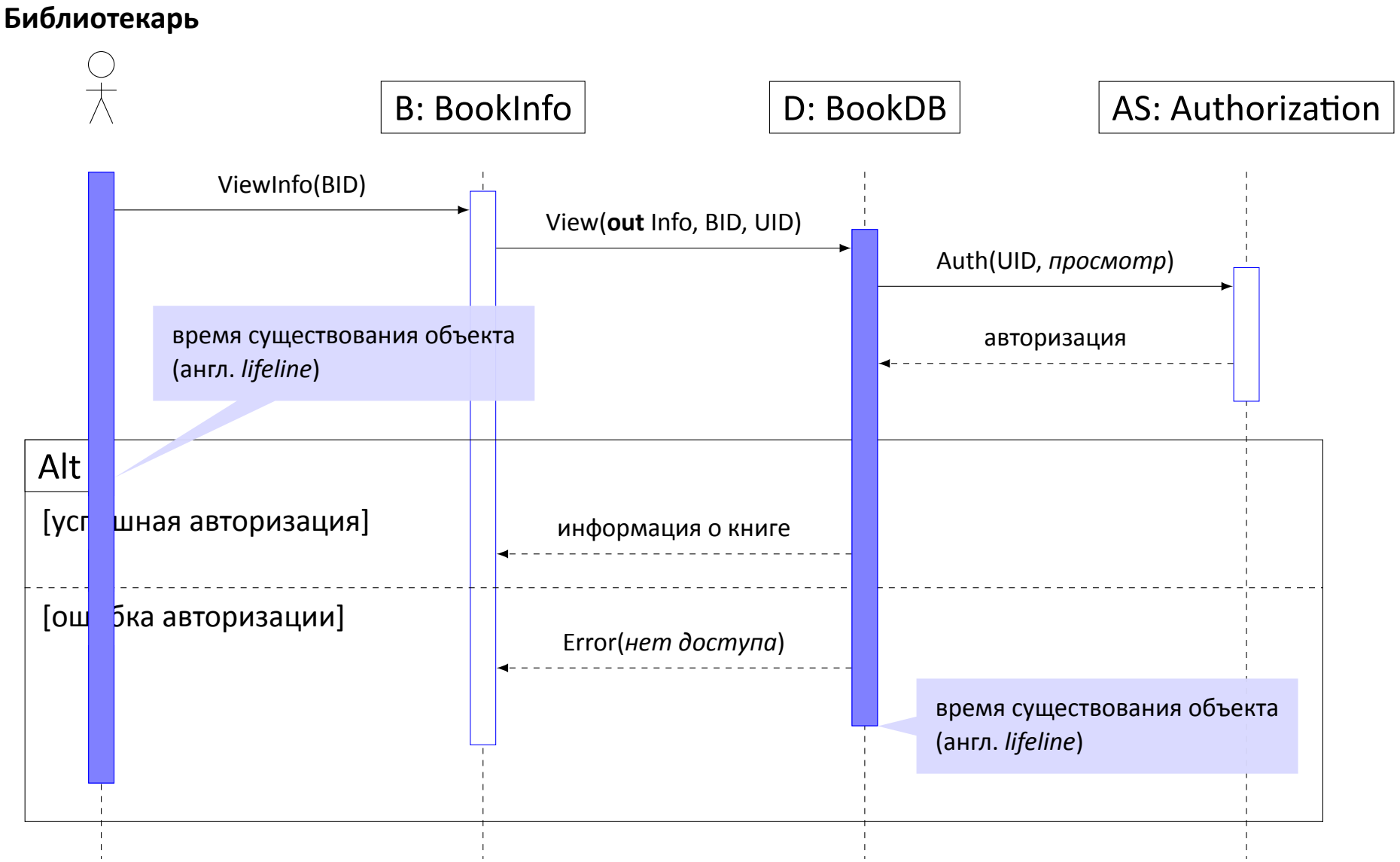


Диаграмма последовательности для просмотра информации о книге

Диаграммы последовательностей

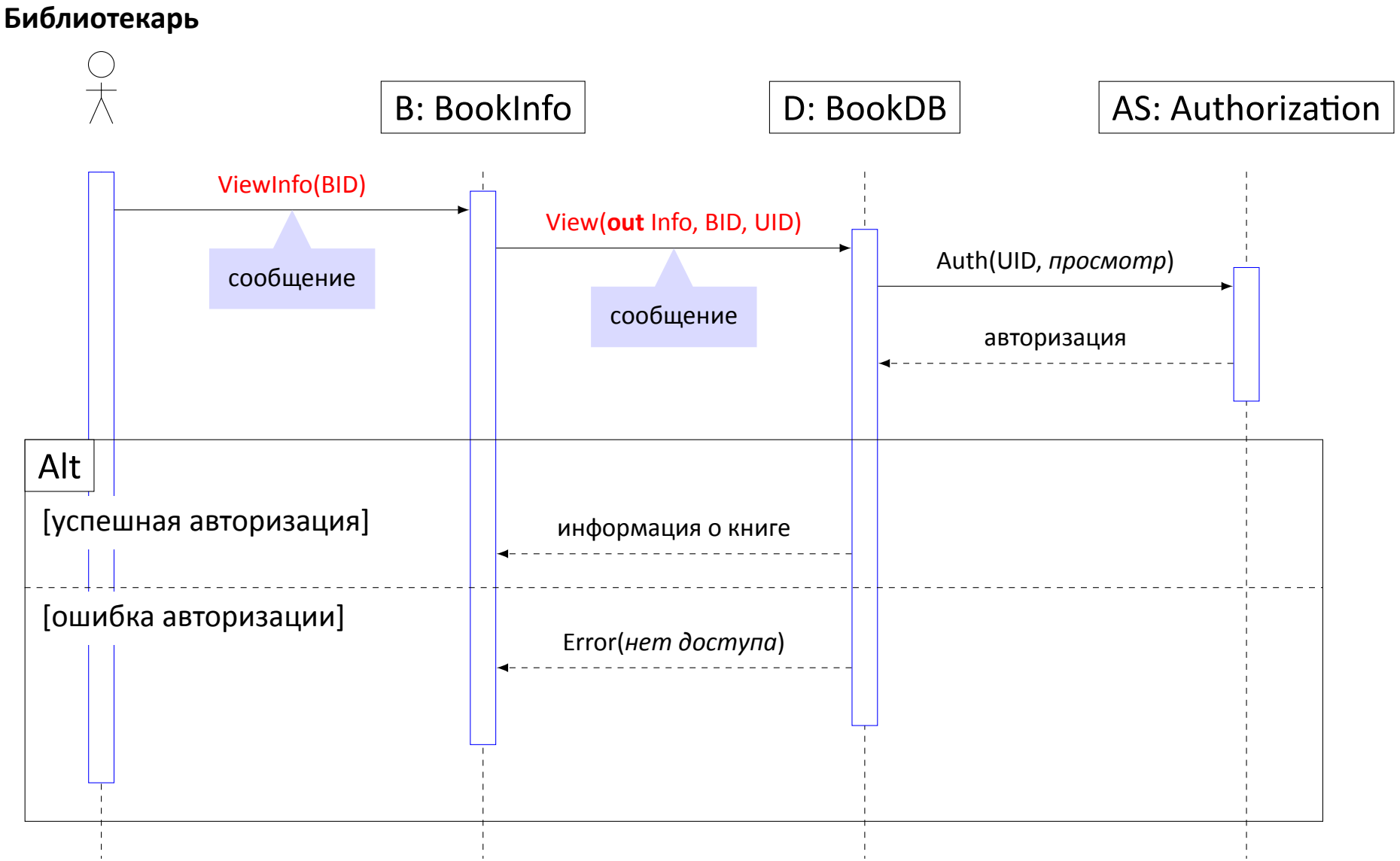


Диаграмма последовательности для просмотра информации о книге

Диаграммы последовательностей

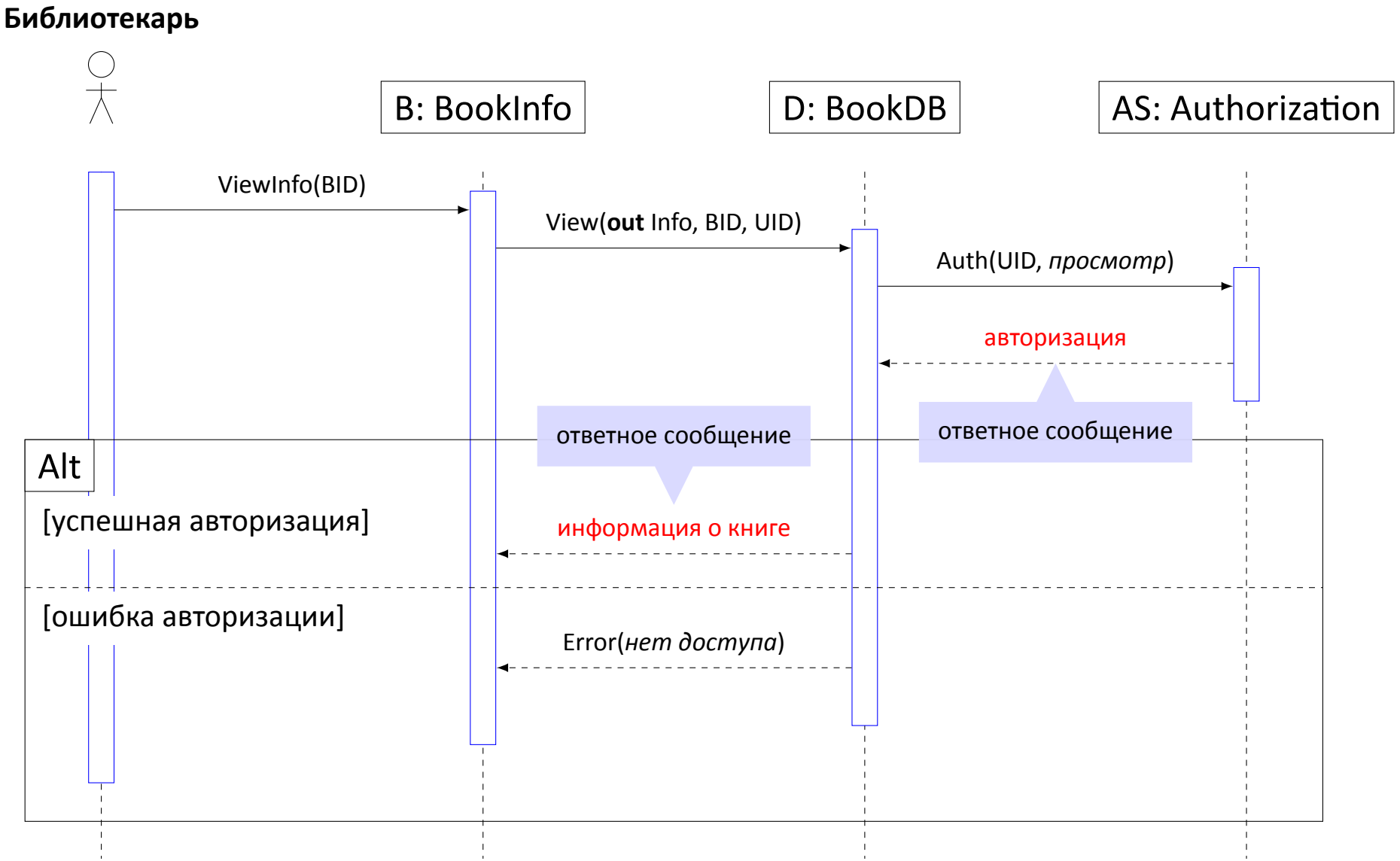


Диаграмма последовательности для просмотра информации о книге

Диаграммы последовательностей

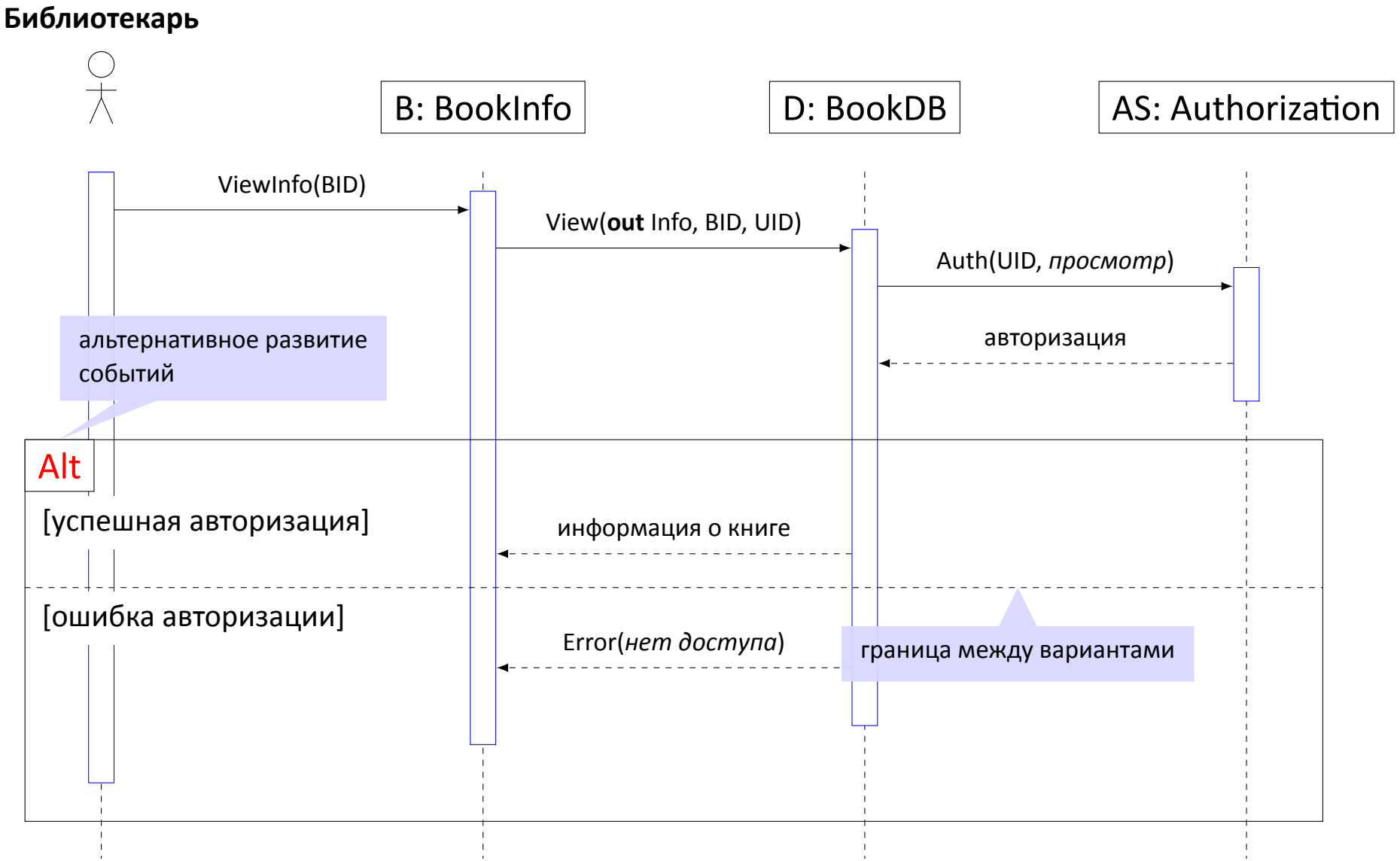


Диаграмма последовательности для просмотра информации о книге

Диаграммы последовательностей

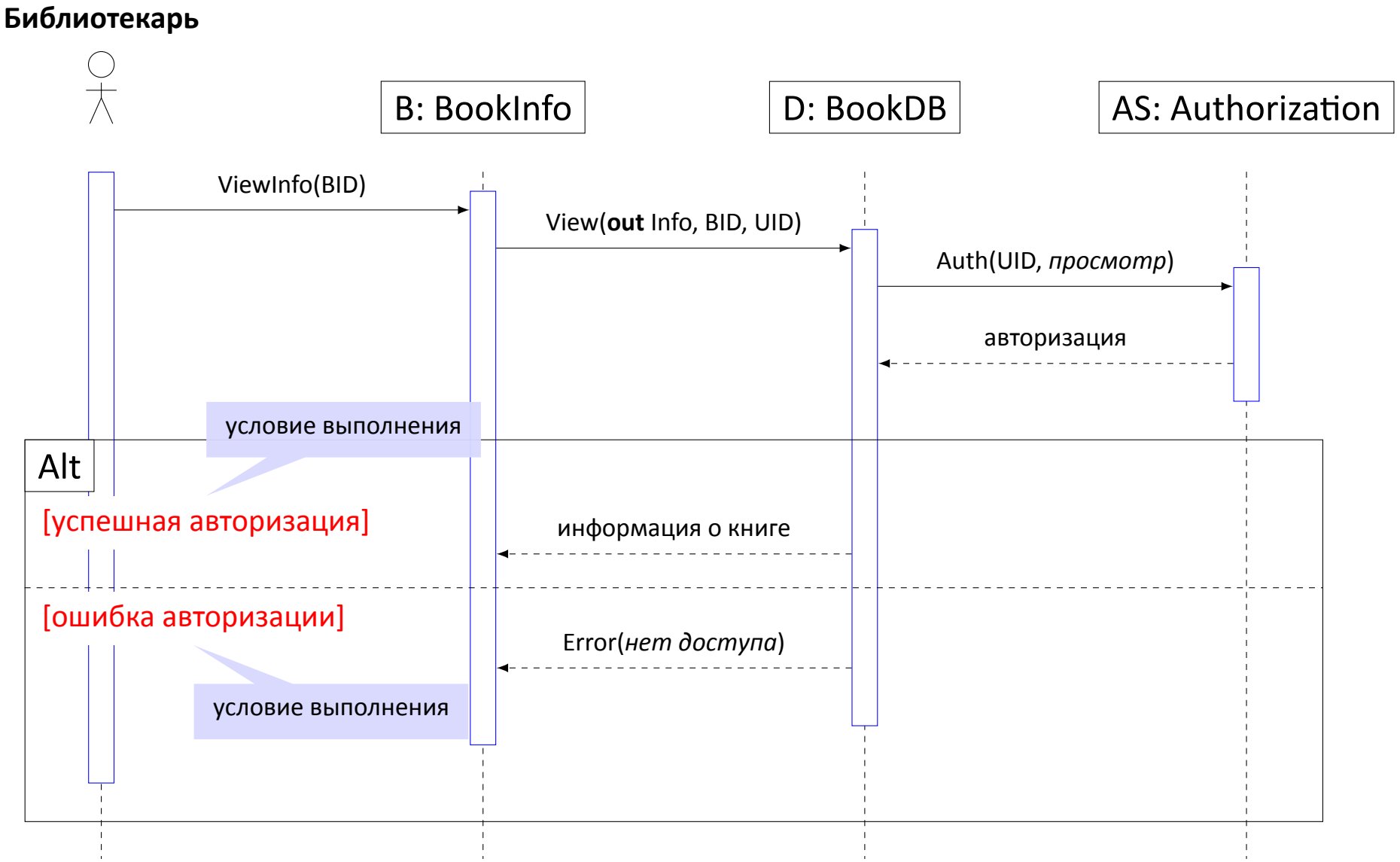


Диаграмма последовательности для просмотра информации о книге

Диаграммы последовательностей (продолжение)

Интерпретация примера:

1. **Библиотекарь** запрашивает информацию по книге по ее идентификатору **BID** (напр., ISBN). Для отображения информации создается экземпляр **B** класса **BookInfo**, отображающий информацию в виде таблицы.
2. **B** запрашивает базу данных **D**, предоставляя ей идентификатор пользователя **UID**.
3. База данных проверяет право **UID** на **просмотр сведений** о книге с помощью системы авторизации **AS**.
4. Если авторизация выполнена успешно, **D** возвращает данные о книге. **B** отображает их.
5. Если авторизация не удалась, возвращается **ошибка**. **B** отображает сведения об ошибке.

Структурные модели

Цели:

- определение архитектуры системы;
- определение структур хранения данных.

Диаграмма классов UML:

Диаграмма для нескольких классов предметной области «Библиотека». *Курсив* = другие классы.

Структурные модели

Цели:

- определение архитектуры системы;
- определение структур хранения данных.

Диаграмма классов UML:

Диаграмма для нескольких классов предметной области «Библиотека». *Курсив* = другие классы.

Структурные модели

Цели:

- определение архитектуры системы;
- определение структур хранения данных.

Диаграмма классов UML:

Диаграмма для нескольких классов предметной области «Библиотека». *Курсив* = другие классы.

Структурные модели

Цели:

- определение архитектуры системы;
- определение структур хранения данных.

Диаграмма классов UML:

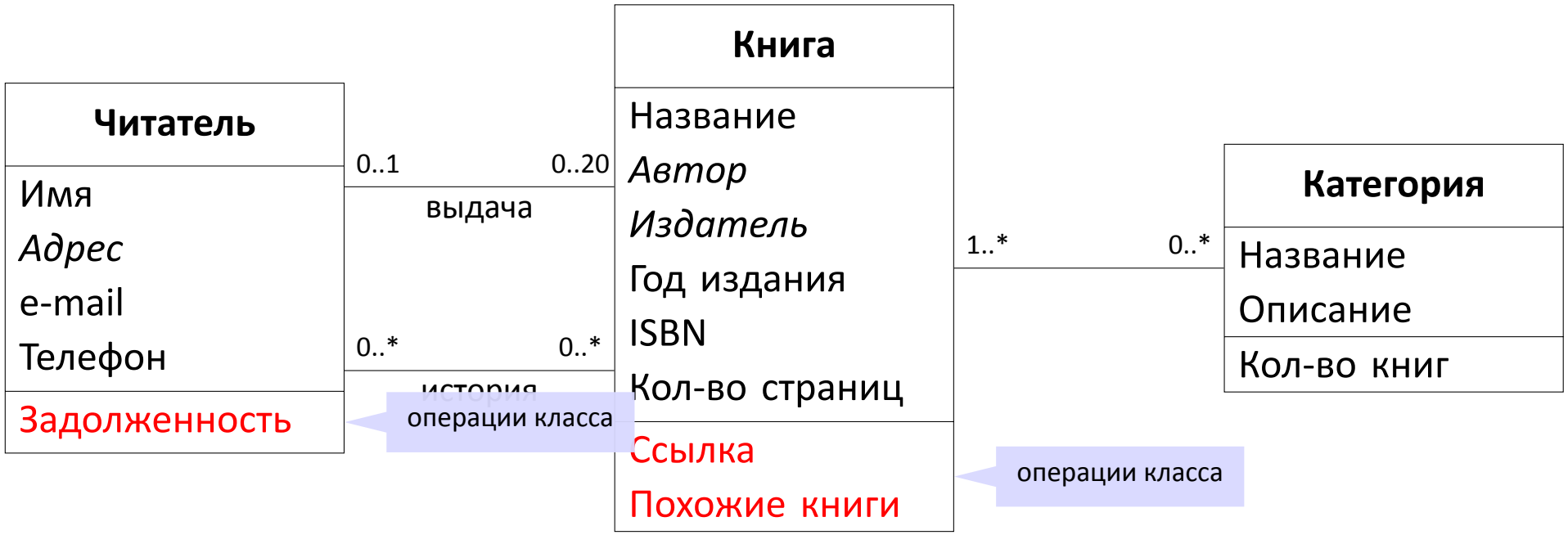


Диаграмма для нескольких классов предметной области «Библиотека». *Курсив* = другие классы.

Структурные модели

Цели:

- определение архитектуры системы;
- определение структур хранения данных.

Диаграмма классов UML:

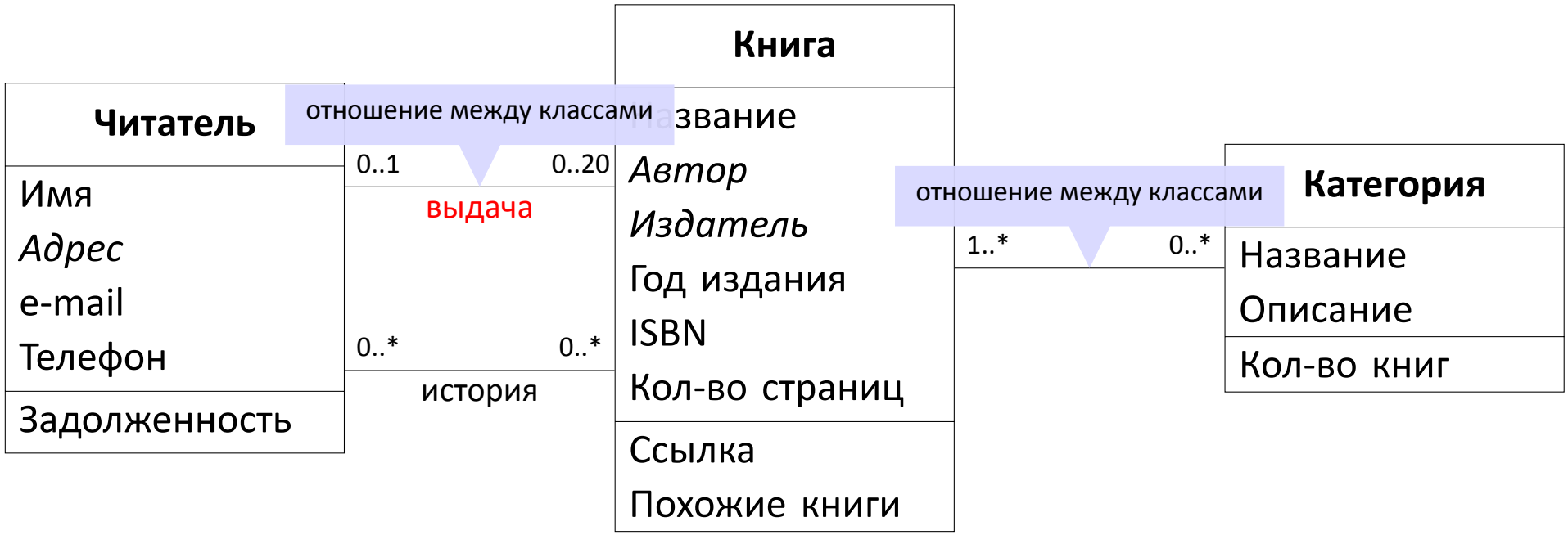


Диаграмма для нескольких классов предметной области «Библиотека». *Курсив* = другие классы.

Структурные модели

Цели:

- определение архитектуры системы;
- определение структур хранения данных.

Диаграмма классов UML:

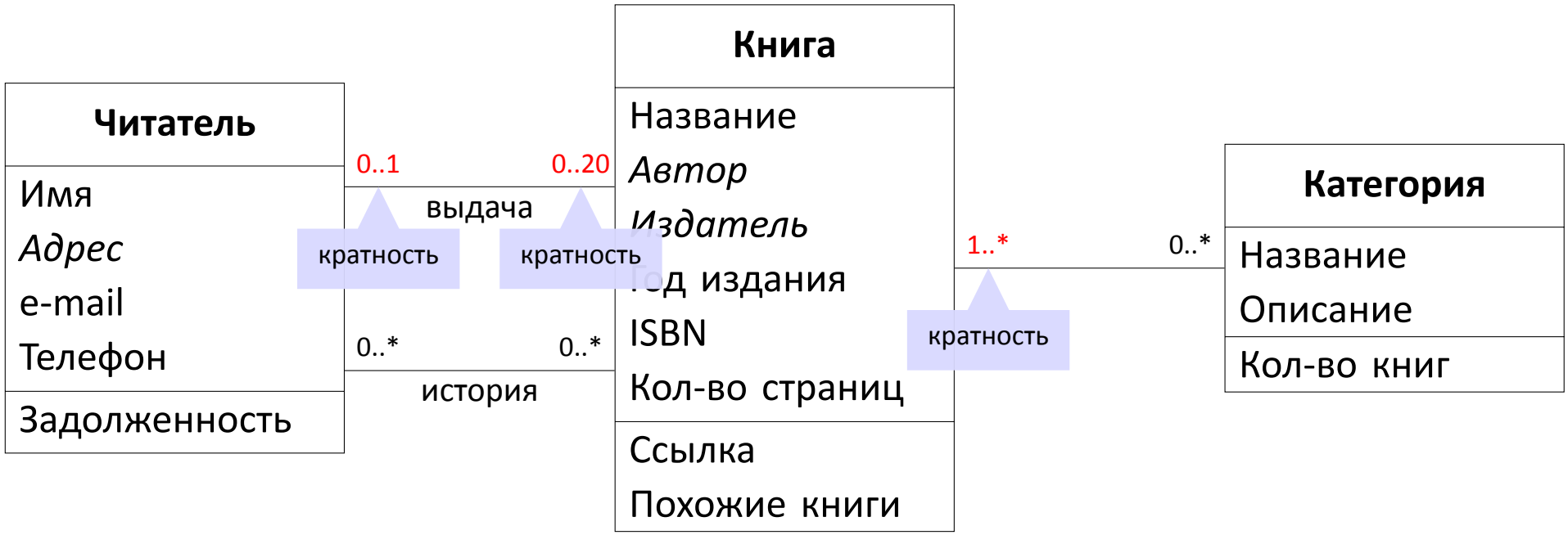


Диаграмма для нескольких классов предметной области «Библиотека». *Курсив* = другие классы.

Структурные модели

Цели:

- определение архитектуры системы;
- определение структур хранения данных.

Диаграмма классов UML:

Диаграмма для нескольких классов предметной области «Библиотека». *Курсив* = другие классы.

Структурные модели

Цели:

- определение архитектуры системы;
- определение структур хранения данных.

Диаграмма классов UML:

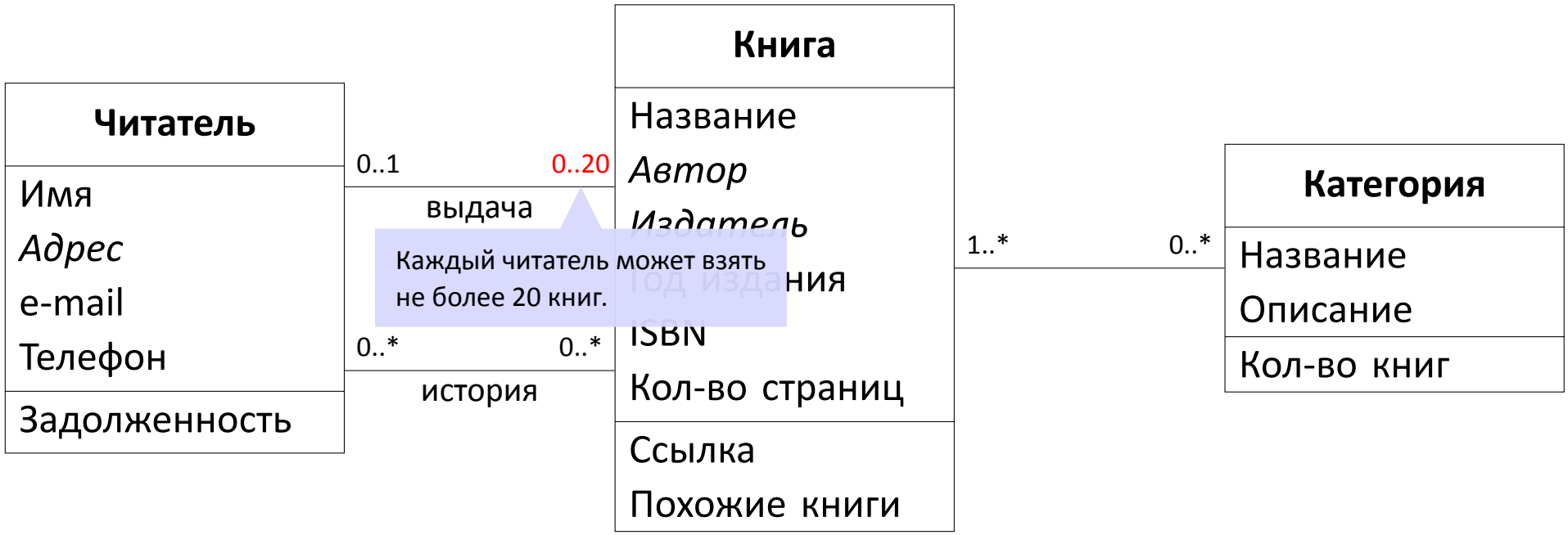


Диаграмма для нескольких классов предметной области «Библиотека». *Курсив* = другие классы.

Структурные модели

Цели:

- определение архитектуры системы;
- определение структур хранения данных.

Диаграмма классов UML:

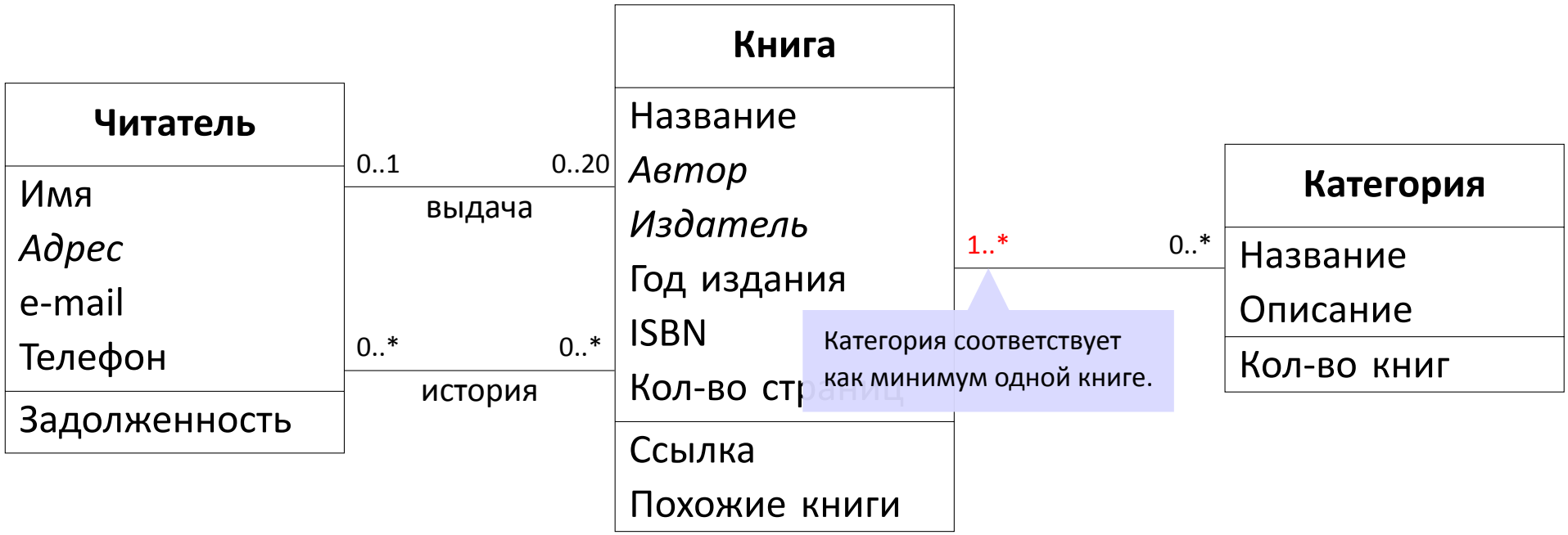


Диаграмма для нескольких классов предметной области «Библиотека». *Курсив* = другие классы.

Структурные модели

Цели:

- определение архитектуры системы;
- определение структур хранения данных.

Диаграмма классов UML:

Диаграмма для нескольких классов предметной области «Библиотека». *Курсив* = другие классы.

Отношения обобщения

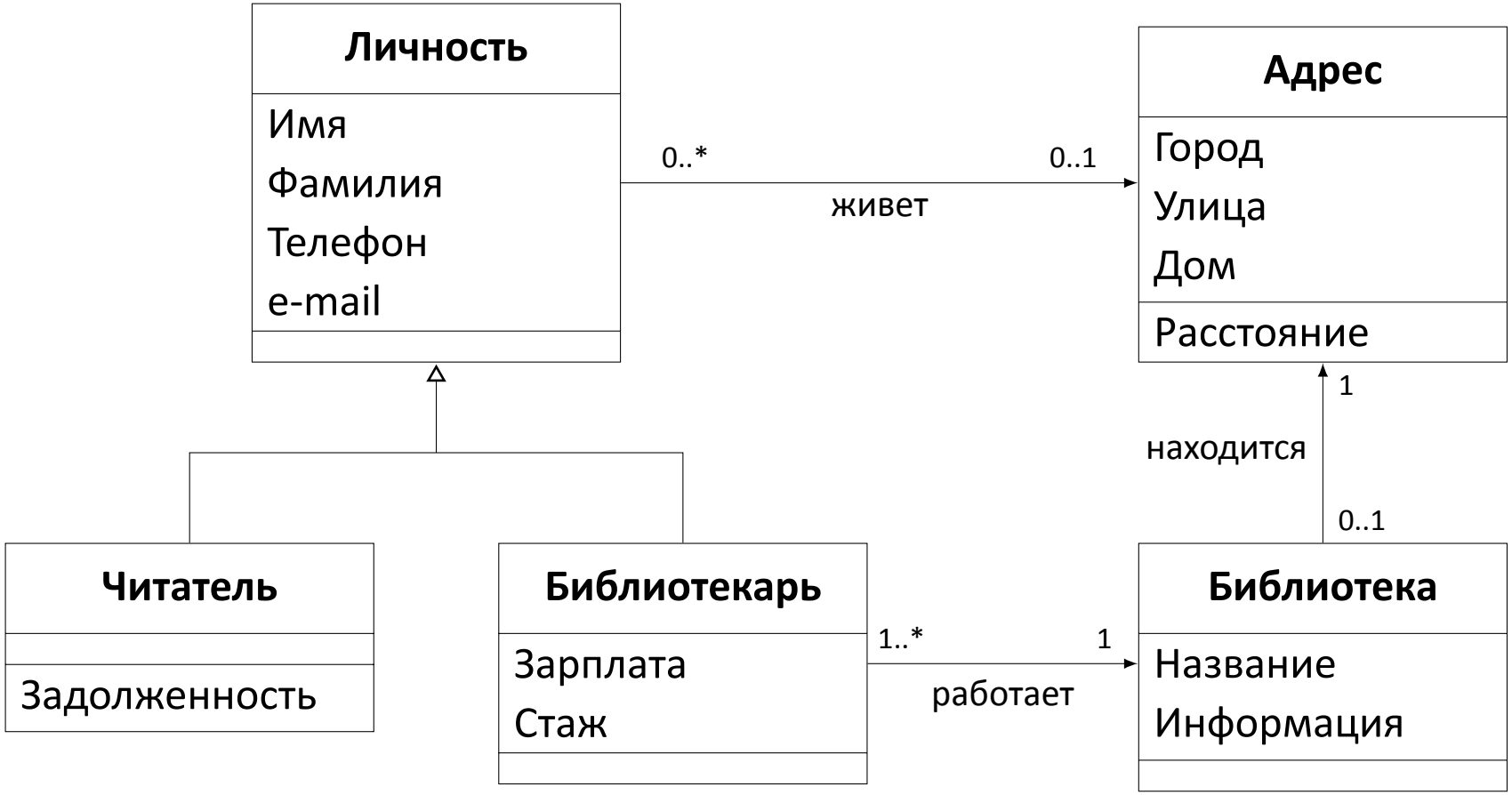


Диаграмма классов, демонстрирующая отношения обобщения (англ. *generalization*).

Отношения обобщения

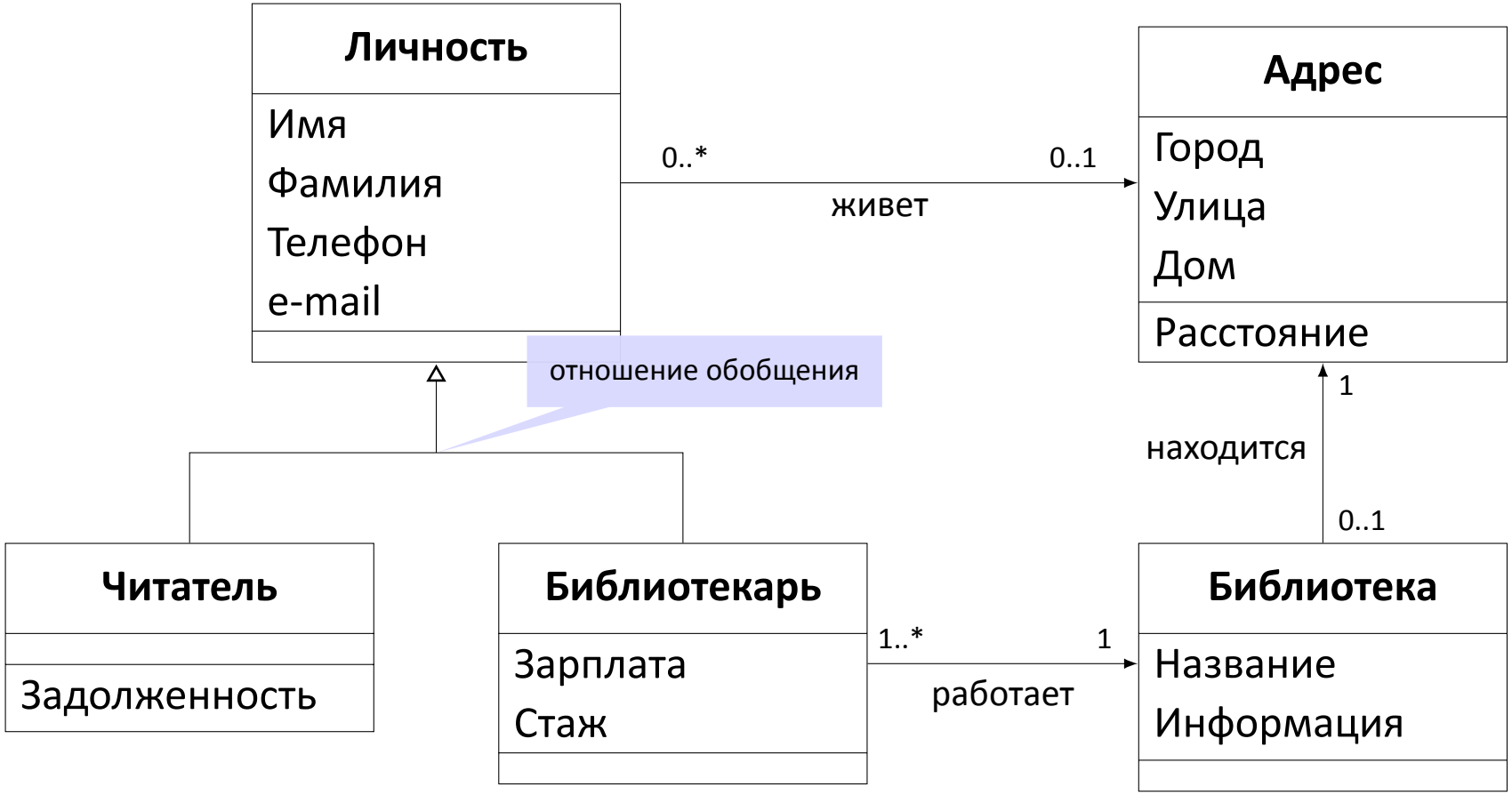


Диаграмма классов, демонстрирующая отношения обобщения (англ. *generalization*).

Отношения обобщения

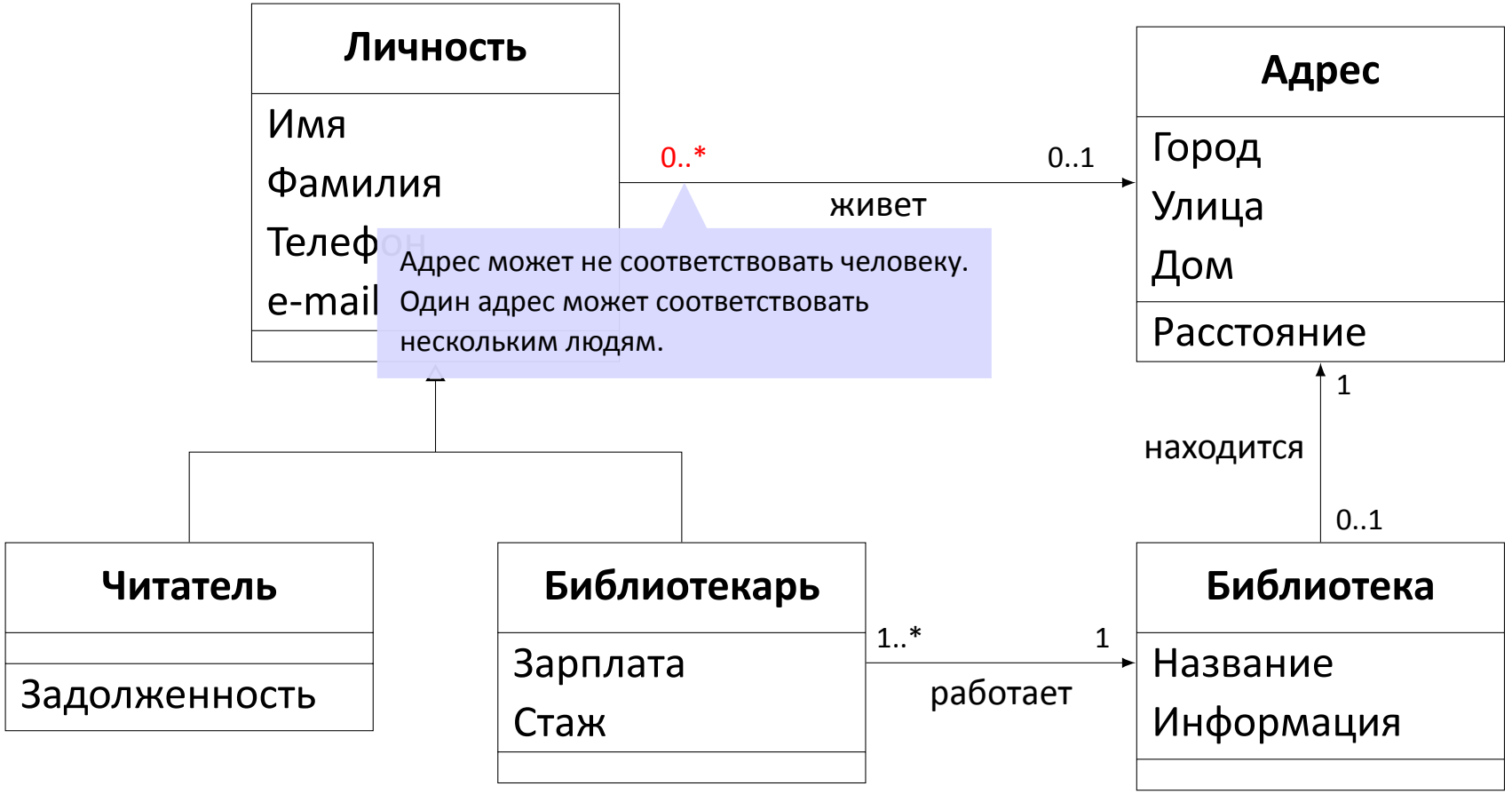


Диаграмма классов, демонстрирующая отношения обобщения (англ. *generalization*).

Отношения обобщения

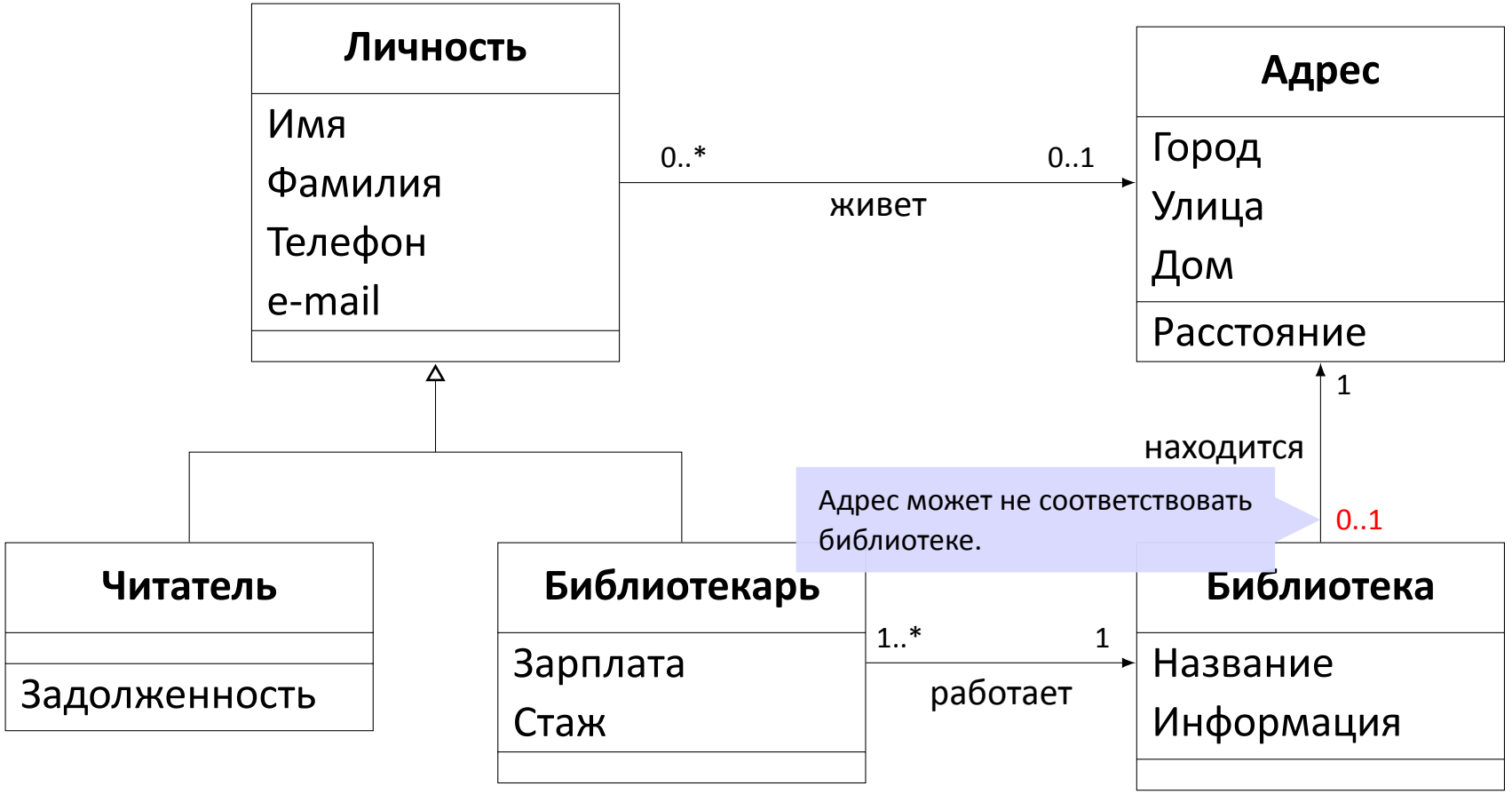


Диаграмма классов, демонстрирующая отношения обобщения (англ. *generalization*).

Отношения обобщения

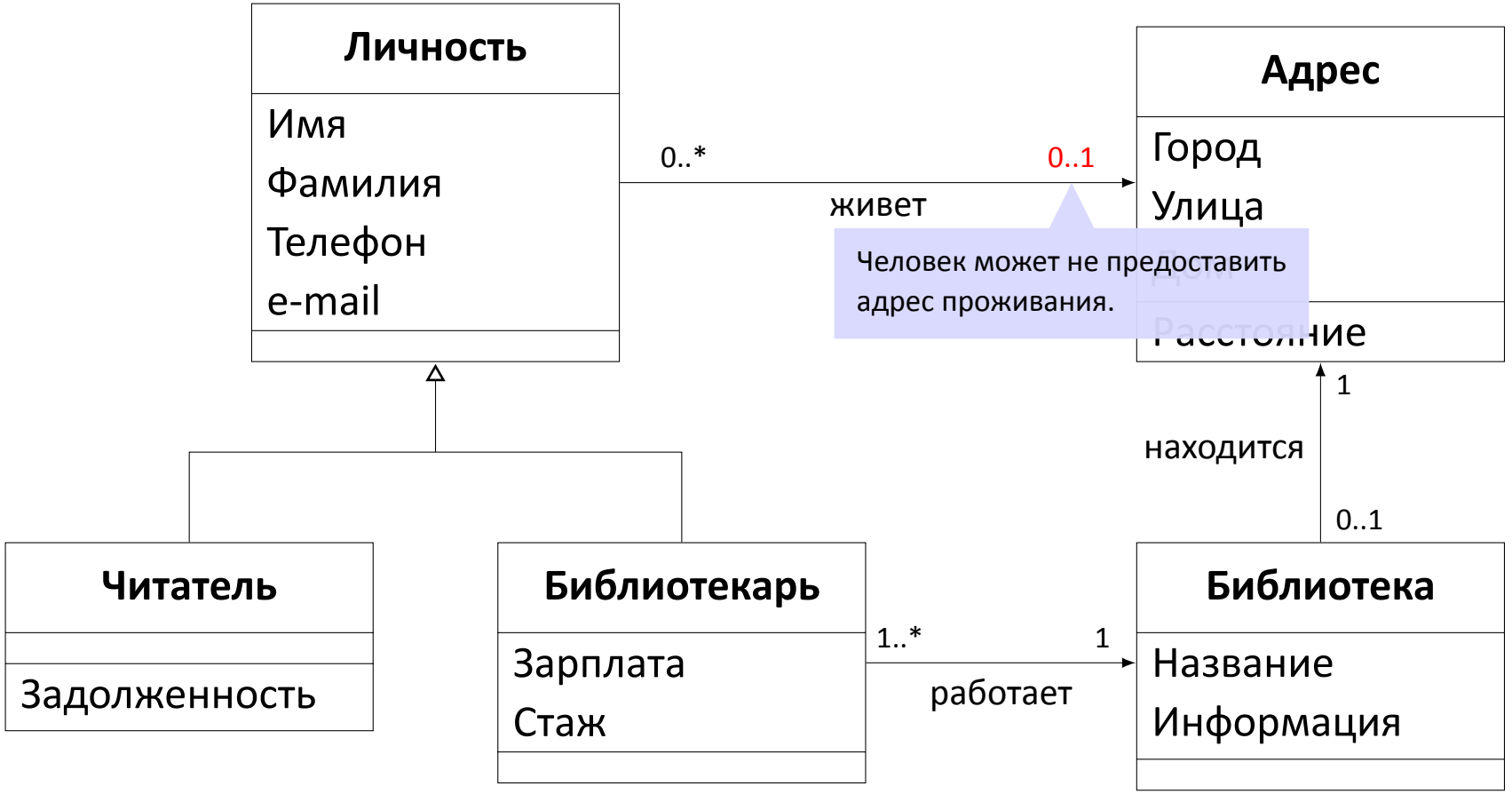
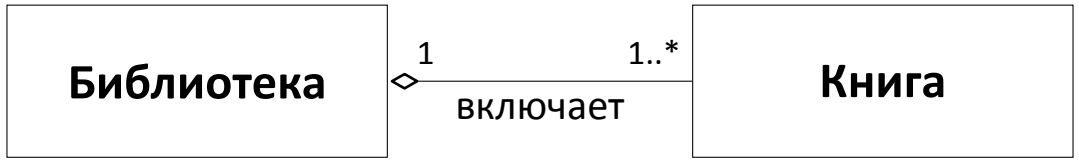


Диаграмма классов, демонстрирующая отношения обобщения (англ. *generalization*).

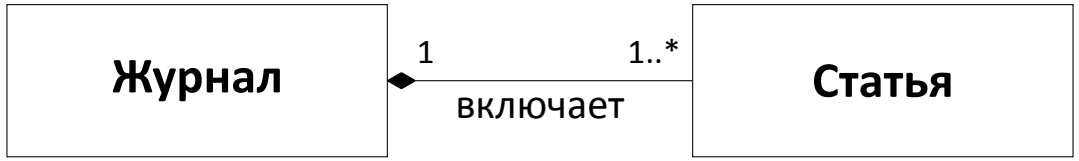
Отношения агрегации и композиции

Агрегация («часть/целое», слабая связь)



Каждая книга принадлежит одной из библиотек. Эта библиотека может измениться.

Композиция («часть/целое», сильная связь)



Журнал состоит из нескольких статей; для каждой статьи содержащий ее журнал фиксирован.

Модели поведения

Цель: определение реакции системы на внешние и внутренние входные сигналы.

Виды сигналов:

- данные (*data-driven modeling*)

  Область применения: системы обработки данных (напр., транзакций).

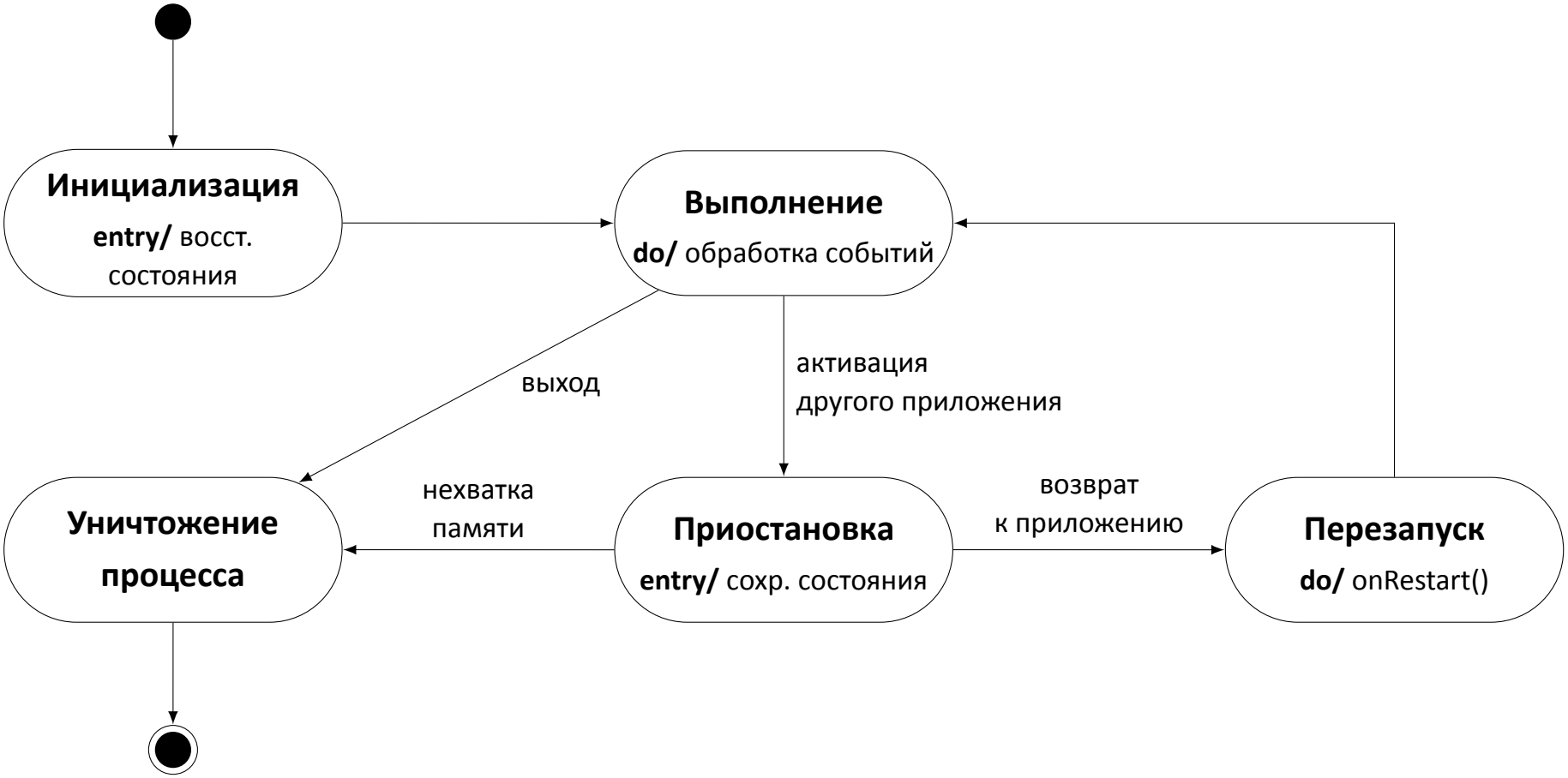
  Диаграммы UML: диаграмма деятельности, диаграмма последовательности.

- события (*event-driven modeling*)

  Область применения: системы реального времени (напр., микроконтроллеры).

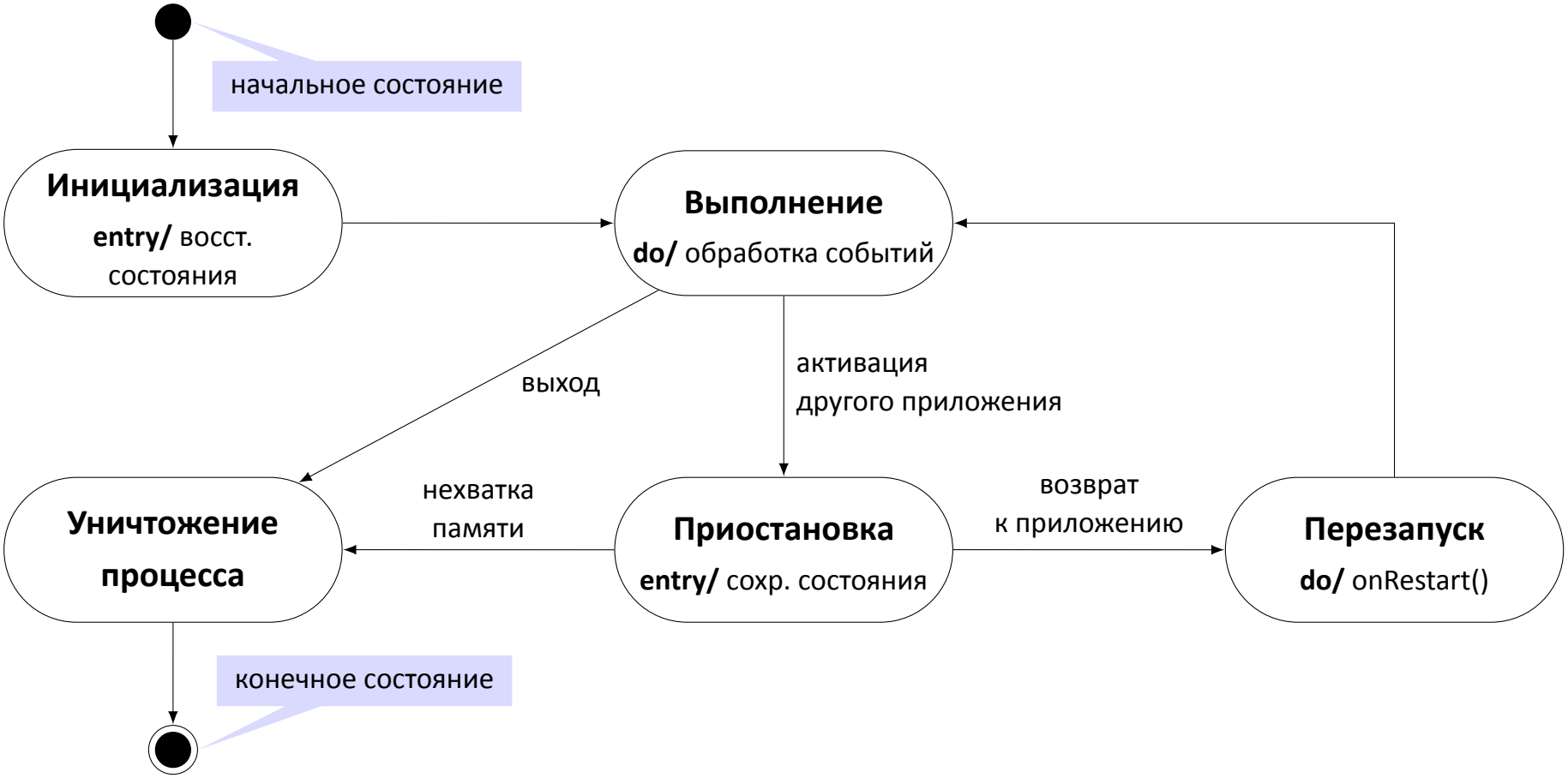
  Диаграммы UML: диаграмма состояний.

Диаграммы состояний



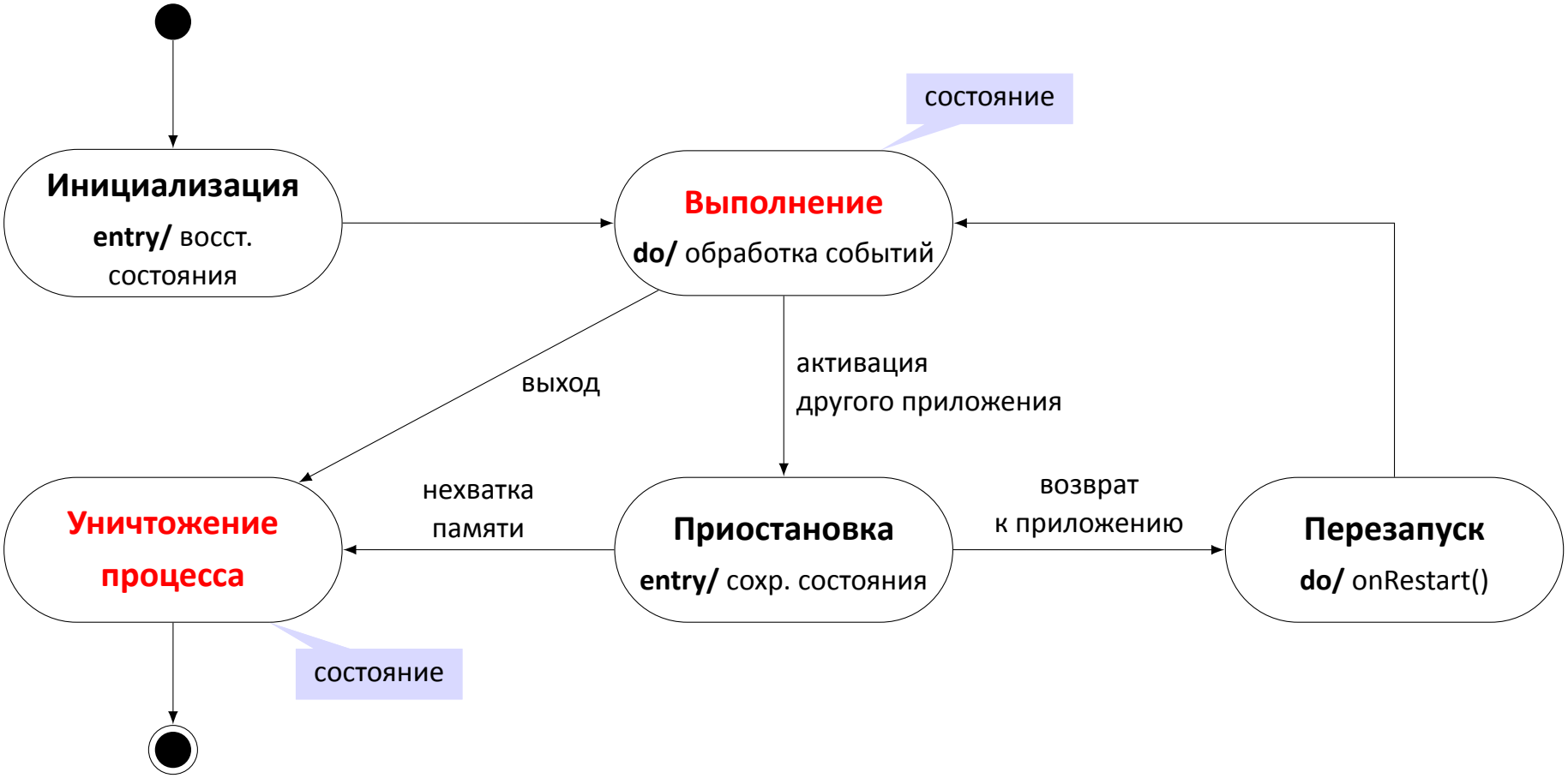
Упрощенная диаграмма состояний для жизненного цикла программы (*activity*) в Android

Диаграммы состояний



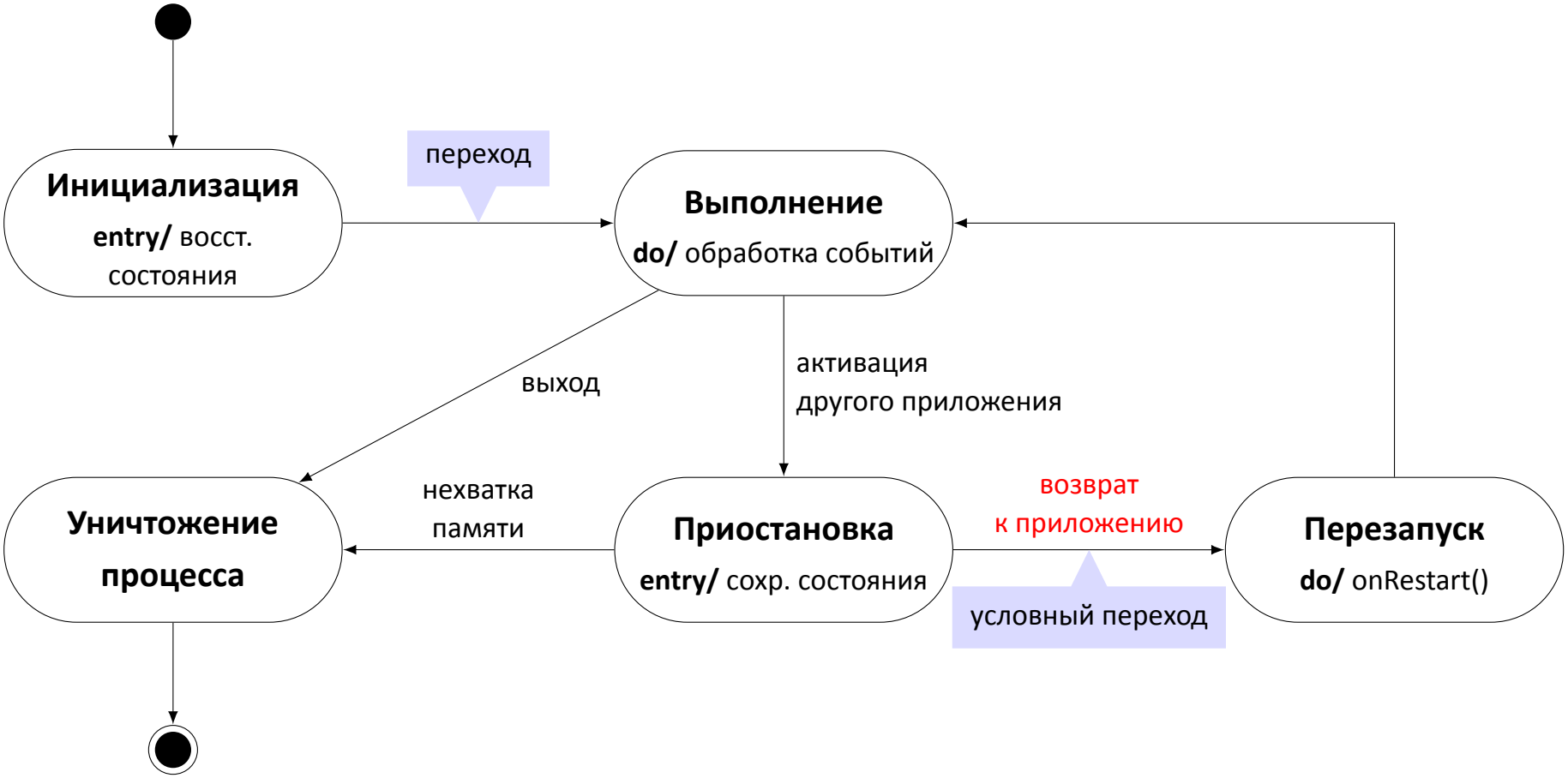
Упрощенная диаграмма состояний для жизненного цикла программы (*activity*) в Android

Диаграммы состояний



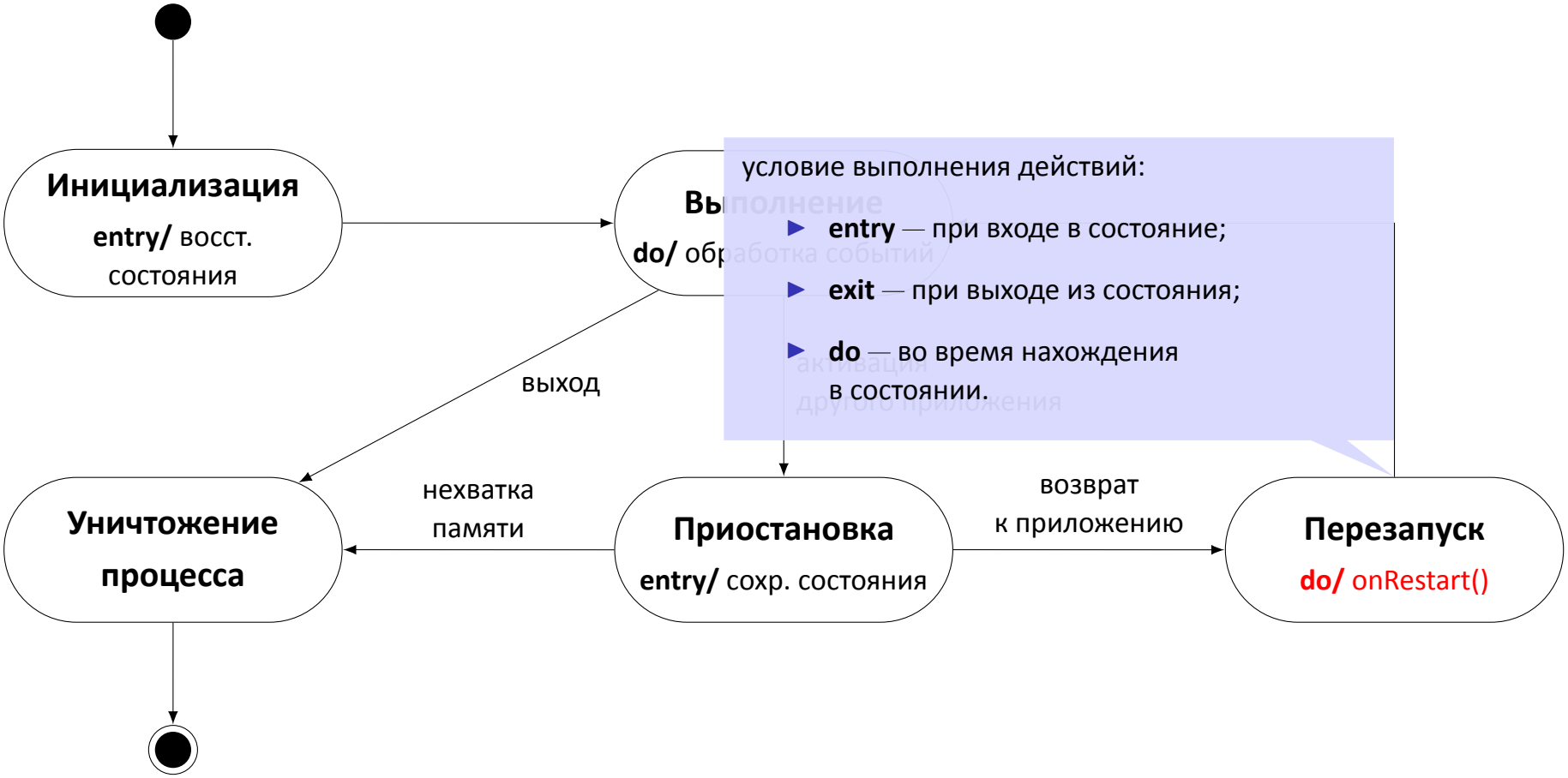
Упрощенная диаграмма состояний для жизненного цикла программы (*activity*) в Android

Диаграммы состояний



Упрощенная диаграмма состояний для жизненного цикла программы (*activity*) в Android

Диаграммы состояний



Упрощенная диаграмма состояний для жизненного цикла программы (*activity*) в Android

Выводы

1. Модели нужны для разработки отдельных аспектов программных систем: контекста выполнения, взаимодействий, структуры и поведения системы.
2. Моделирование важно для детализации требований к программному обеспечению, а также для проектирования общей архитектуры системы и отдельных элементов.
3. Одним из стандартов моделирования являются графические модели на основе языка UML. В прикладном моделировании используются 5 основных типов диаграмм UML: диаграммы деятельности, последовательности, вариантов использования, классов и состояний.

Материалы

- Sommerville, Ian

  Software Engineering.

  Pearson, 2011. — 790 p.

- Лавріщева К. М.

  Програмна інженерія (підручник).

  К., 2008. — 319 с.

Спасибо за внимание!