



# **Как кэшировать информацию в Android-приложении и не стрелять себе в ногу**

Дмитрий Васильев, FunCorp



# FunCorp: цифры и факты

- Основана в 2004
- Не является стартапом и инвестпроектом
- 100+ сотрудников
- Офисы LA, Лимассол, разработка в Москве
- iFunny
- Мемы и большое комьюнити с 2011
- Входит в топ 10 развлекательных приложений в США
- 50кк установок в США, DAU 4,5кк, 370кк просмотров



# Что предстоит?

- Стратегии кэширования
- Полномасштабный рефакторинг
- Определение необходимости
- Советы по реализации
- Немного статистики



# **Кэширование в мобильных приложениях**

- **Базы данных**



# **Кэширование в мобильных приложениях**

- **Базы данных**
- **Файлы**



# **Кэширование в мобильных приложениях**

- **Базы данных**
- **Файлы**
- **Оперативная память**



# **Кэширование в мобильных приложениях**

- **Базы данных**
- **Файлы**
- **Оперативная память**
- **Preferences**



# Лоадеры раздражают





**Цель** — оптимизировать  
получение и обработку  
данных для отображения



# **Почему это важно?**

- **Кэширование ускоряет доставку контента  
пользователю**



# **Почему это важно?**

- **Кэширование ускоряет доставку контента пользователю**
- **Быстрое отображение улучшает UX**



# **Почему это важно?**

- **Кэширование ускоряет доставку контента пользователю**
- **Быстрое отображение улучшает UX**
- **Улучшенный UX увеличивает retention**



# Почему это важно?

- **Кэширование ускоряет доставку контента пользователю**
- **Быстрое отображение улучшает UX**
- **Улучшенный UX увеличивает retention**
- **С увеличением retention'a увеличивается профит**



# Размер кэша





# Слабые девайсы на развивающихся рынках





# Плохой пример №1



## Space used

App size	60.80 MB
User data	59.96 MB
Cache	341 MB
Total	462 MB



# Плохой пример №2



Space used

App size

139 MB

User data

88.32 MB

Cache

450 MB

Total

677 MB



# Плохой пример №3



## Space used

App size

94.63 MB

User data

17.68 MB

Cache

539 MB

Total

651 MB



# Никто не идеален



## Space used

App size

46.61 MB

User data

102 MB

Cache

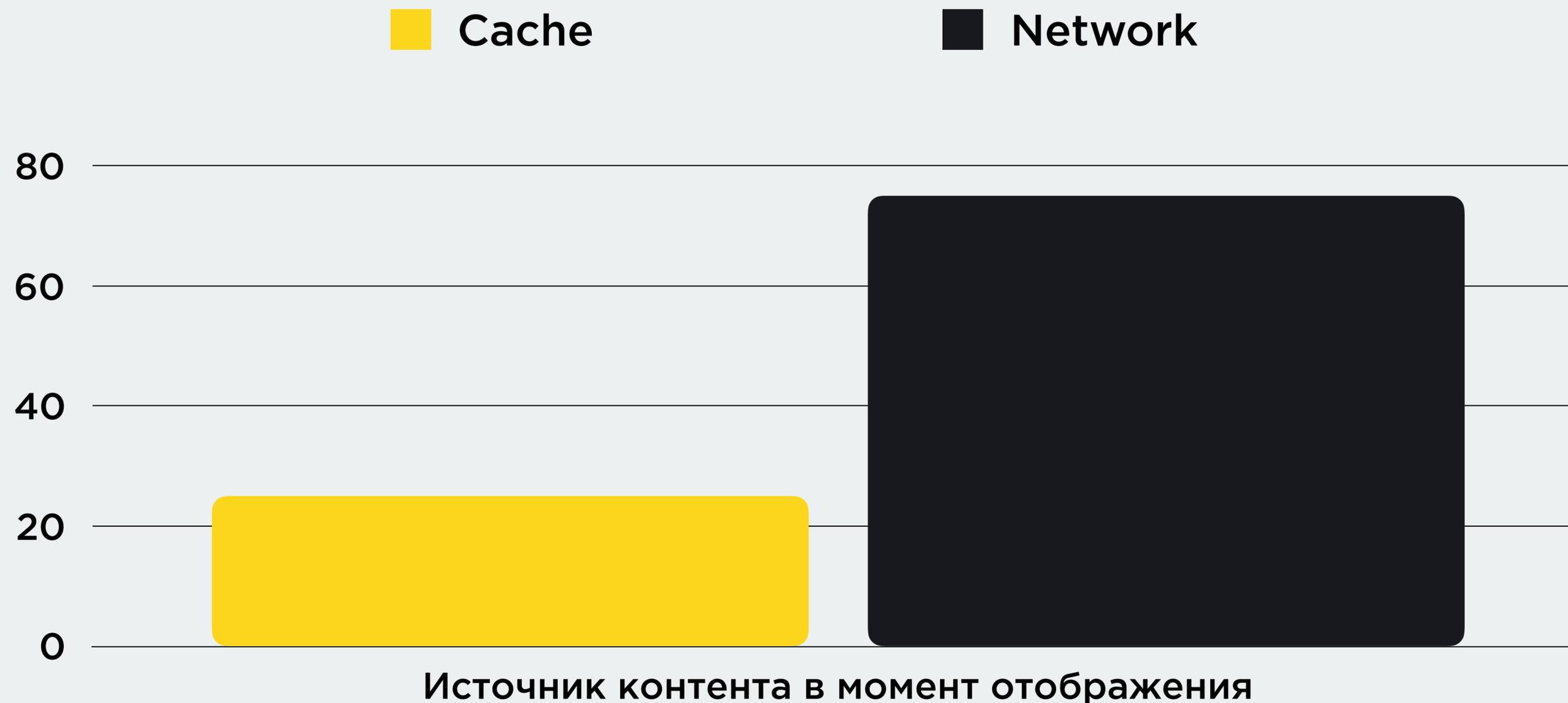
336 MB

Total

484 MB



# Статистика до реализации





# Статистика до реализации

**Progress Bar**

**1,5KK**

**Retry**

**1KK**

**Content**

**100KK**



# Рассматриваемые стратегии кэширования

- Lazy cache



# Рассматриваемые стратегии кэширования

- Lazy cache
- Synchronized cache



# Рассматриваемые стратегии кэширования

- Lazy cache
- Synchronized cache
- Write-through cache



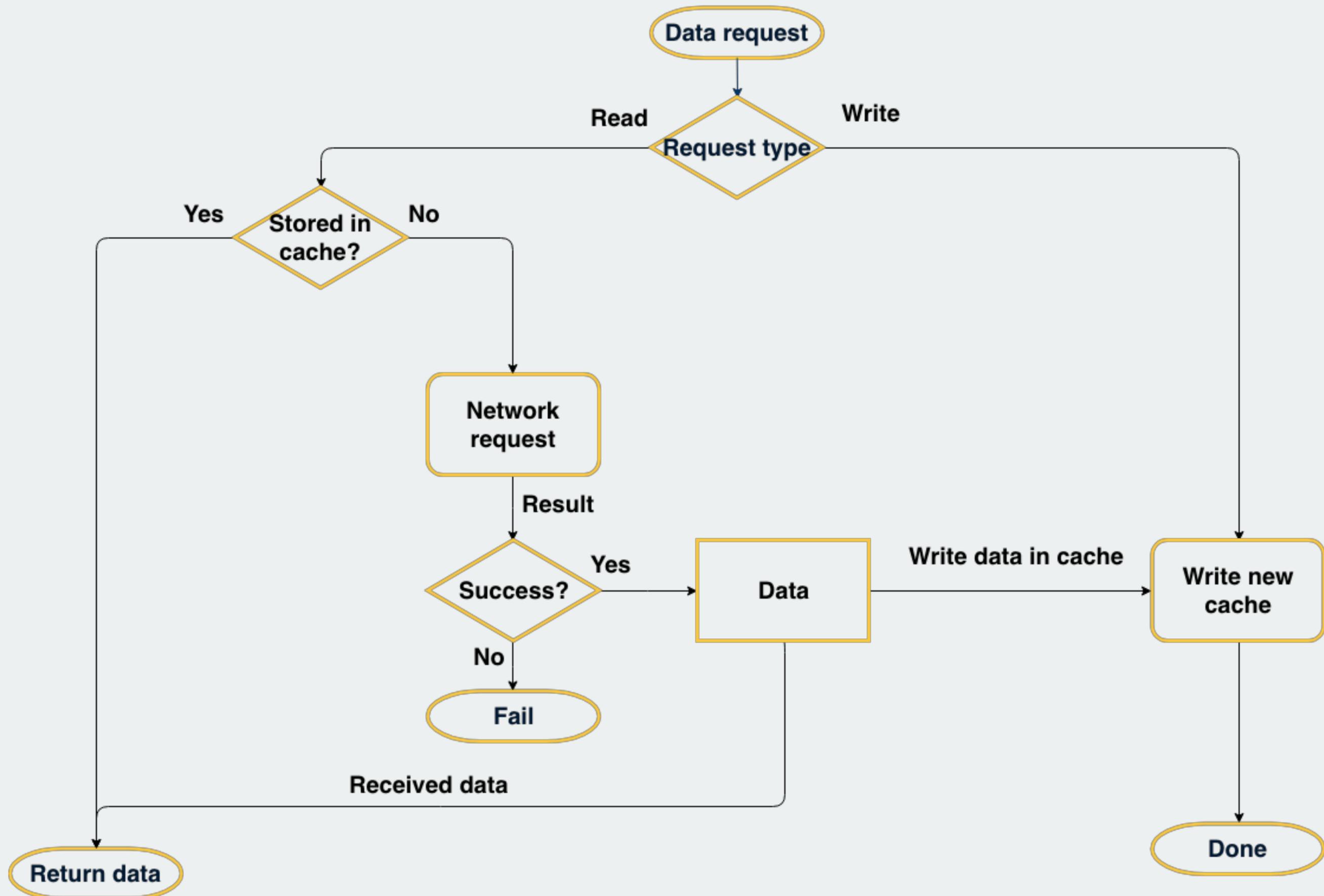
# Рассматриваемые стратегии кэширования

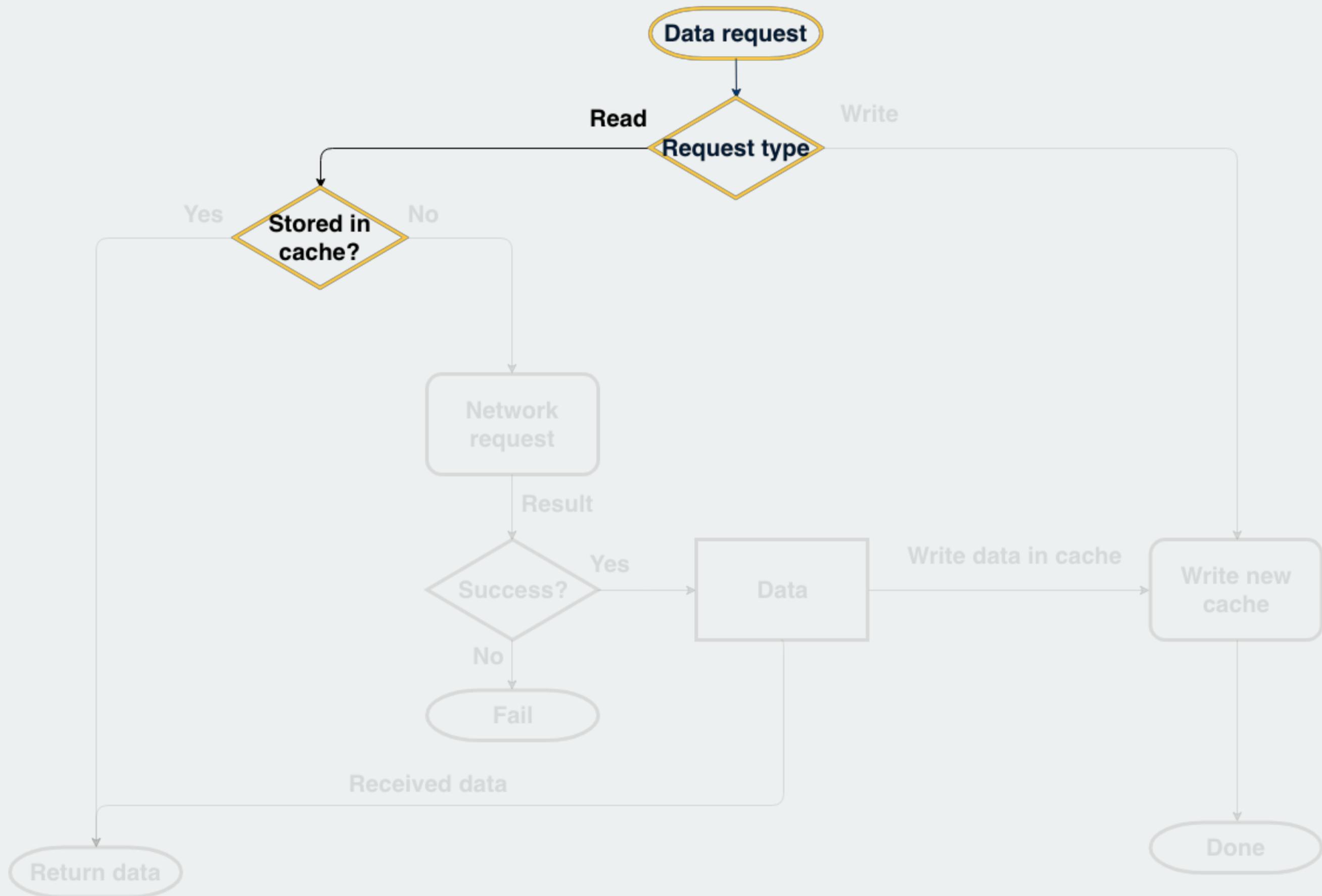
- Lazy cache
- Synchronized cache
- Write-through cache
- Least recent used (LRU) cache

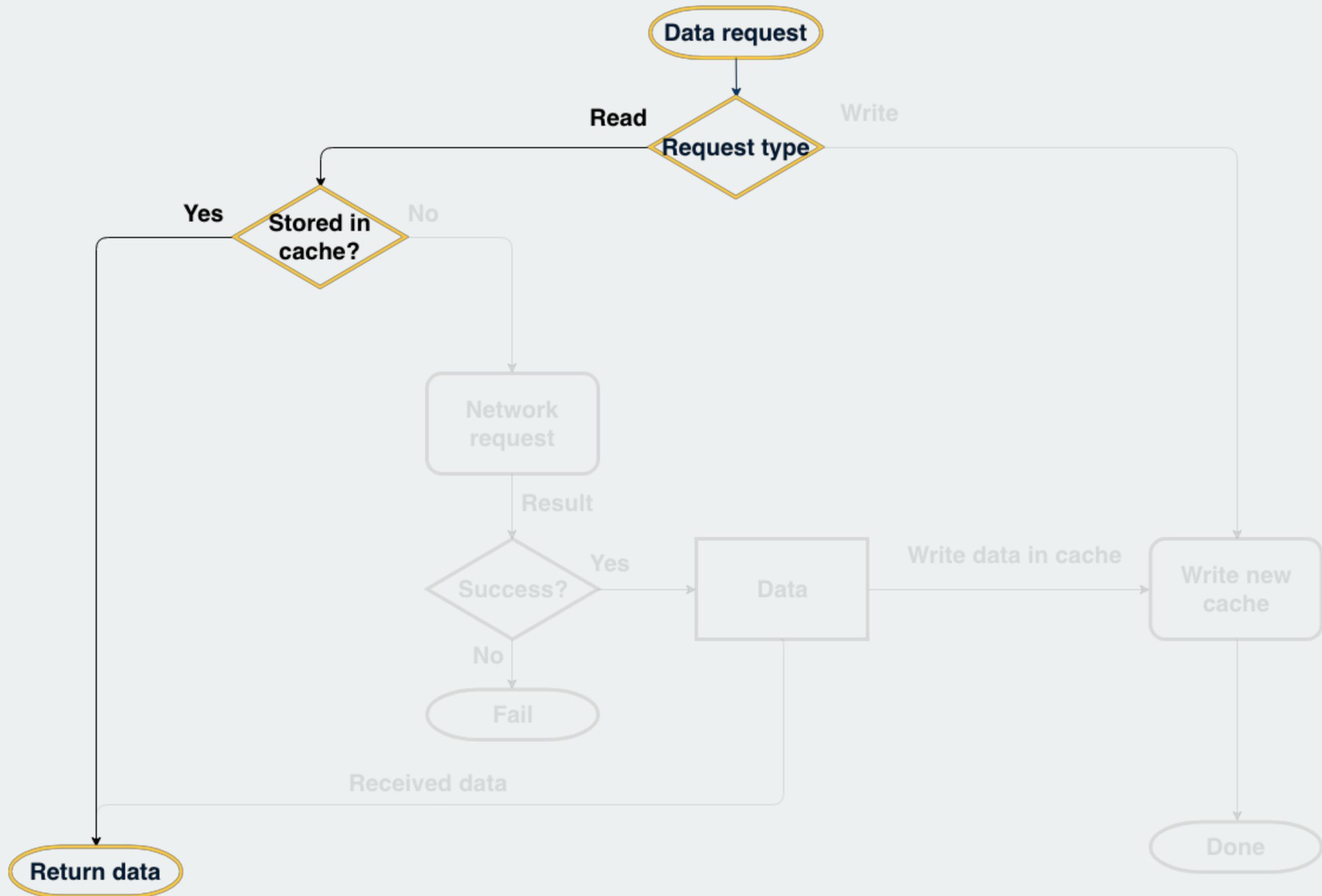


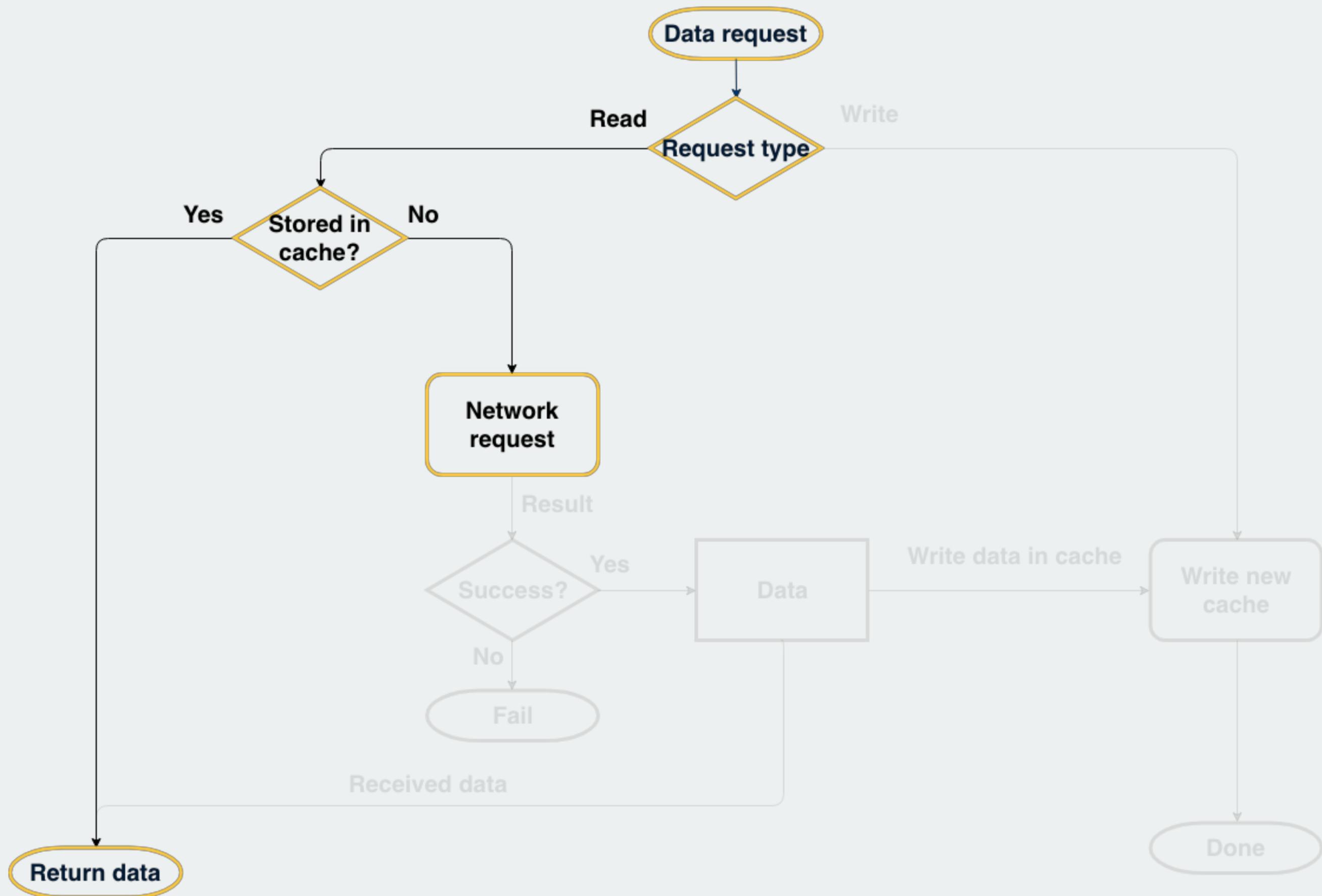
1

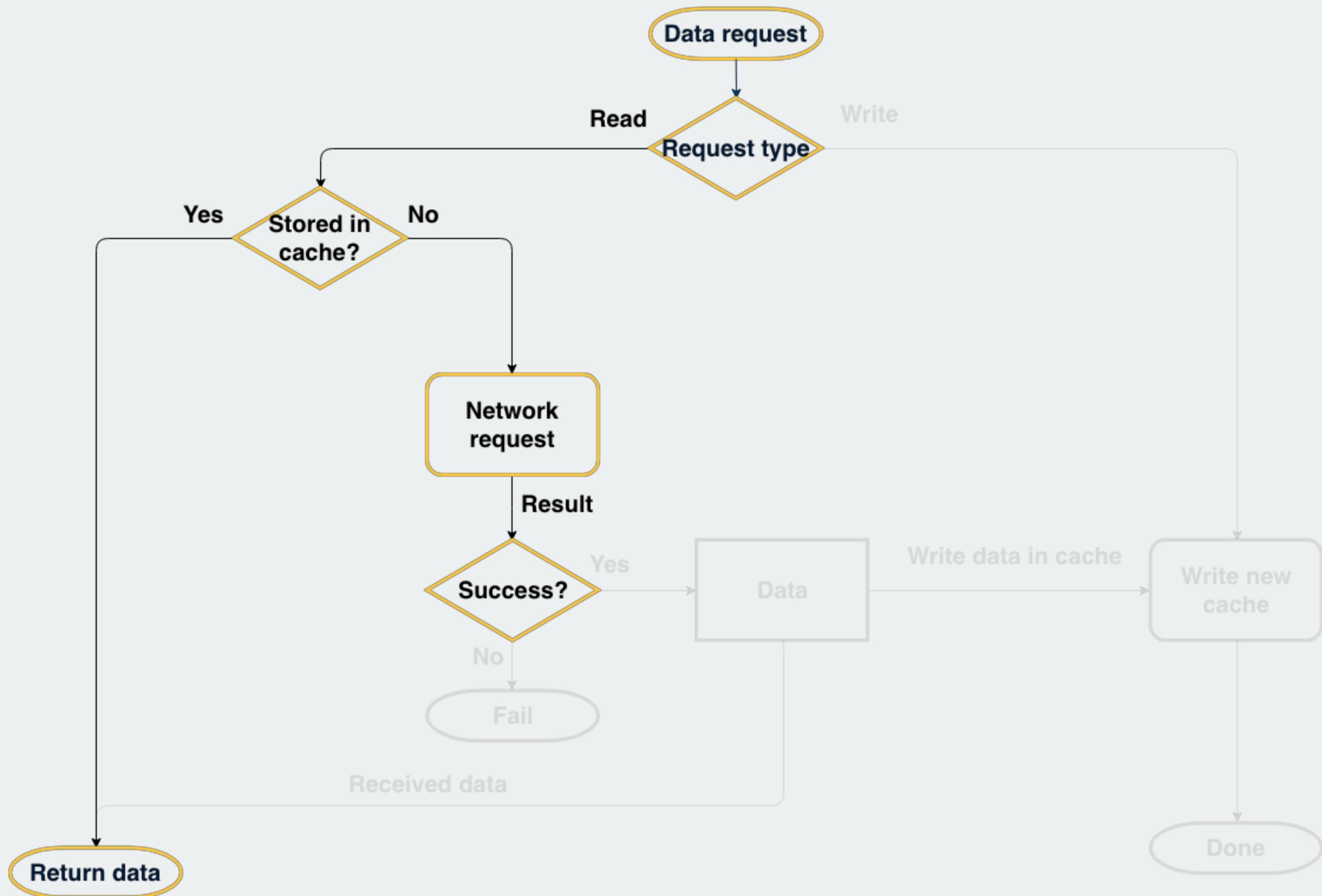
# Lazy cache

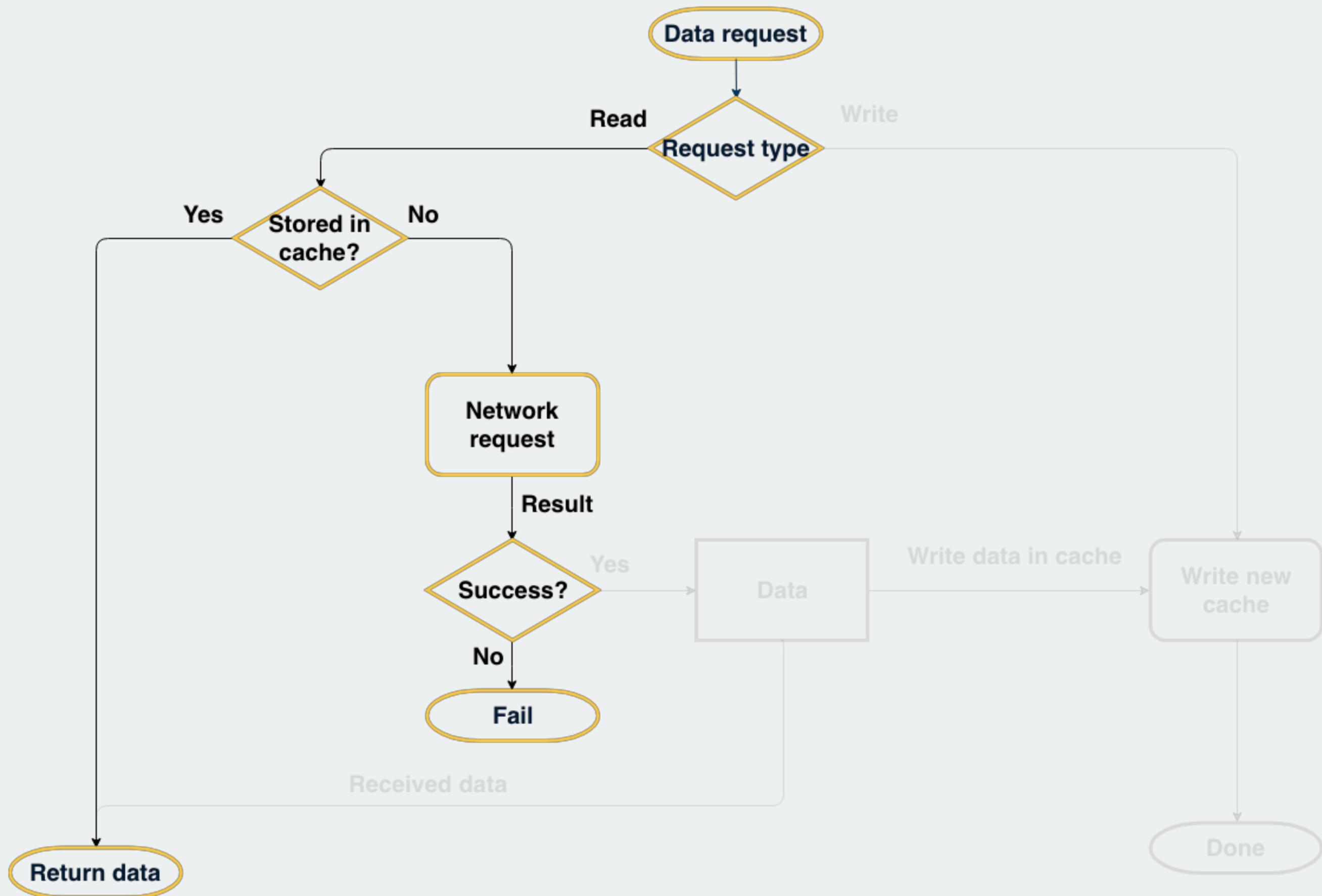


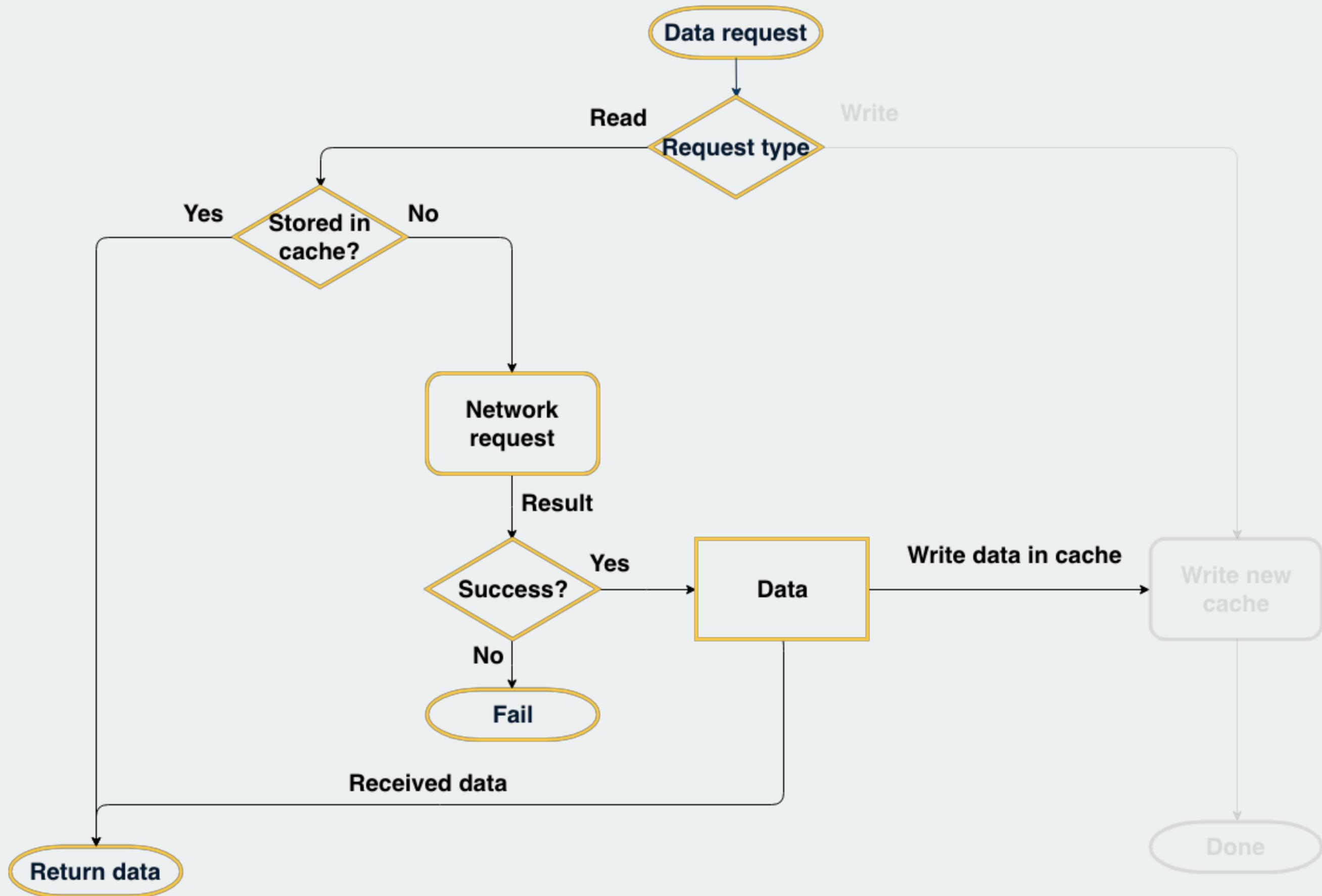


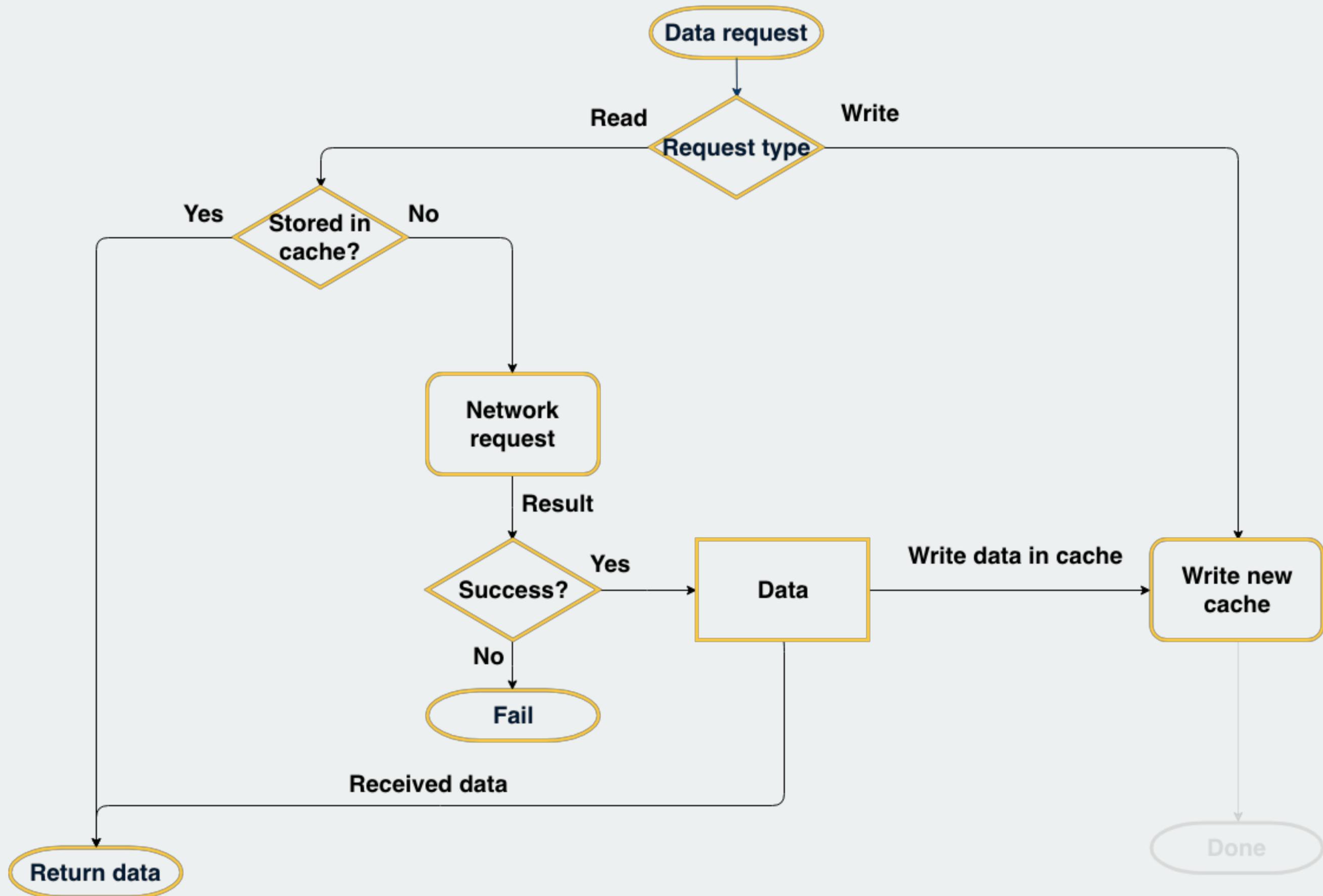


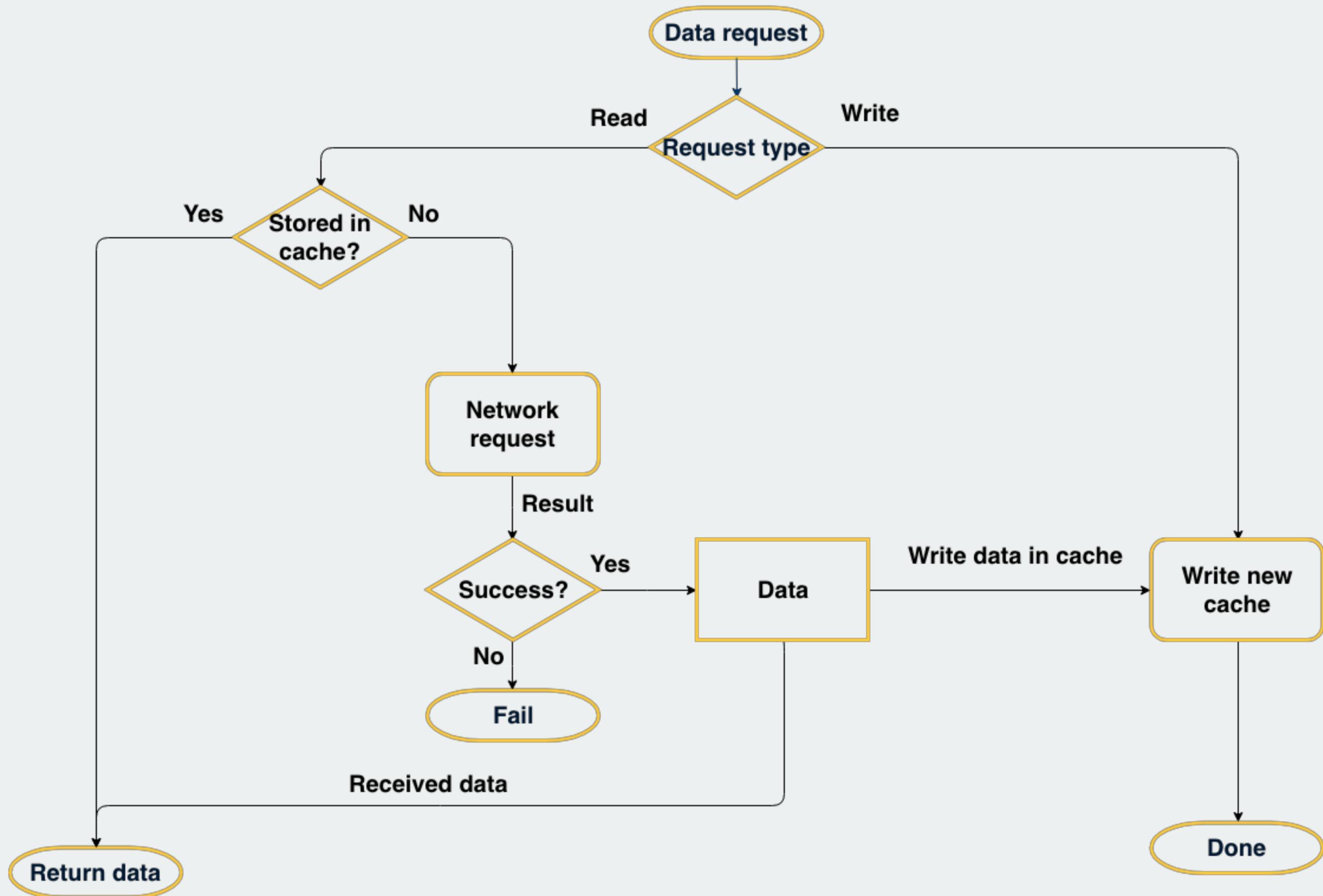














# Плюсы и минусы

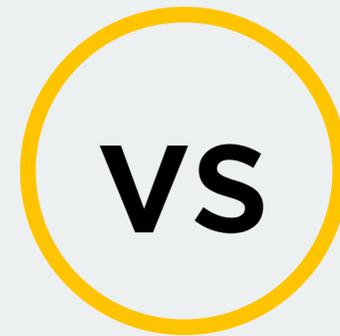
- Просто реализовать





# Плюсы и минусы

- Просто реализовать
- Легко переиспользовать





# Плюсы и минусы

- Просто реализовать
- Легко переиспользовать
- Закашированные данные не зависят от соединения





# Плюсы и минусы

- Просто реализовать
- Легко переиспользовать
- Закешированные данные не зависят от соединения
- Мгновенная доставка данных





# Плюсы и минусы

- Просто реализовать
- Легко переиспользовать
- Закешированные данные не зависят от соединения
- Мгновенная доставка данных



- Нет инвалидации в алгоритме по умолчанию



# Где использовать?

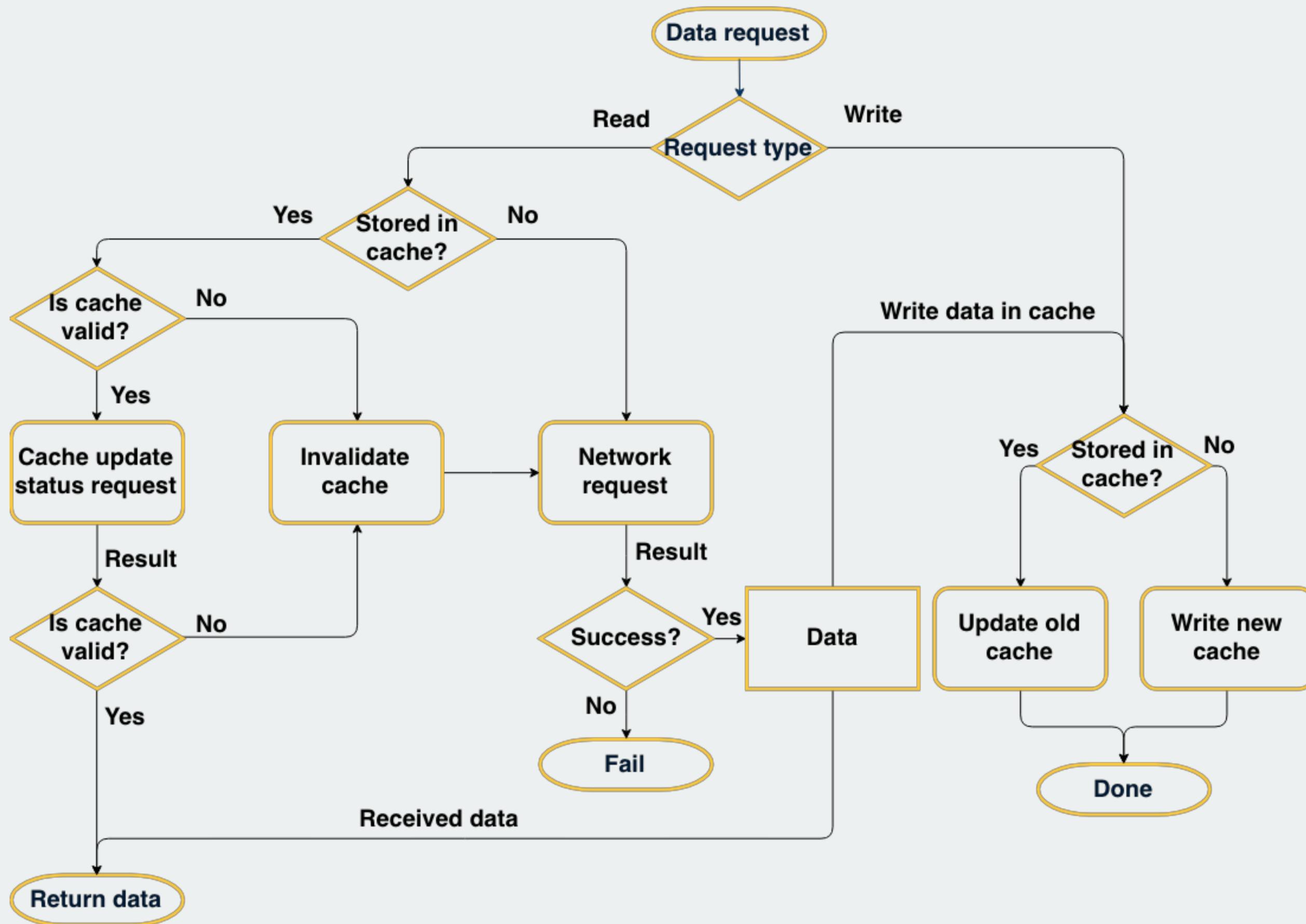
Приложения с возможностью дозагрузки неизменных данных: справочники, читалки, оффлайн приложения и тп.

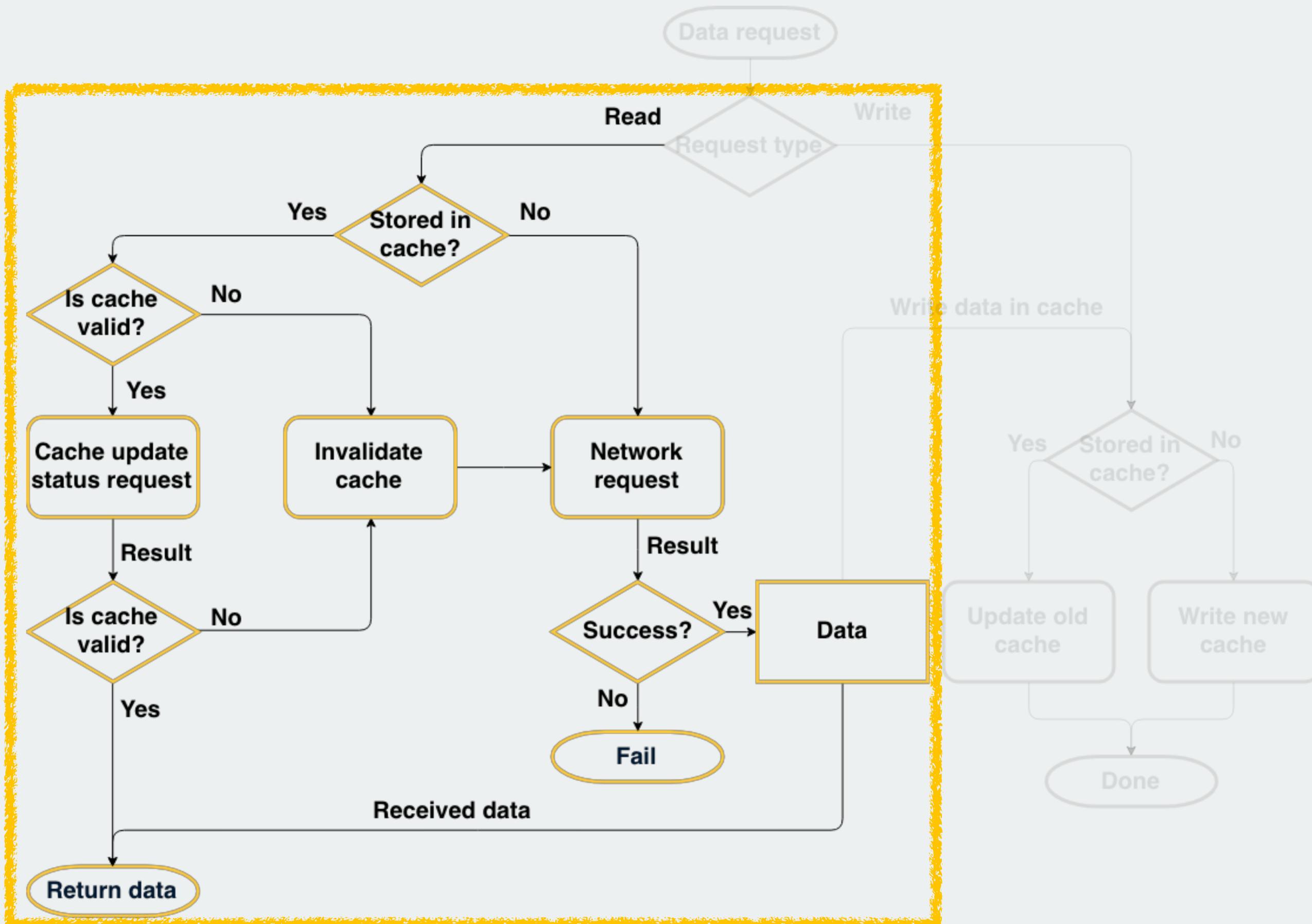


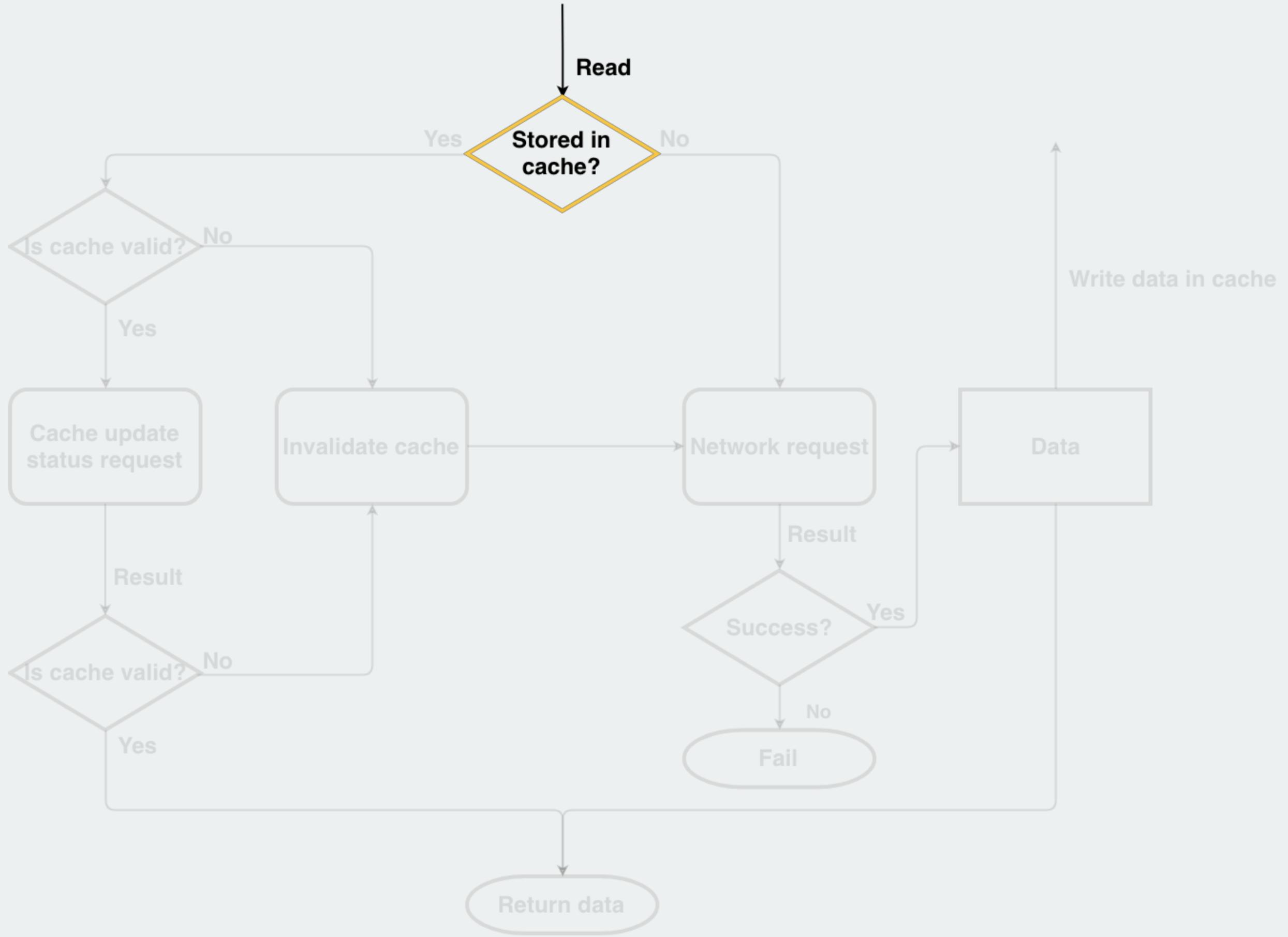


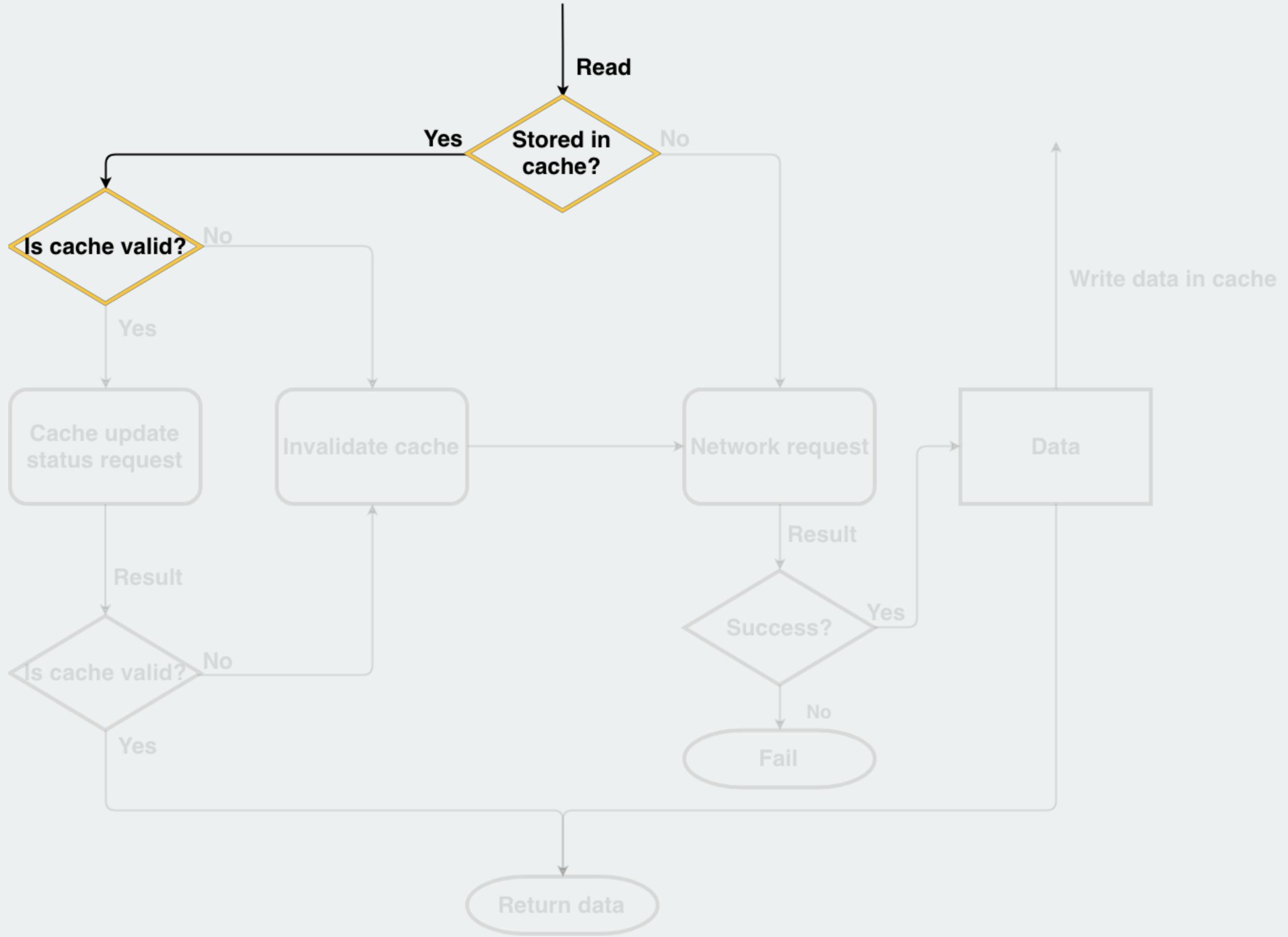
2

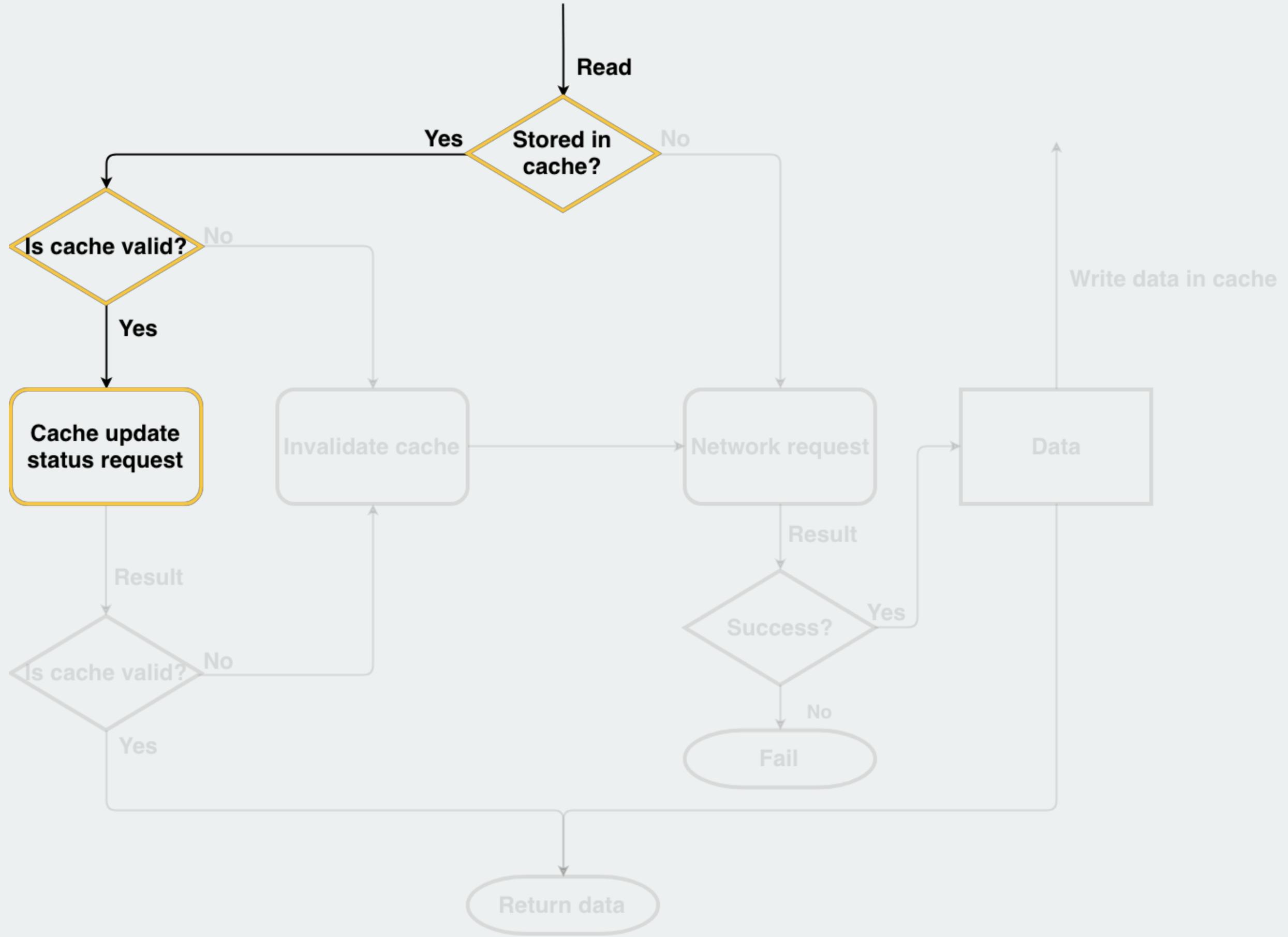
# Synchronized cache

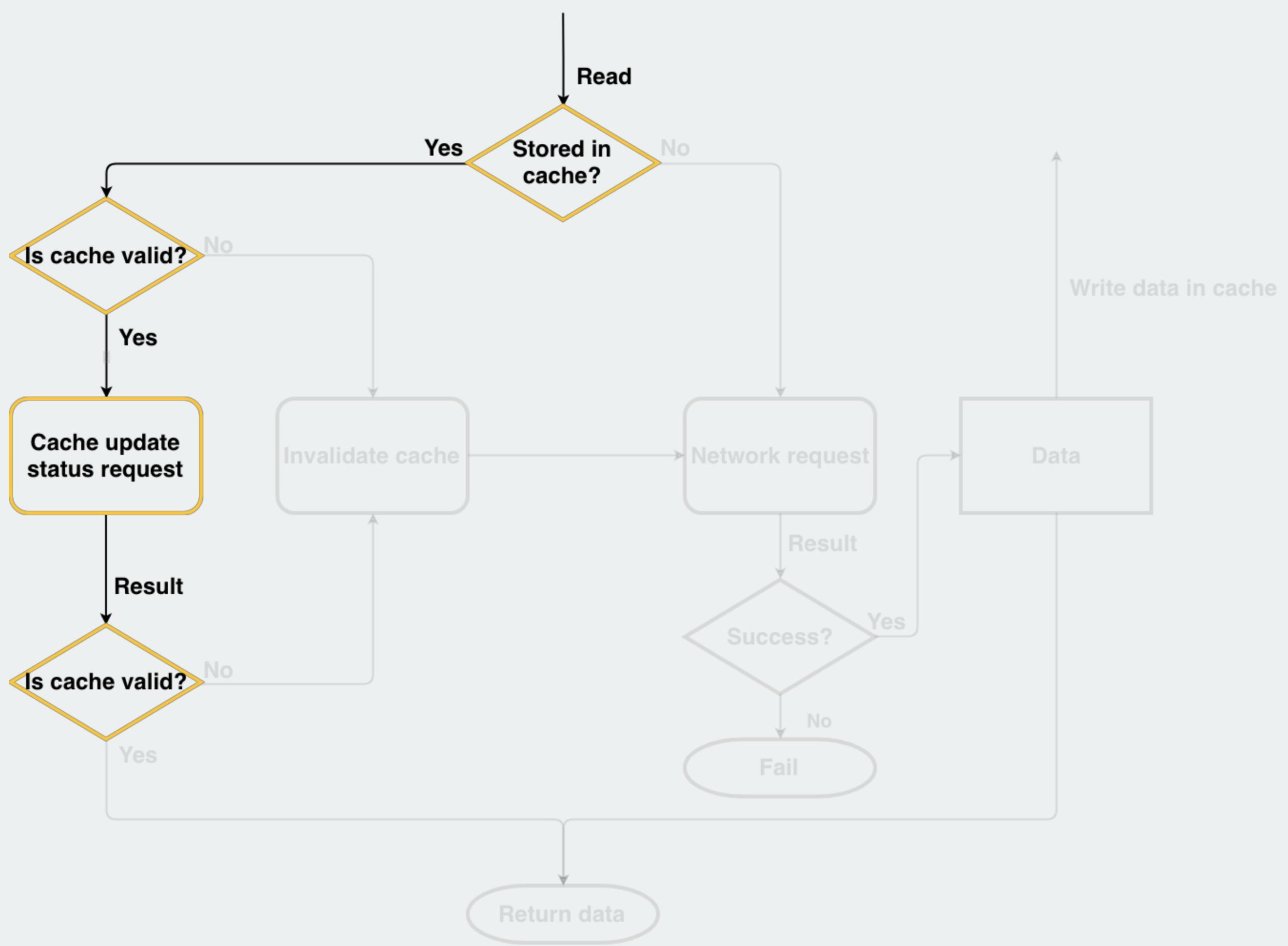


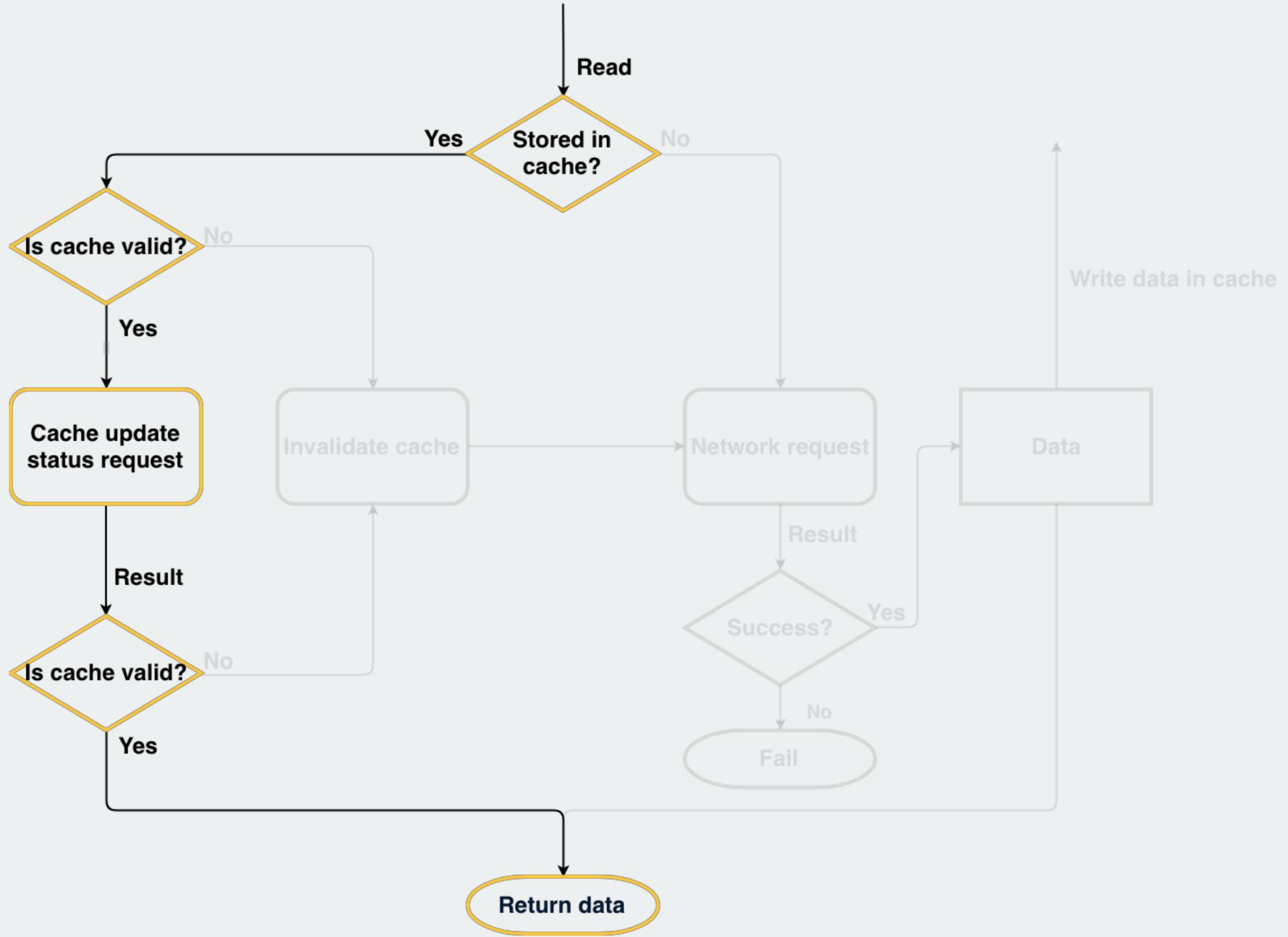


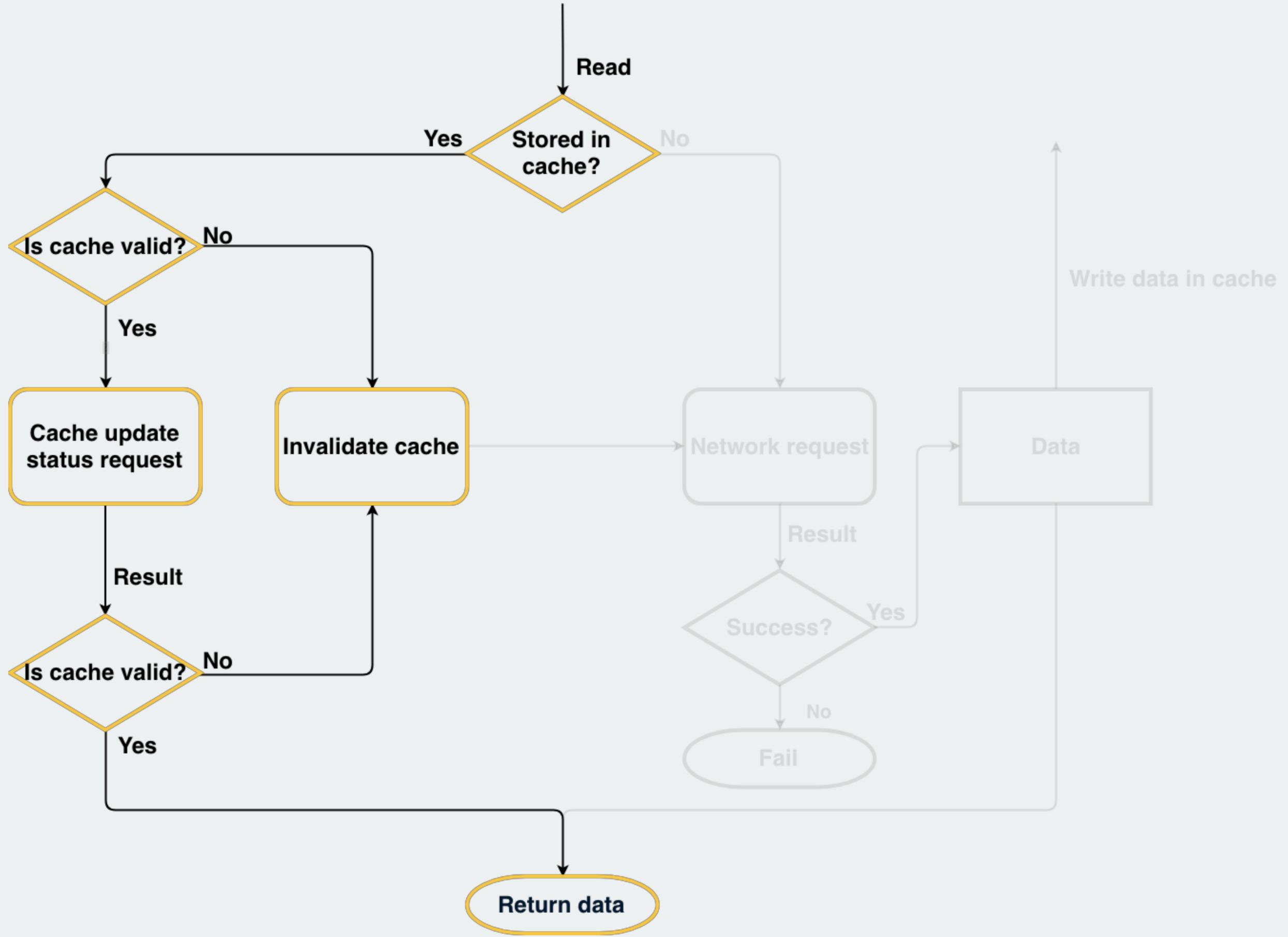


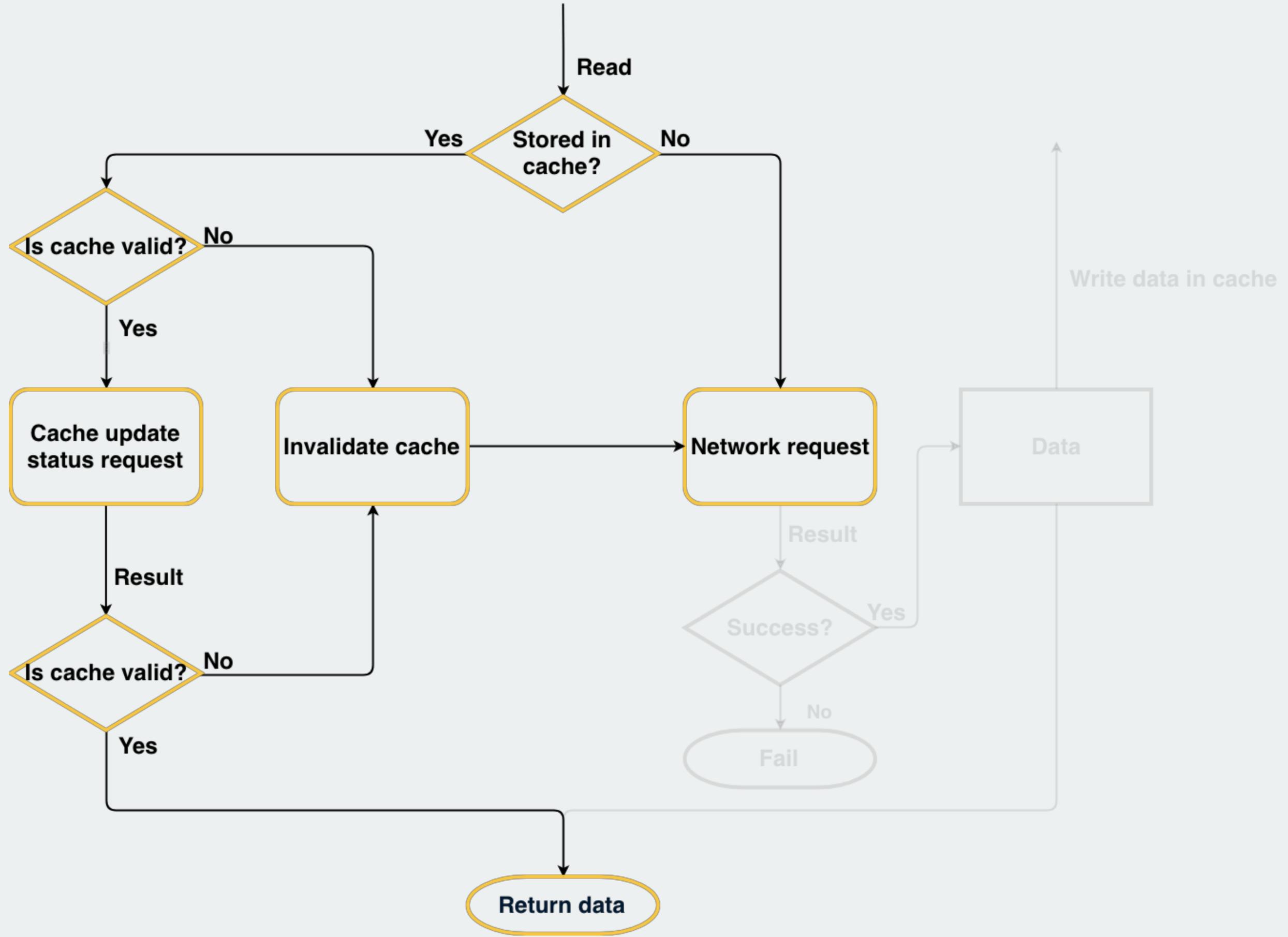


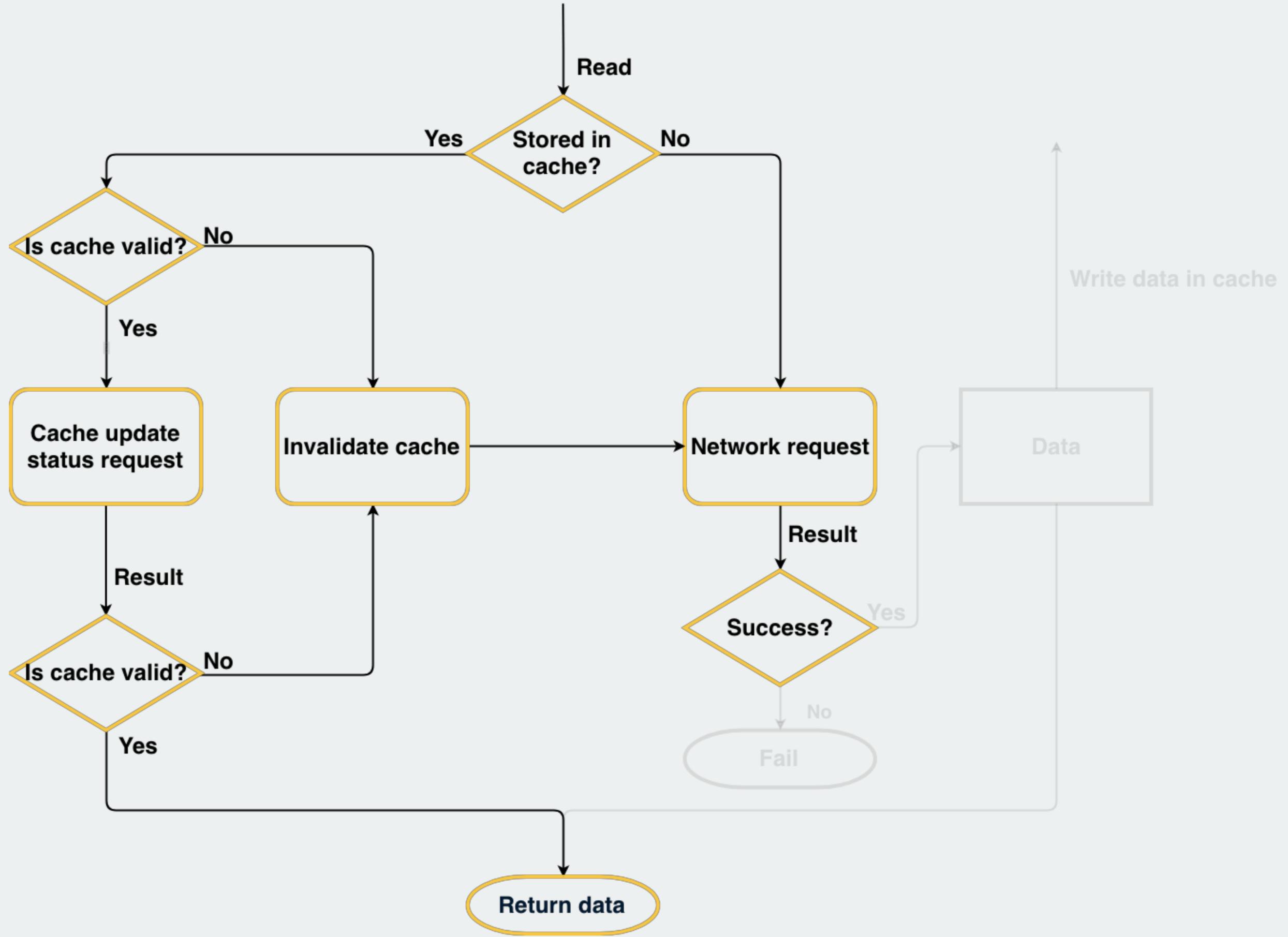


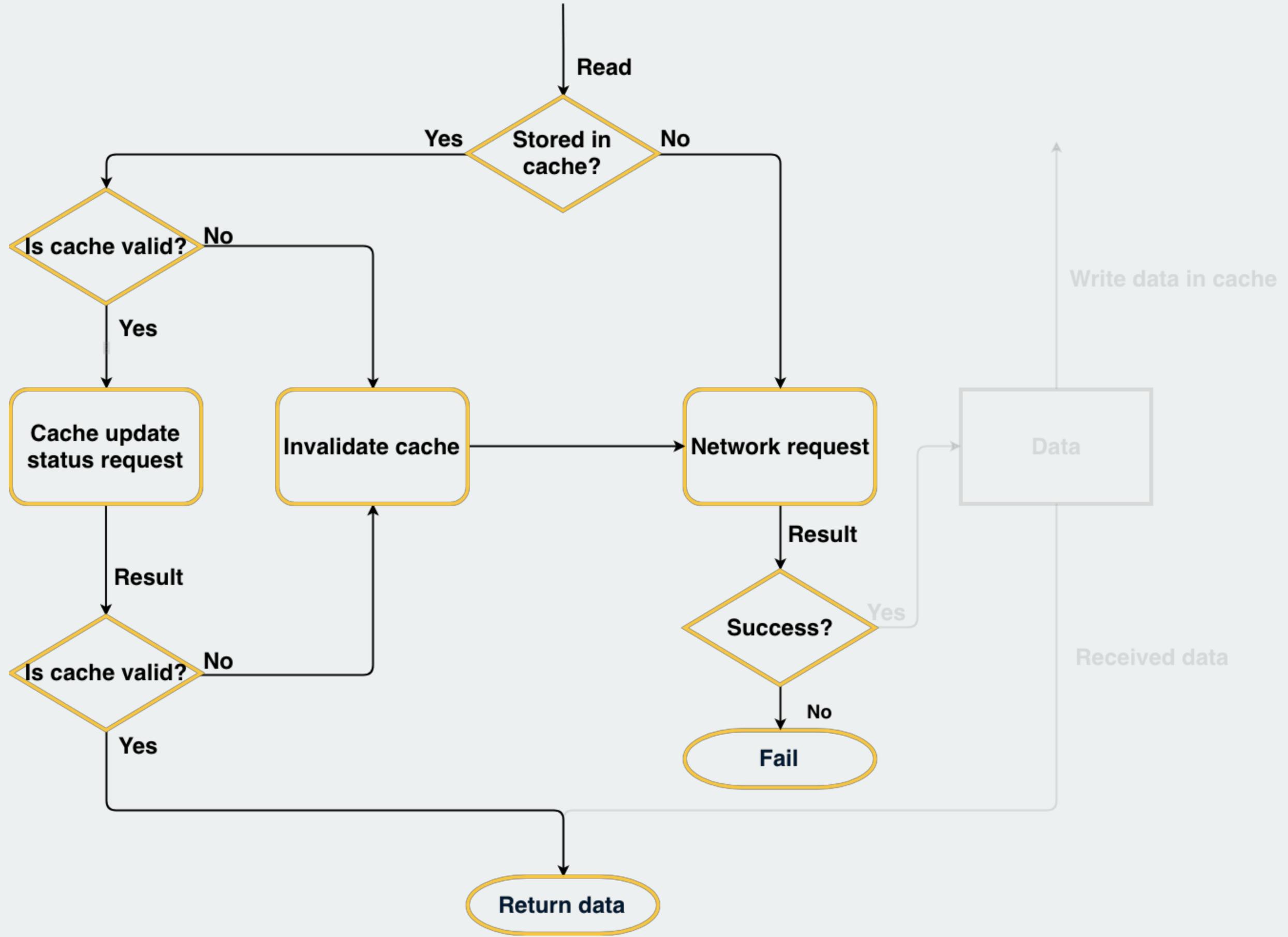


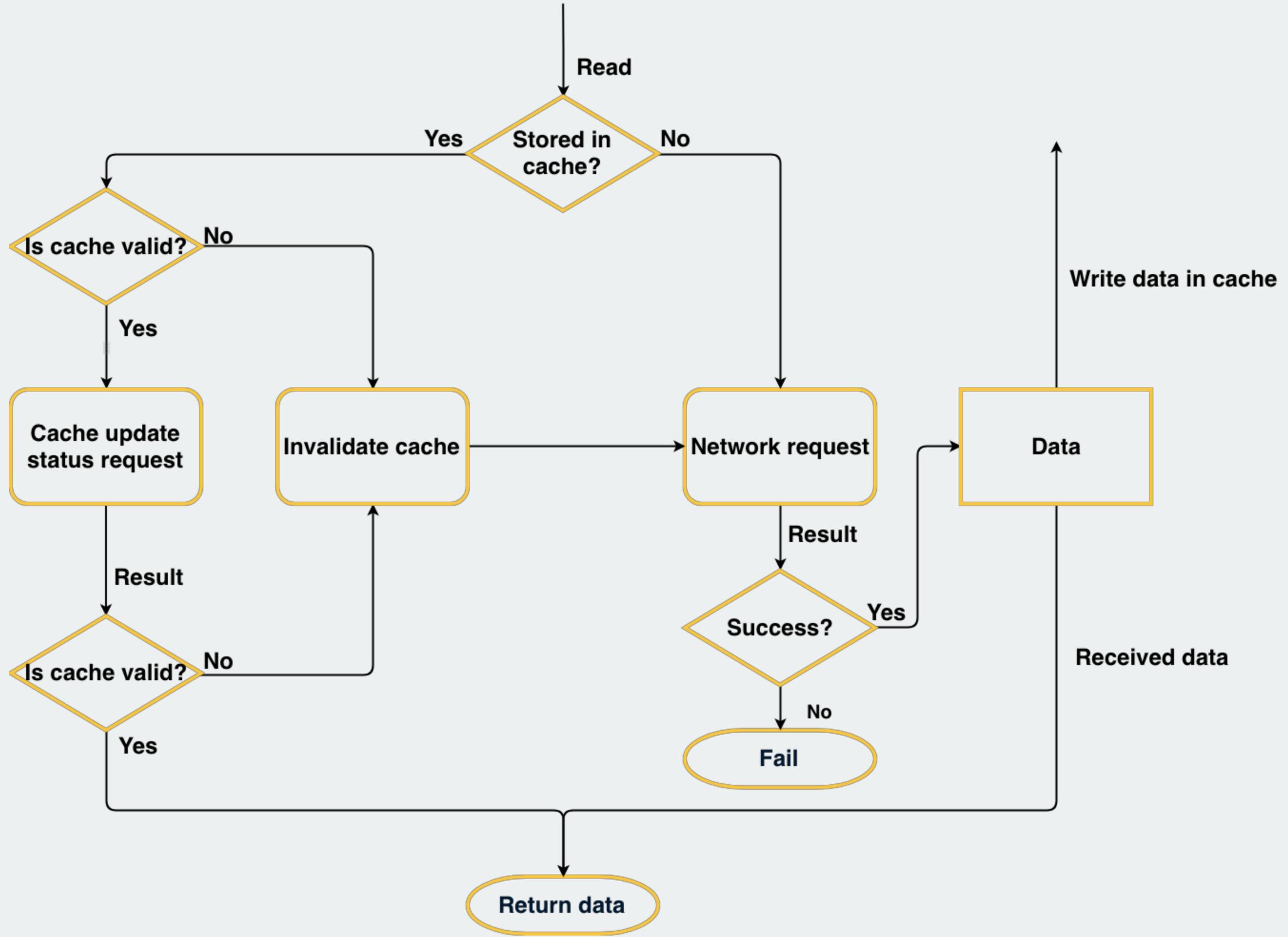


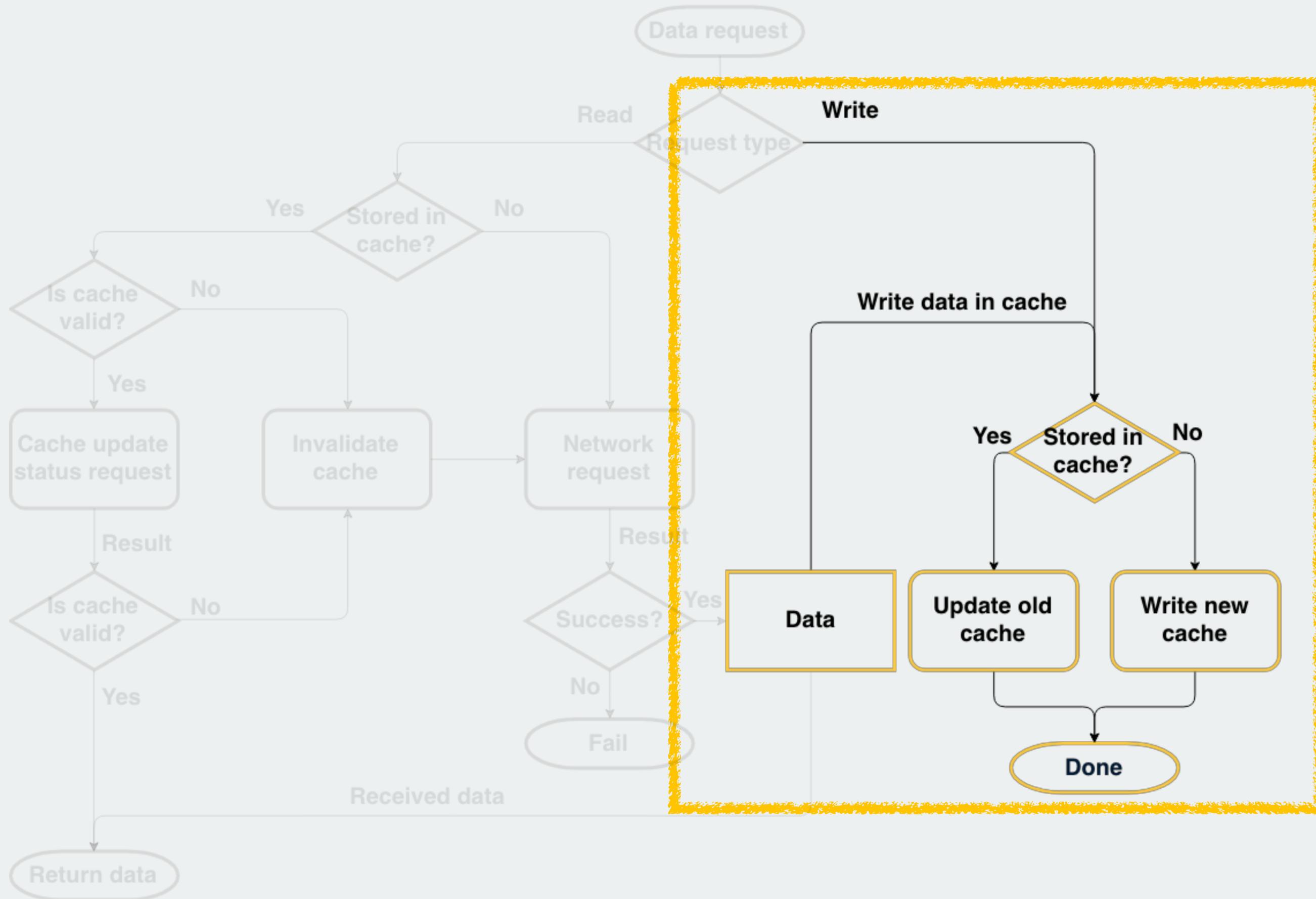


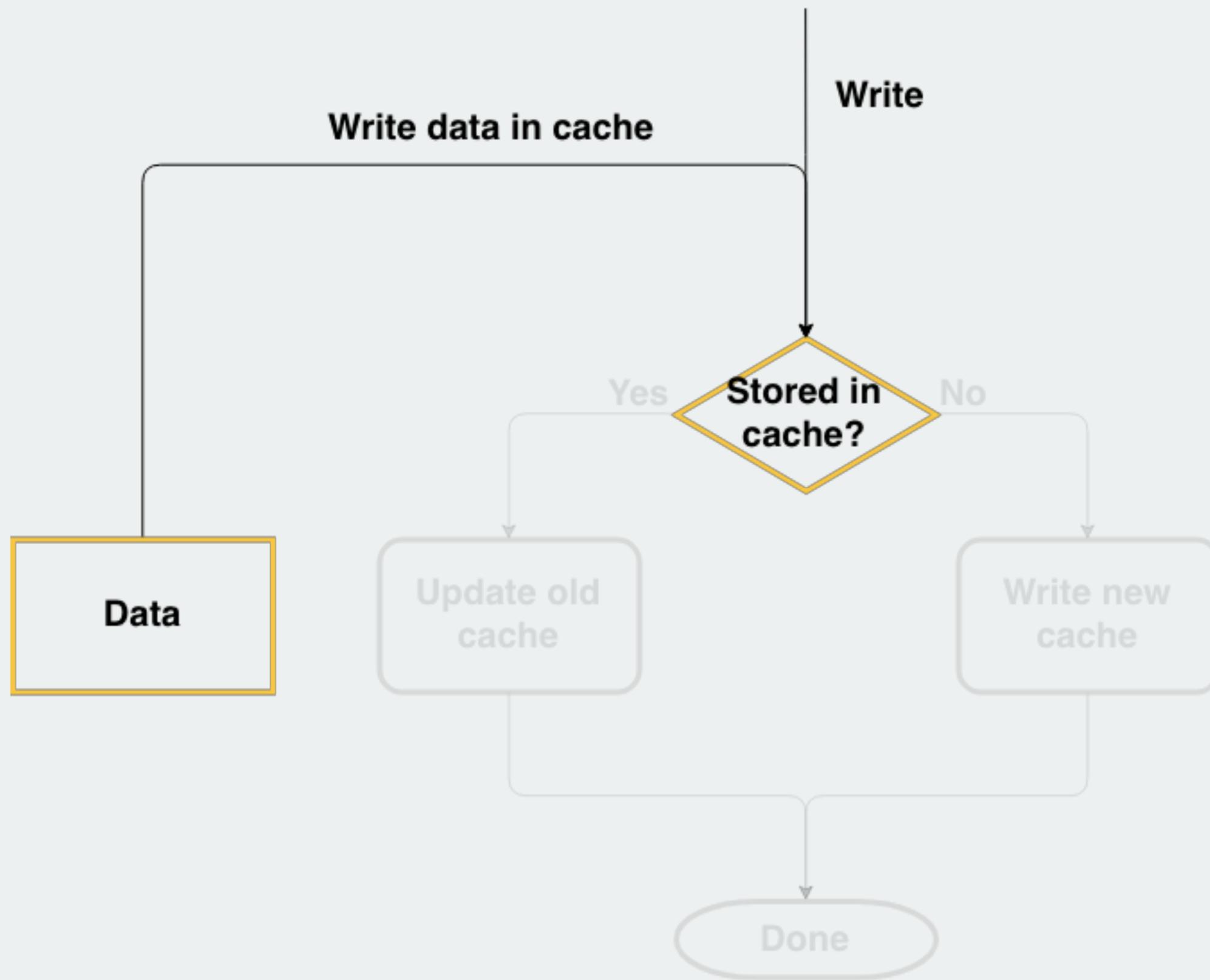


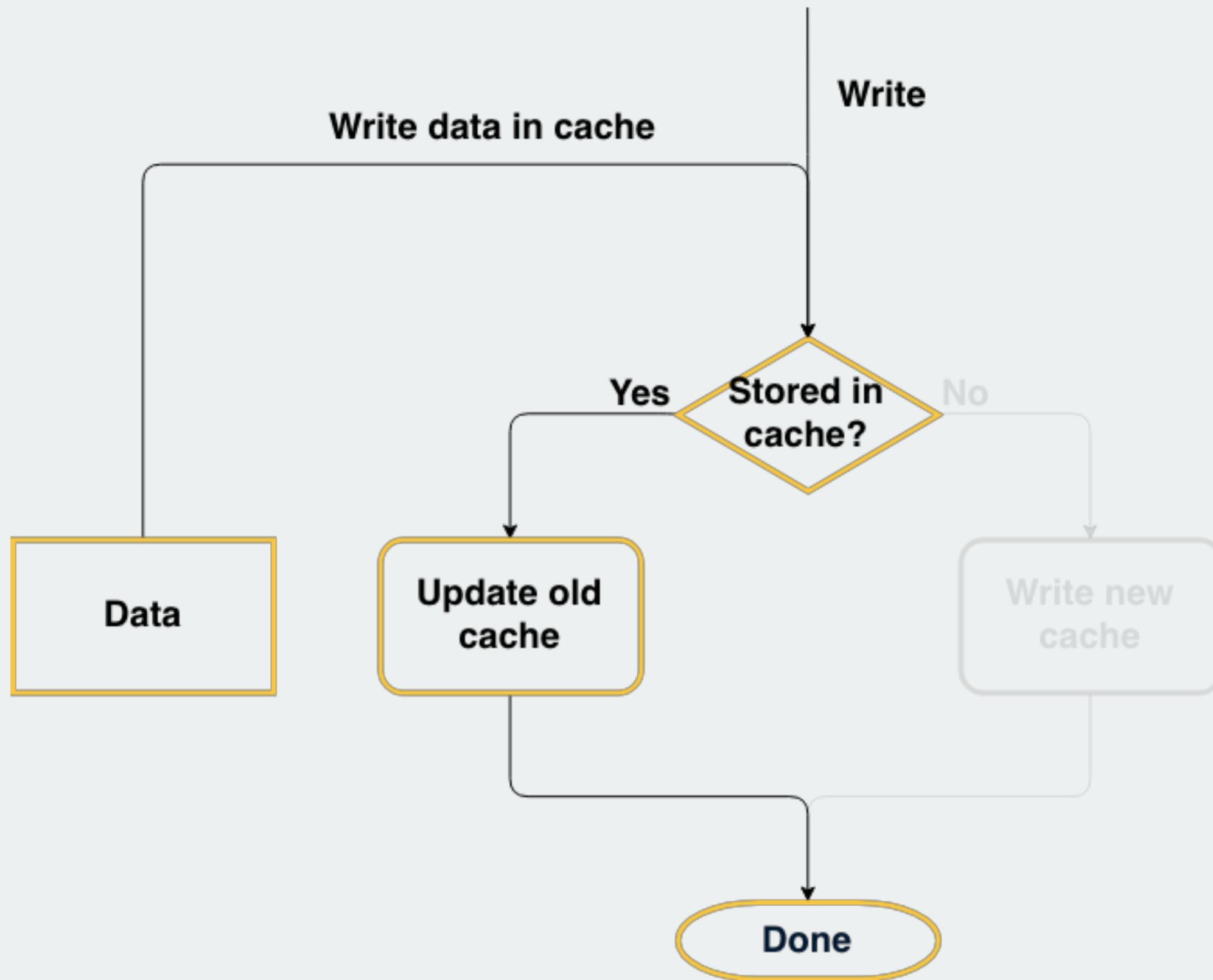


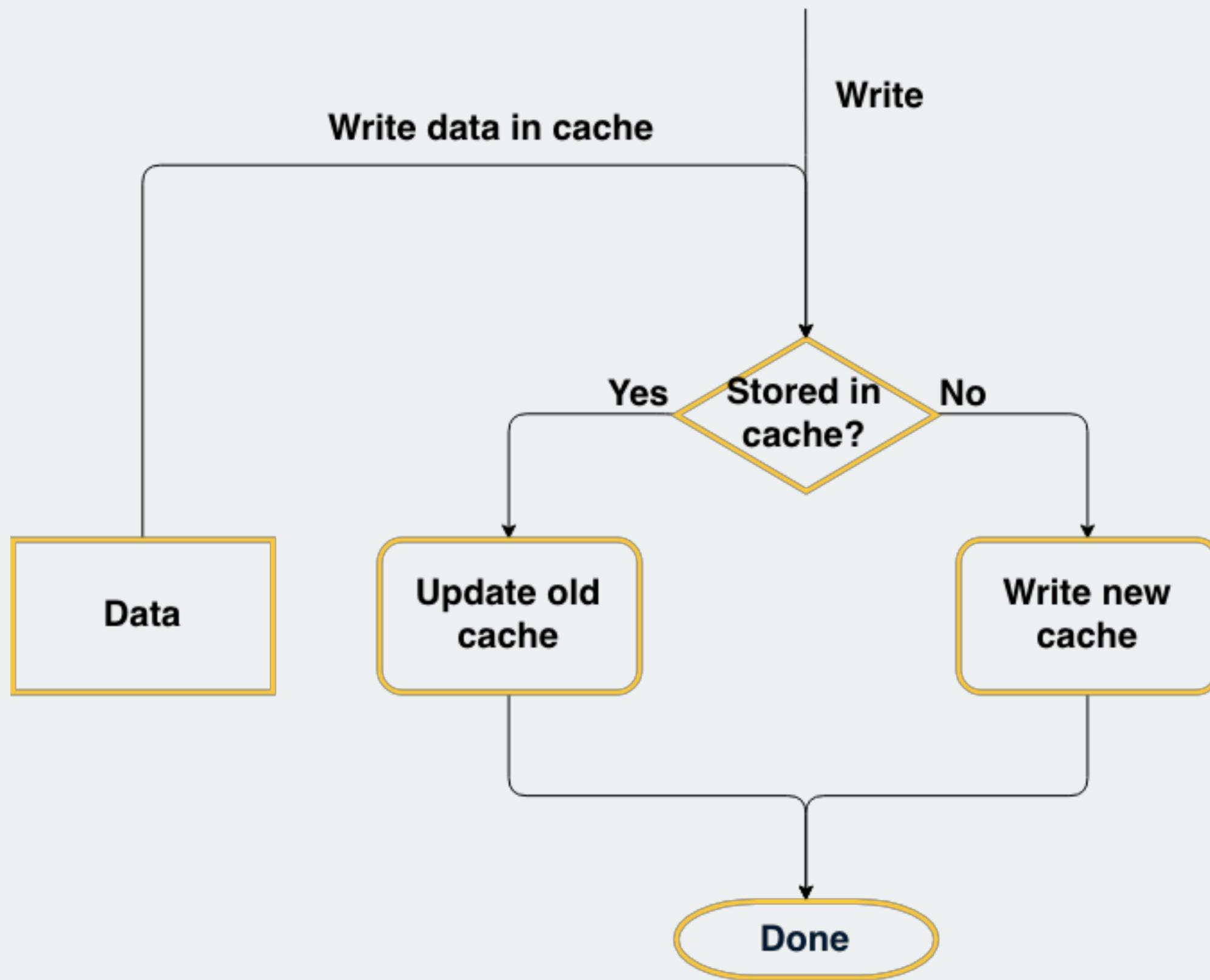








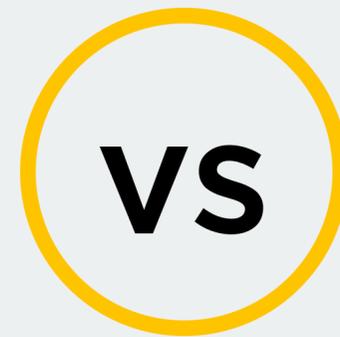






# Плюсы и минусы

- Ускоренная доставка актуальных данных





# Плюсы и минусы

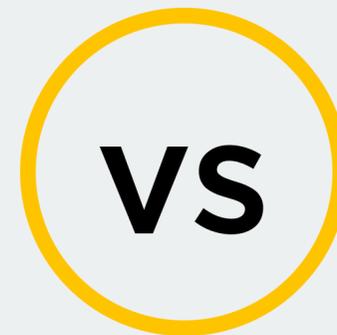
- Ускоренная доставка актуальных данных
- Инвалидация кэша





# Плюсы и минусы

- Ускоренная доставка актуальных данных
- Инвалидация кэша

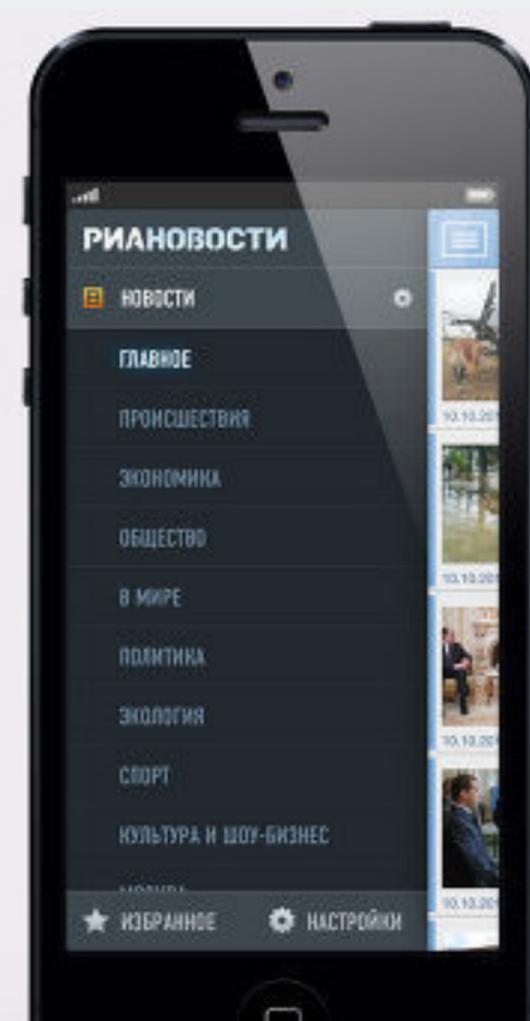
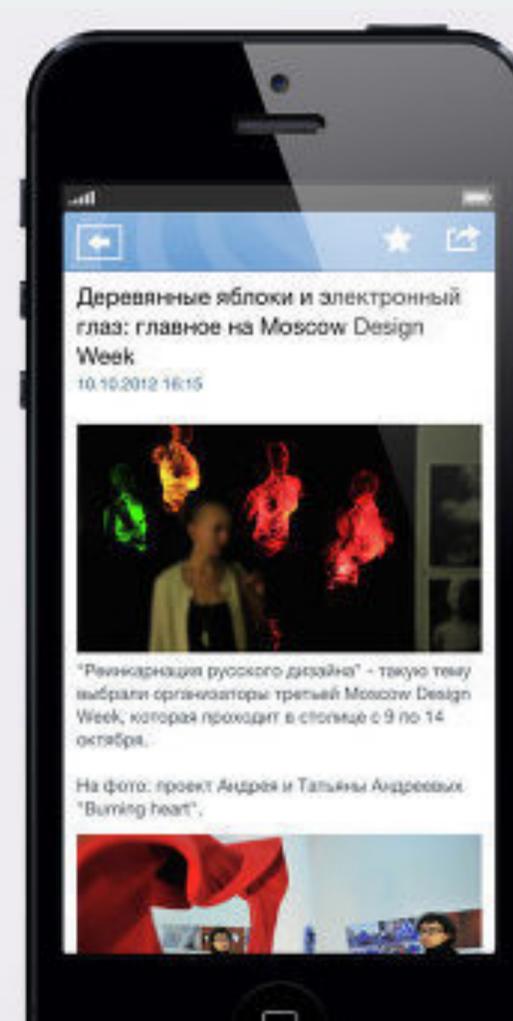
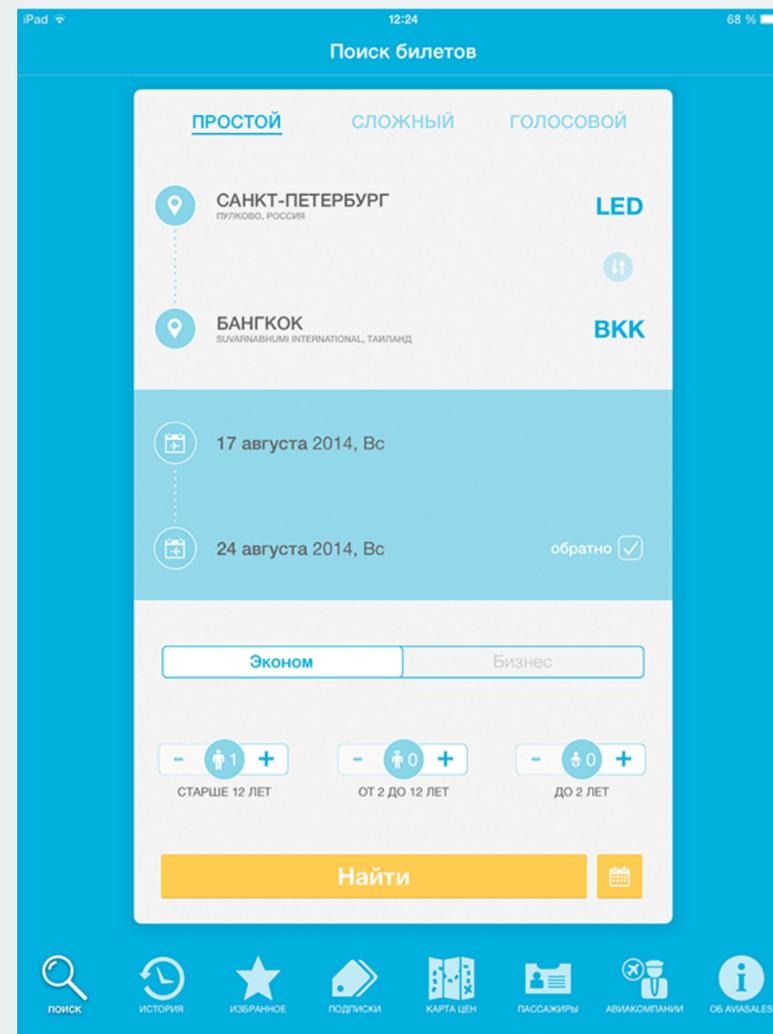
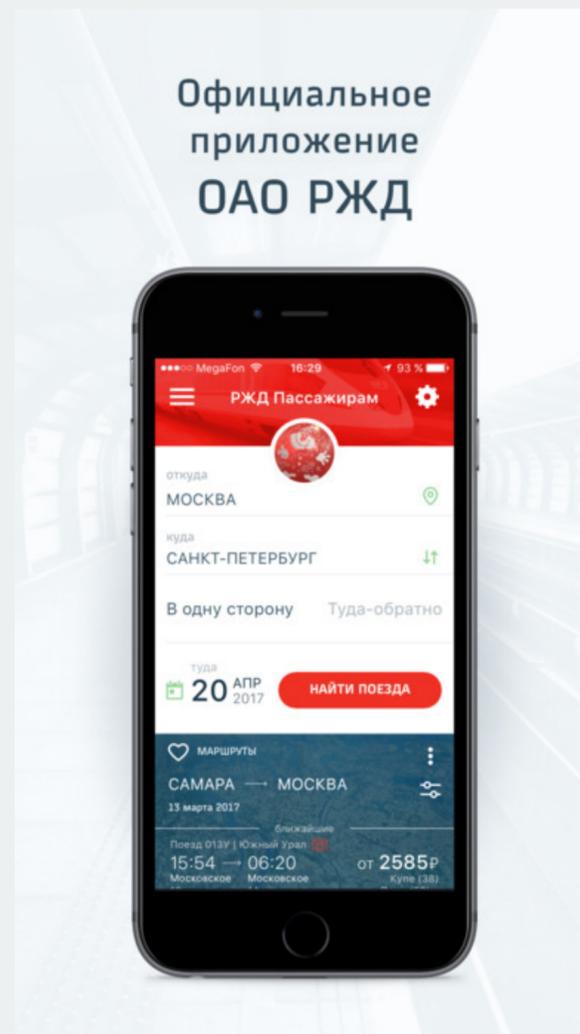


- Зависимость от соединения



# Где использовать?

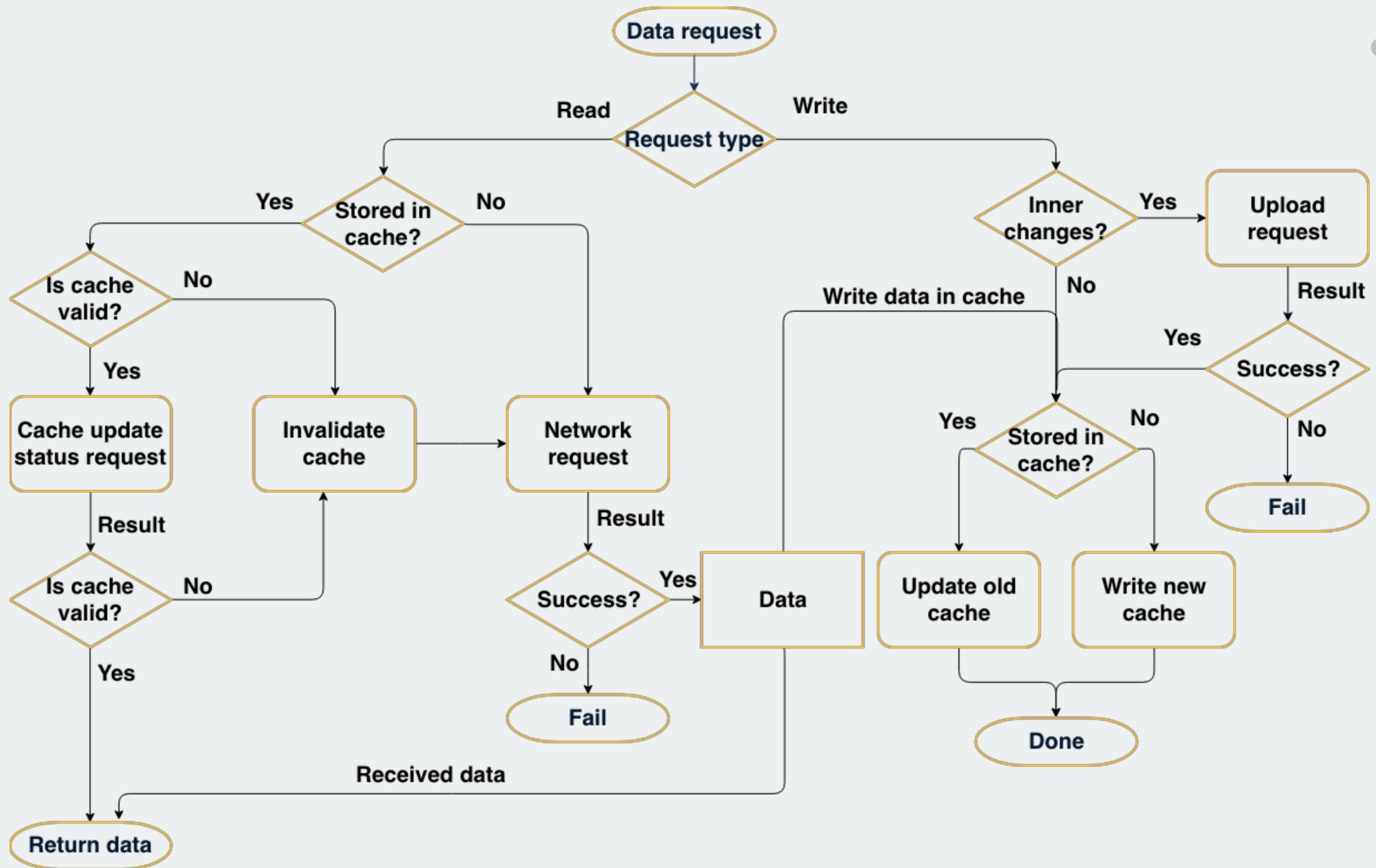
Приложения с не идемпотентными данными. При этом пользователь не может добавить или изменить их.

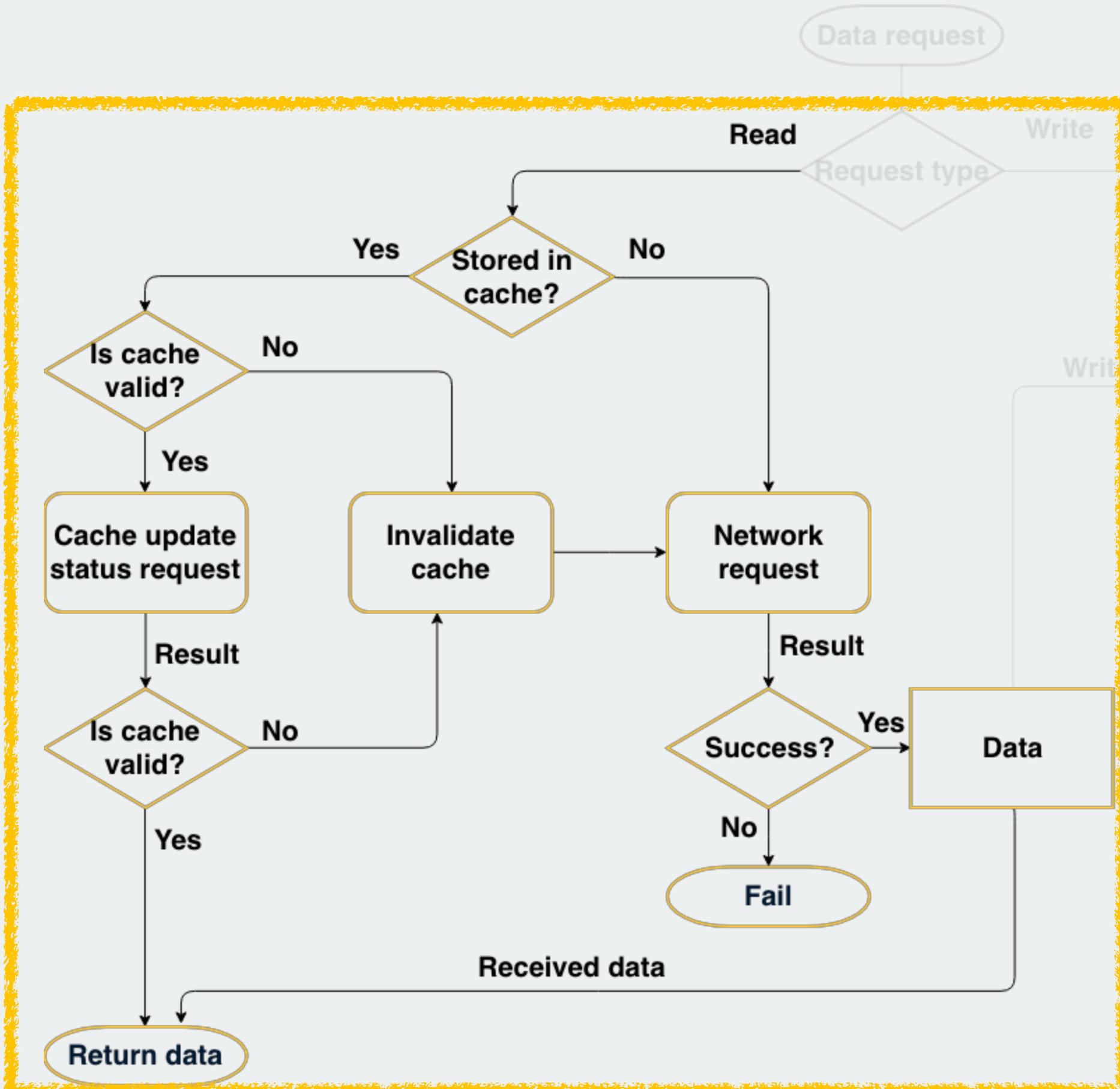


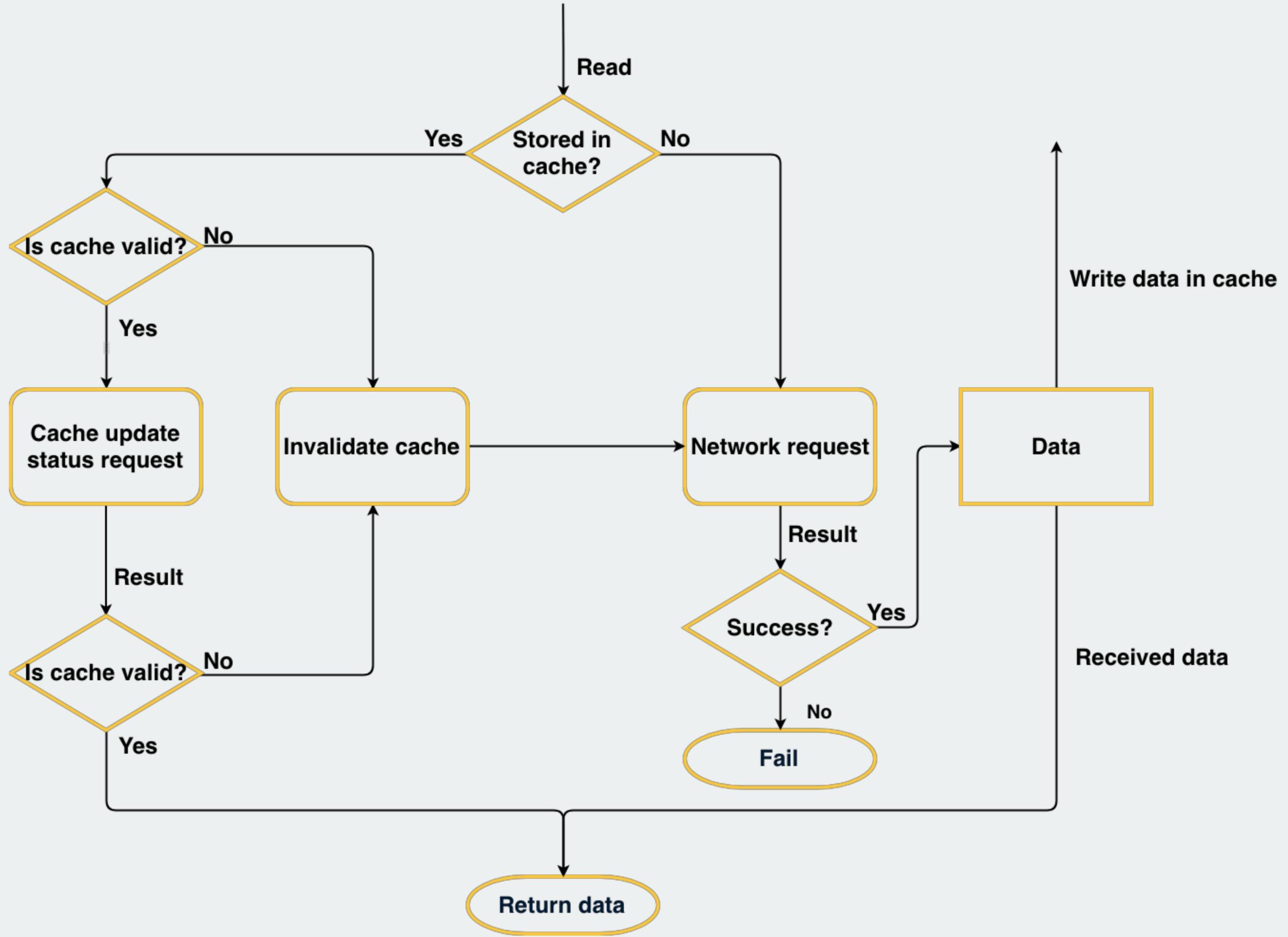


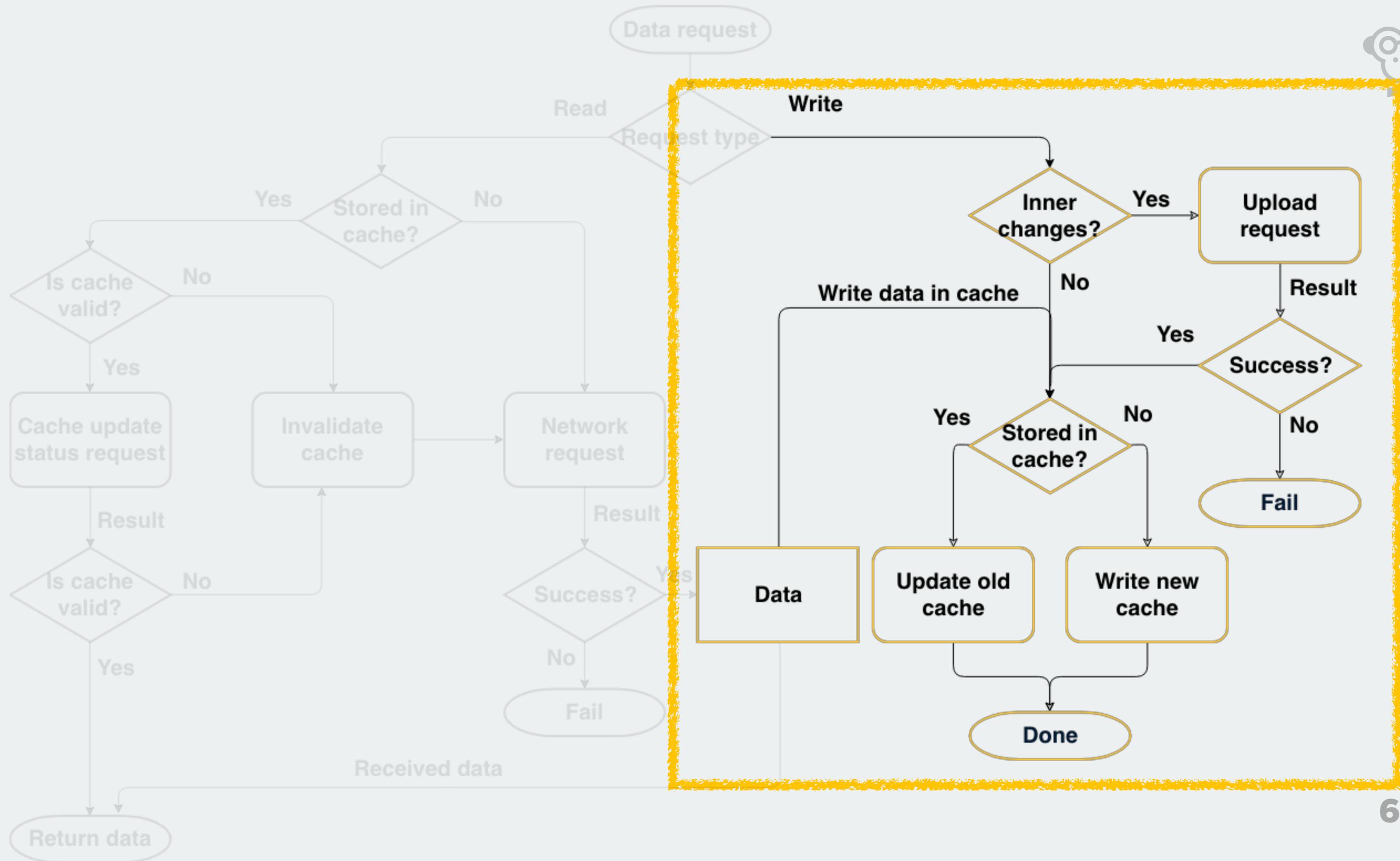
3

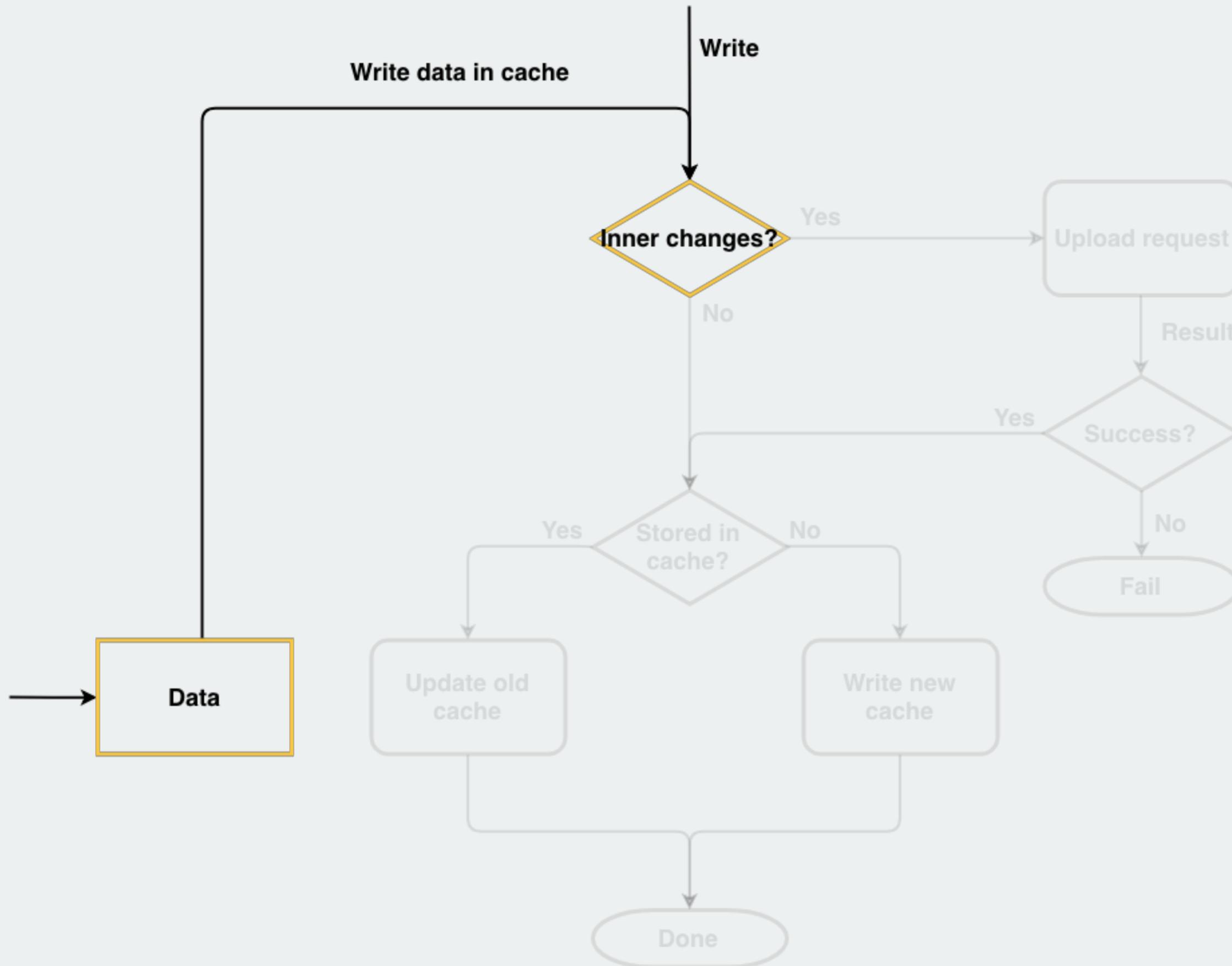
# Write-through cache

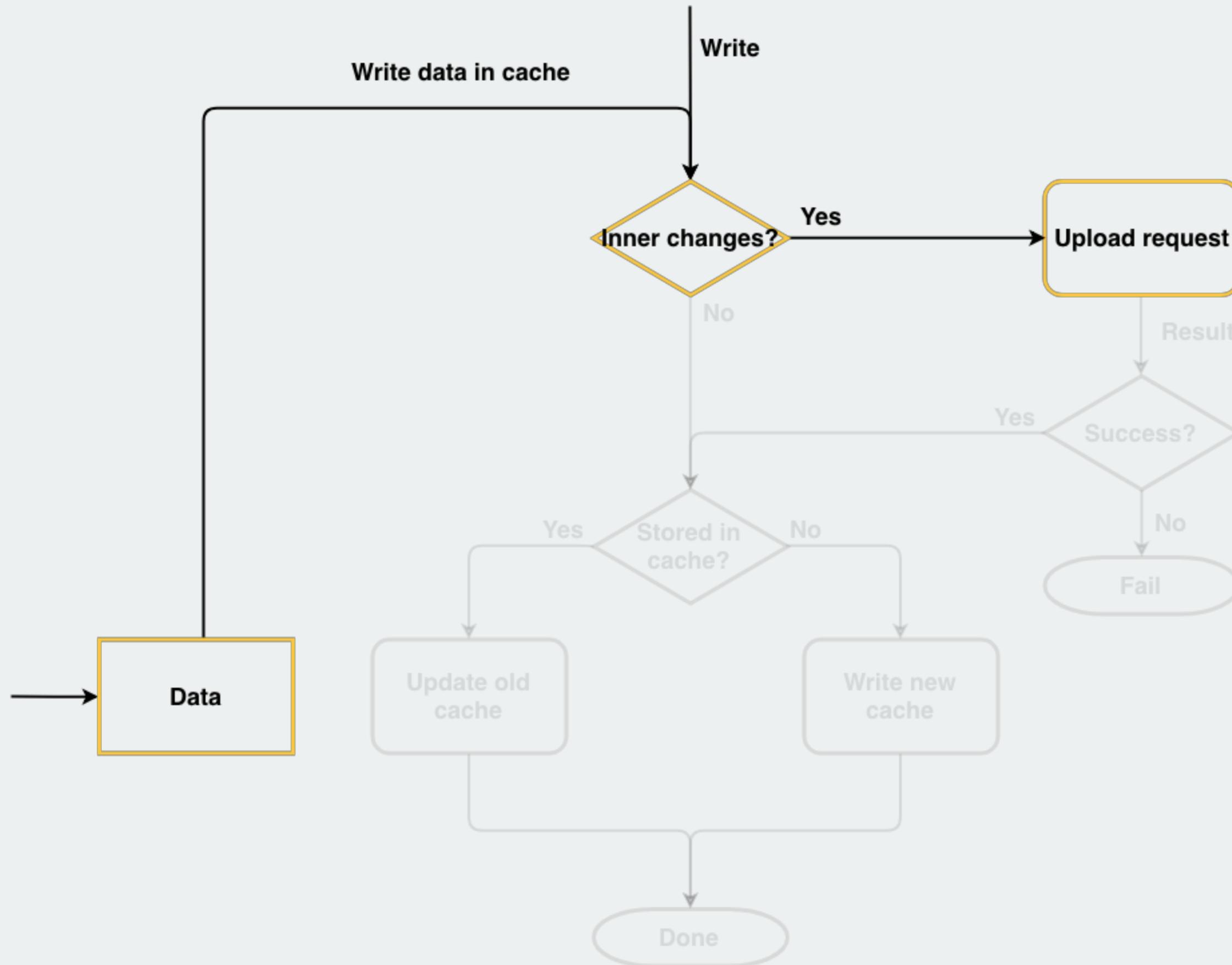


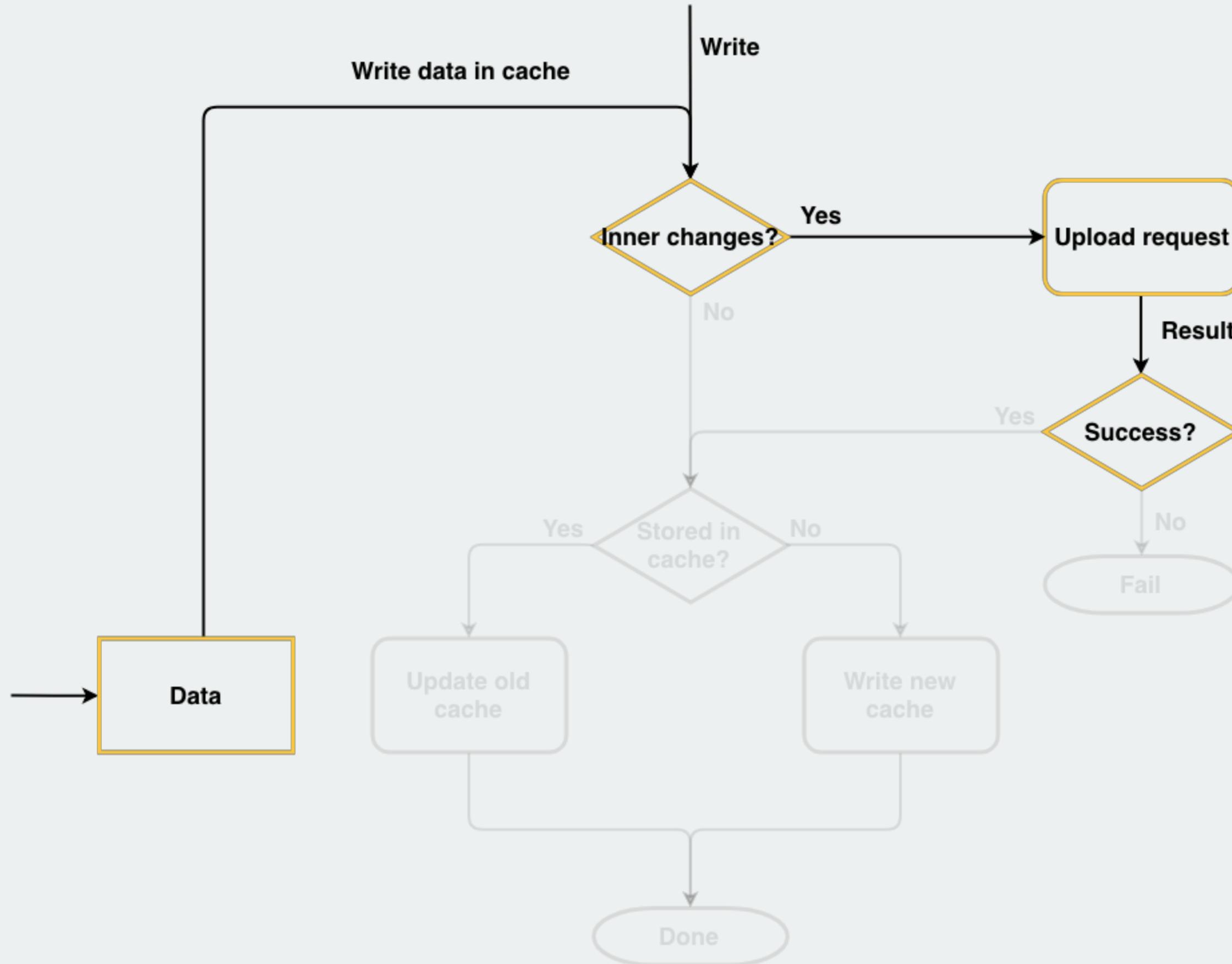


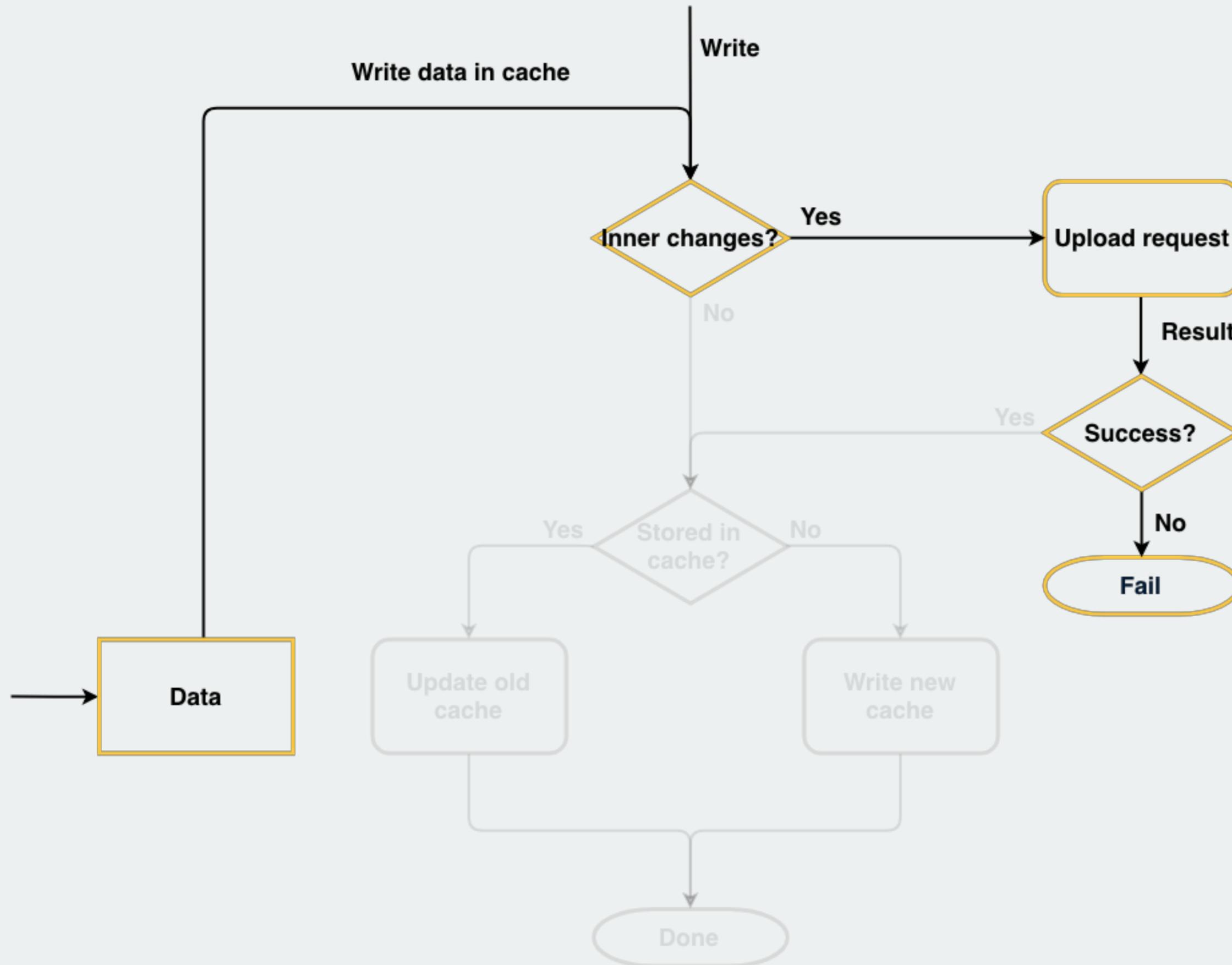


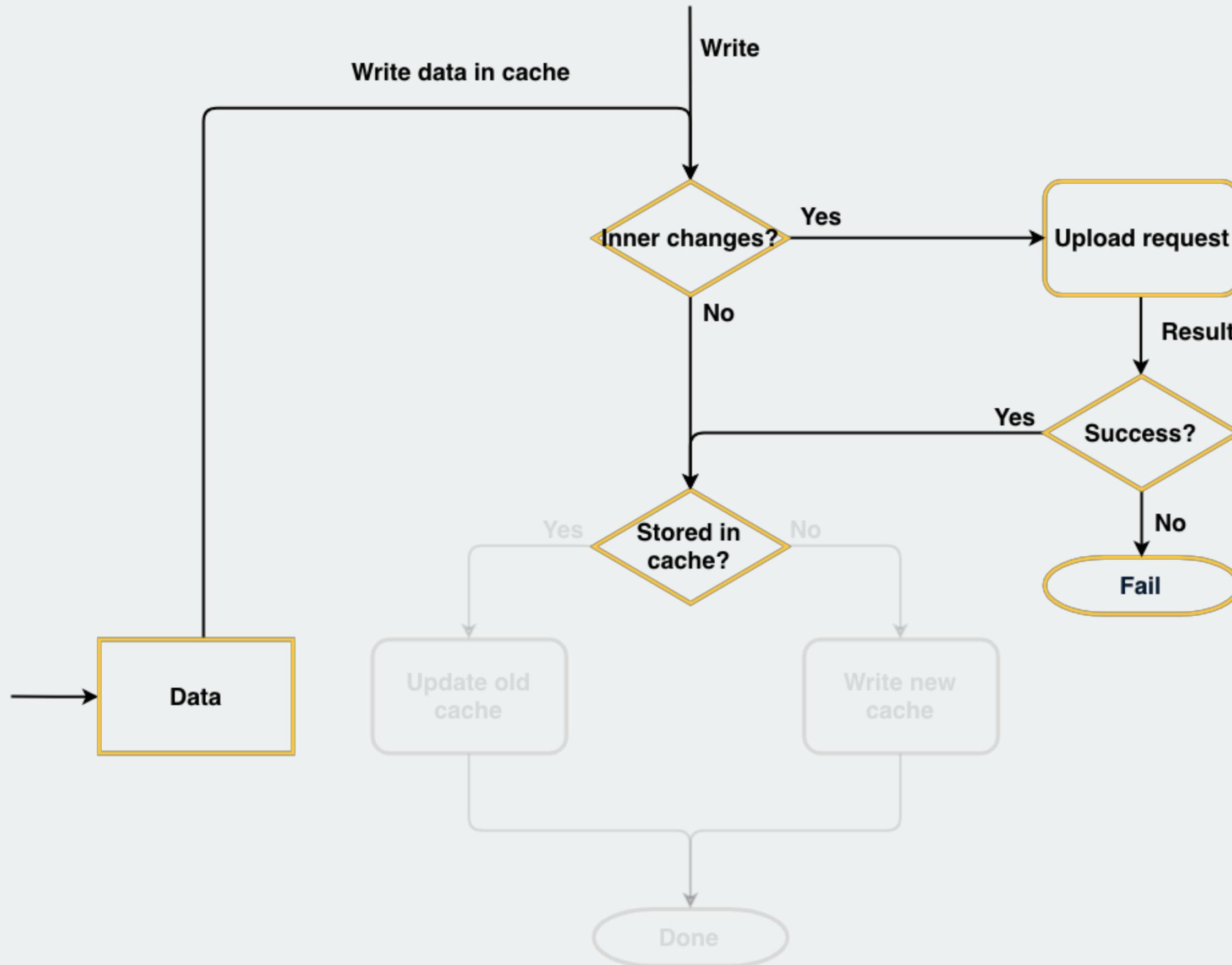


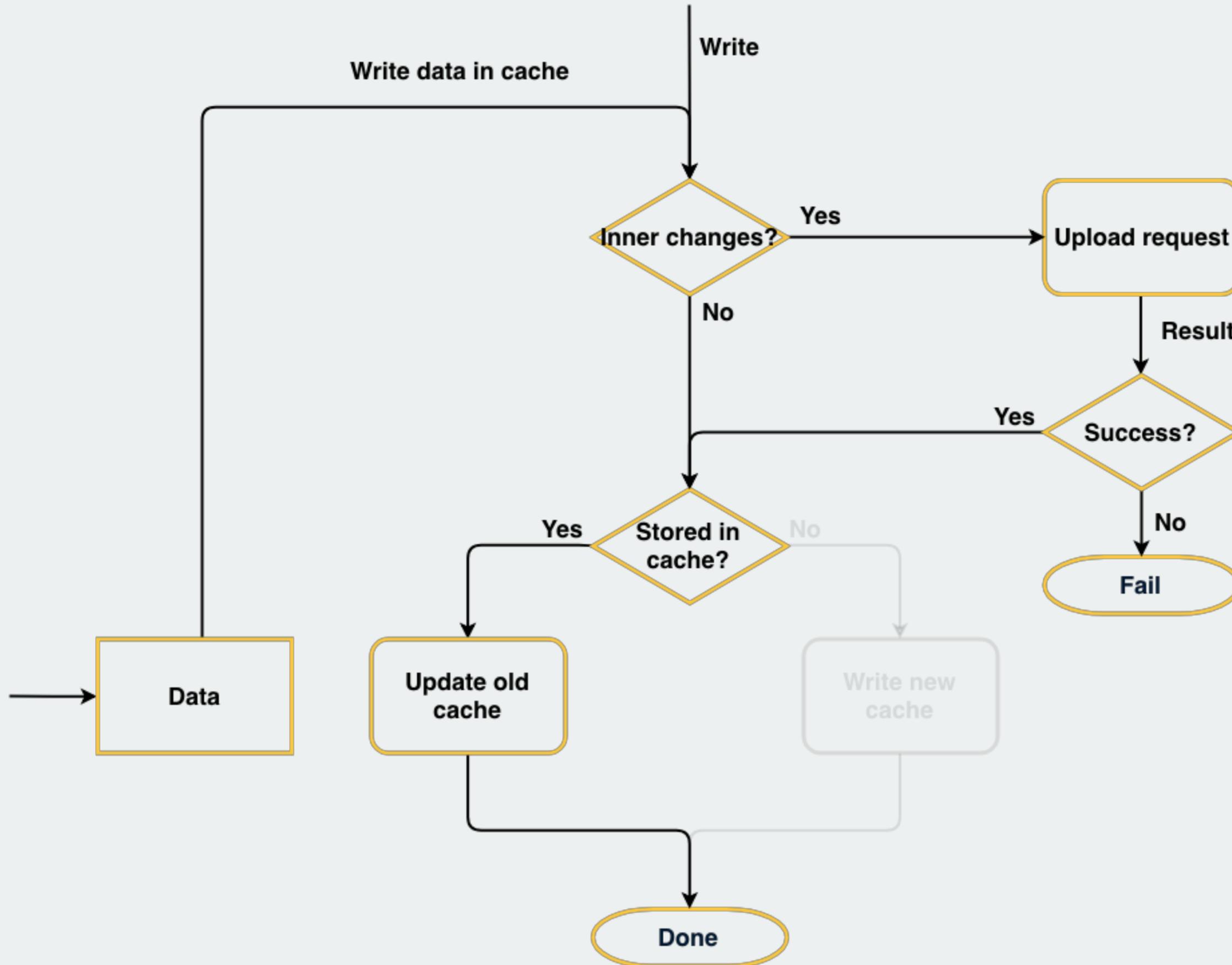


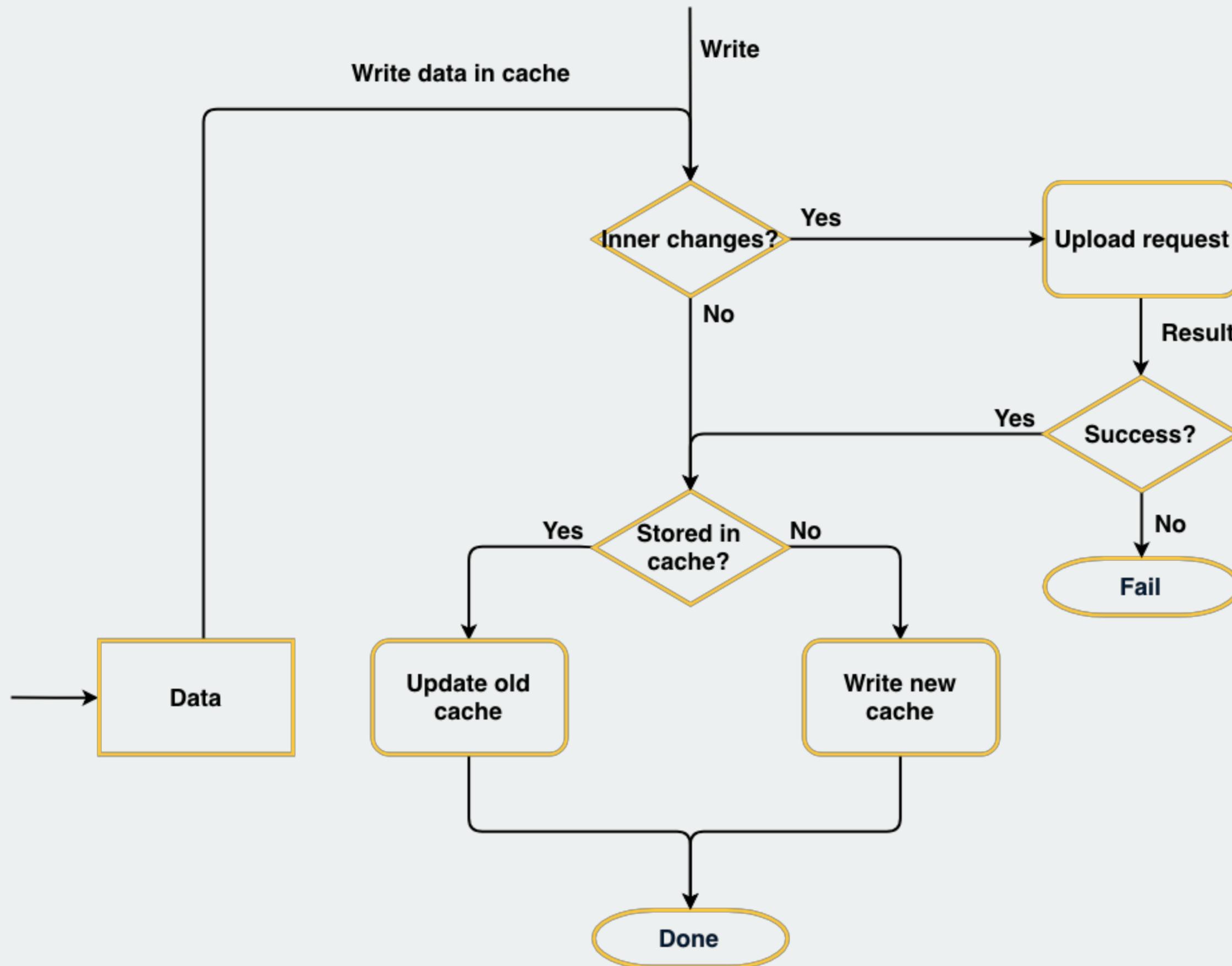














# Плюсы и минусы

- Ускоренная доставка актуальных данных





# Плюсы и минусы

- Ускоренная доставка актуальных данных
- Полная синхронизация с сервером





# Плюсы и минусы

- Ускоренная доставка актуальных данных
- Полная синхронизация с сервером



- Зависимость от соединения



# Плюсы и минусы

- Ускоренная доставка актуальных данных
- Полная синхронизация с сервером



- Зависимость от соединения
- При неудачной записи нужно возвращаться в исходное состояние



# Плюсы и минусы

- Ускоренная доставка актуальных данных
- Полная синхронизация с сервером

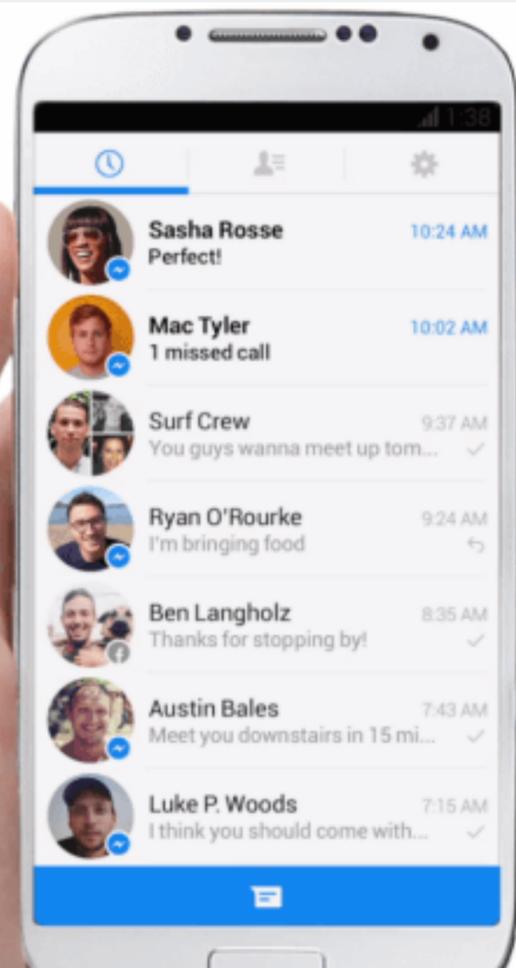


- Зависимость от соединения
- При неудачной записи нужно возвращаться в исходное состояние
- Сложная реализация



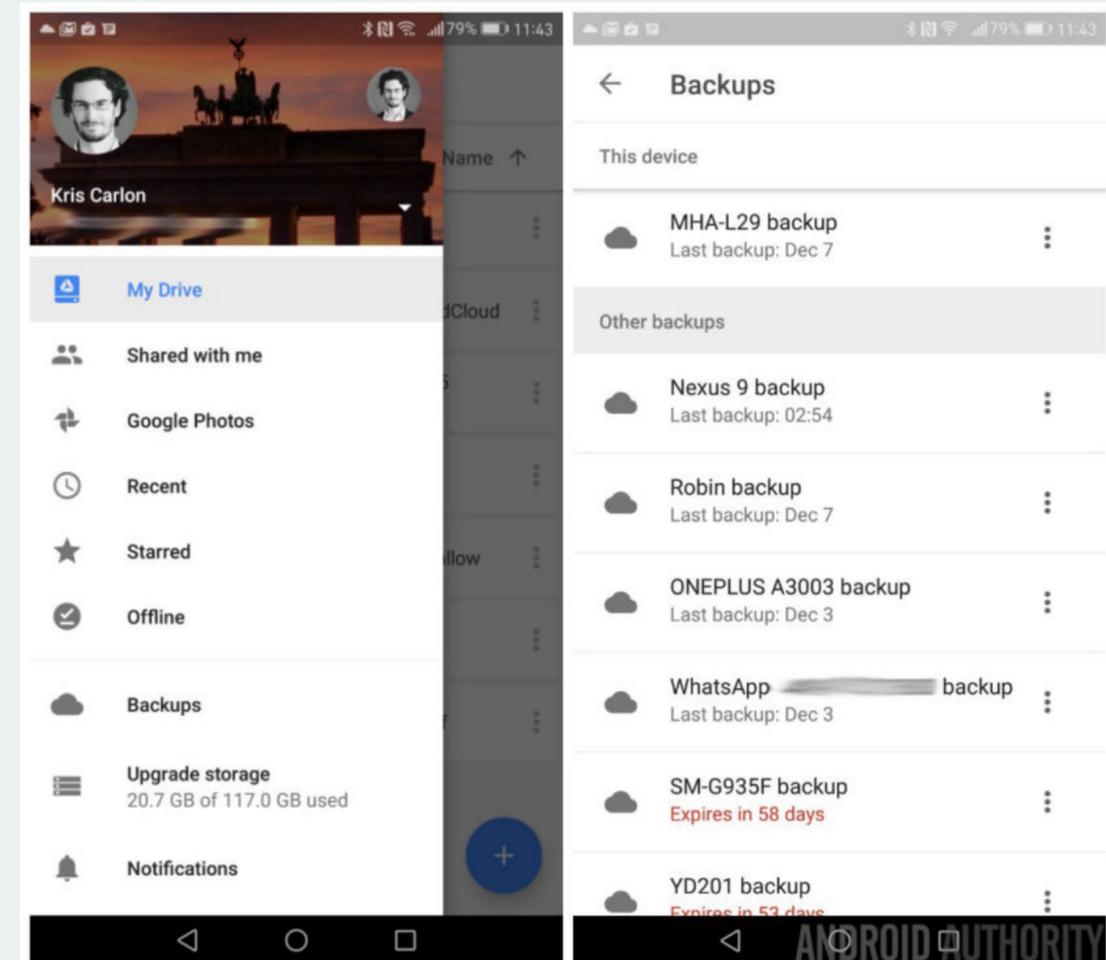
# Где использовать?

Приложения с не идемпотентными данными. При этом пользователь может добавить или изменить их.



Get Messenger

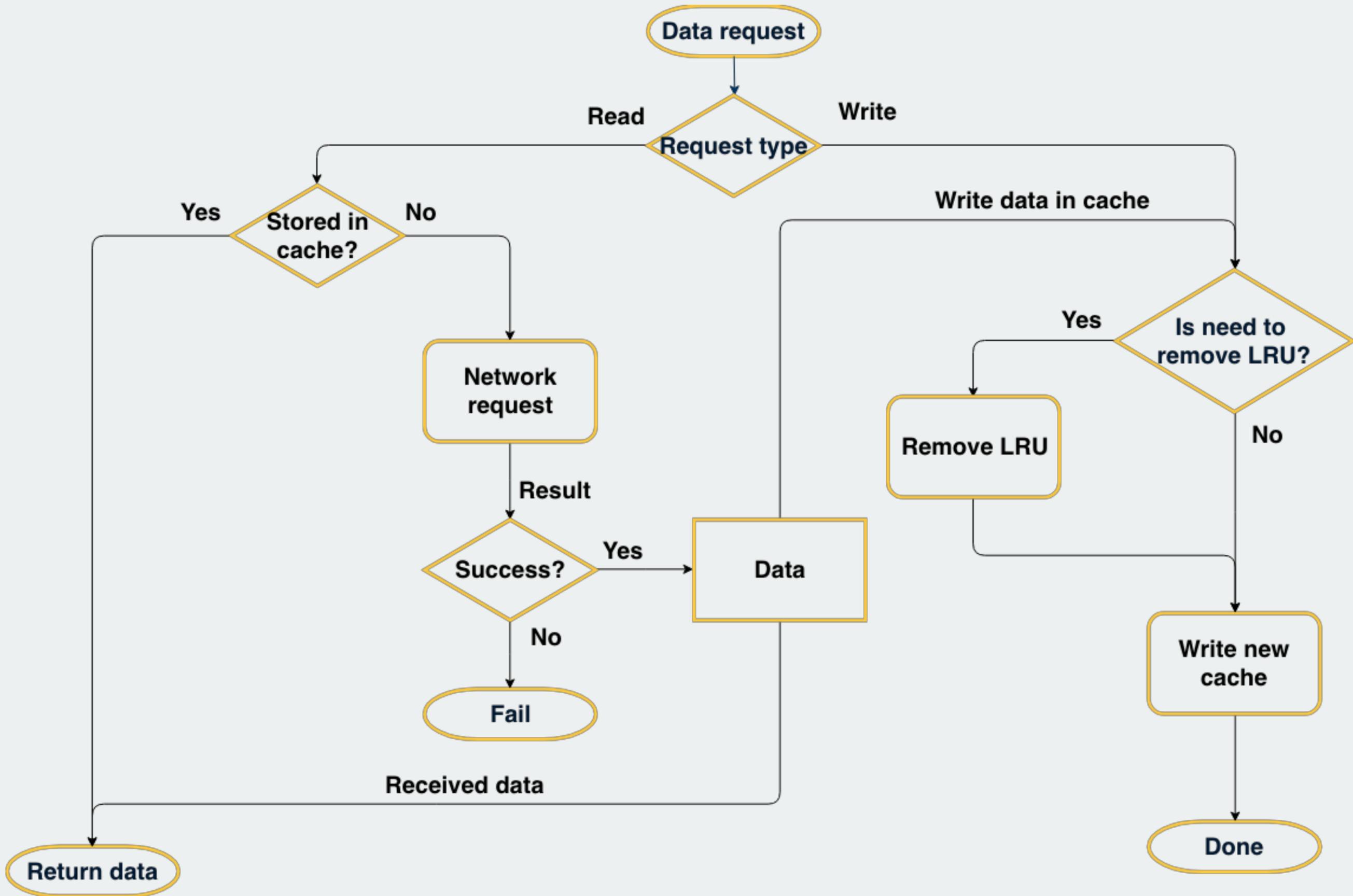
Available now for Android, iPhone and Windows Phone

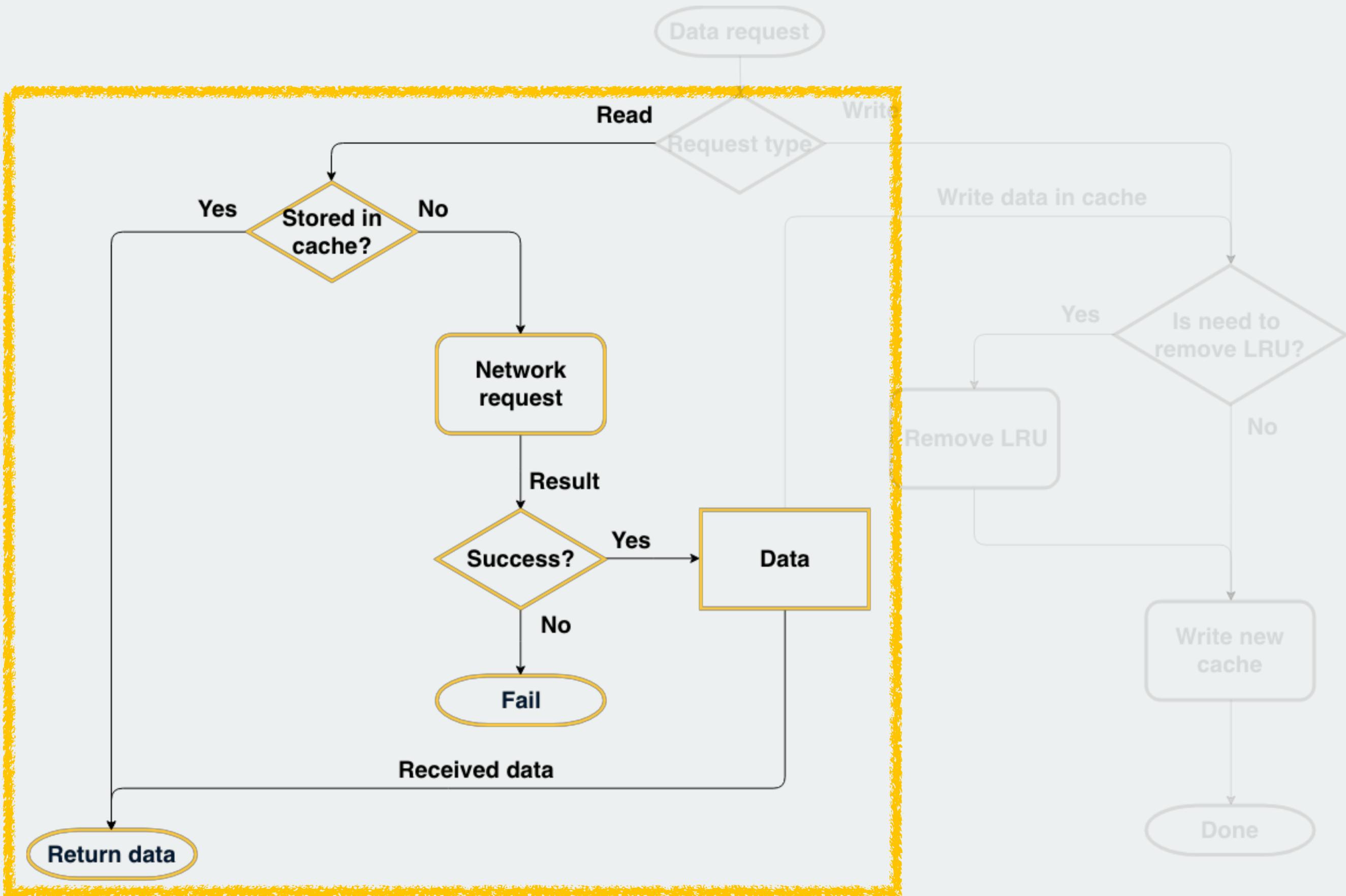


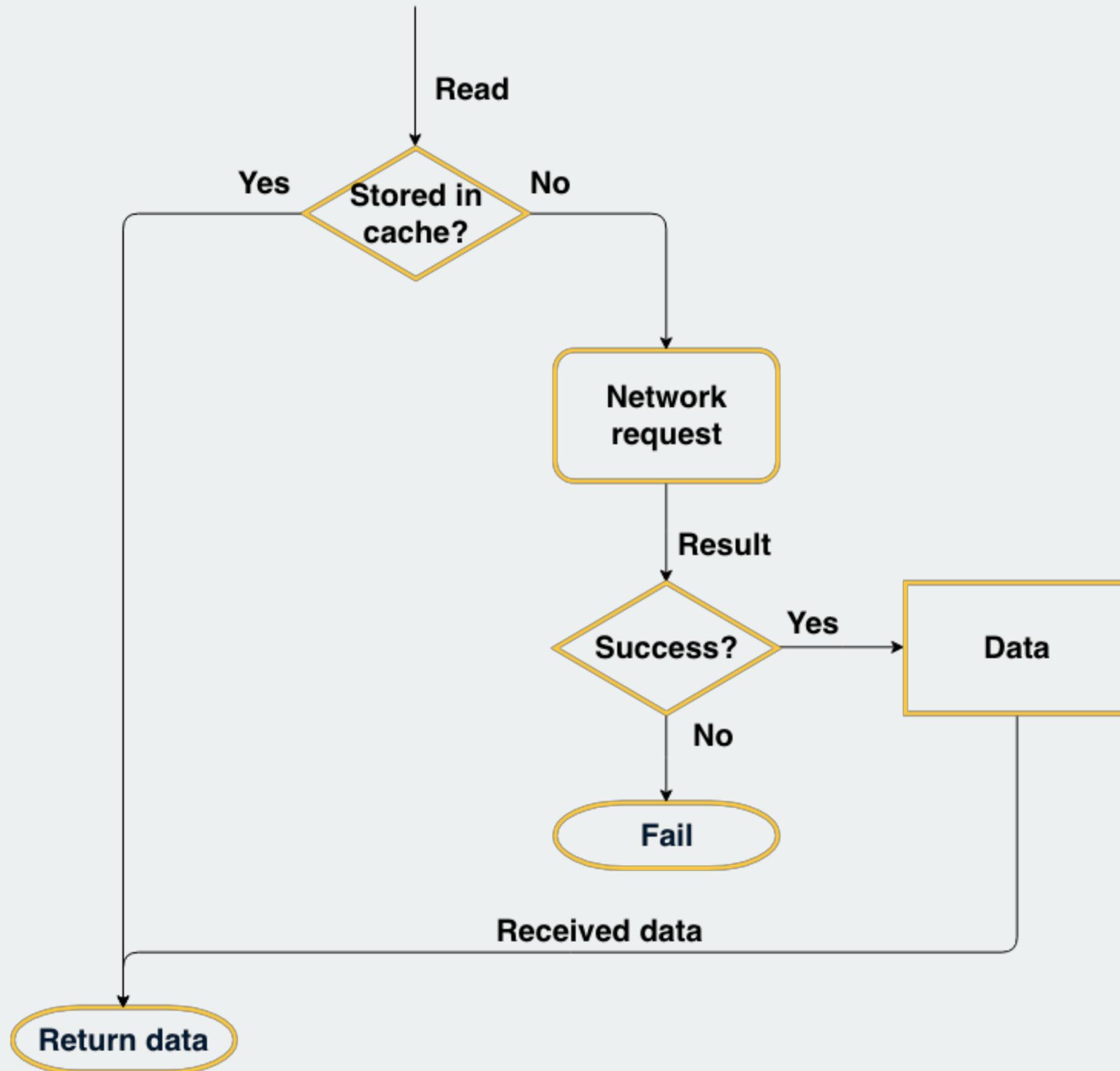


4

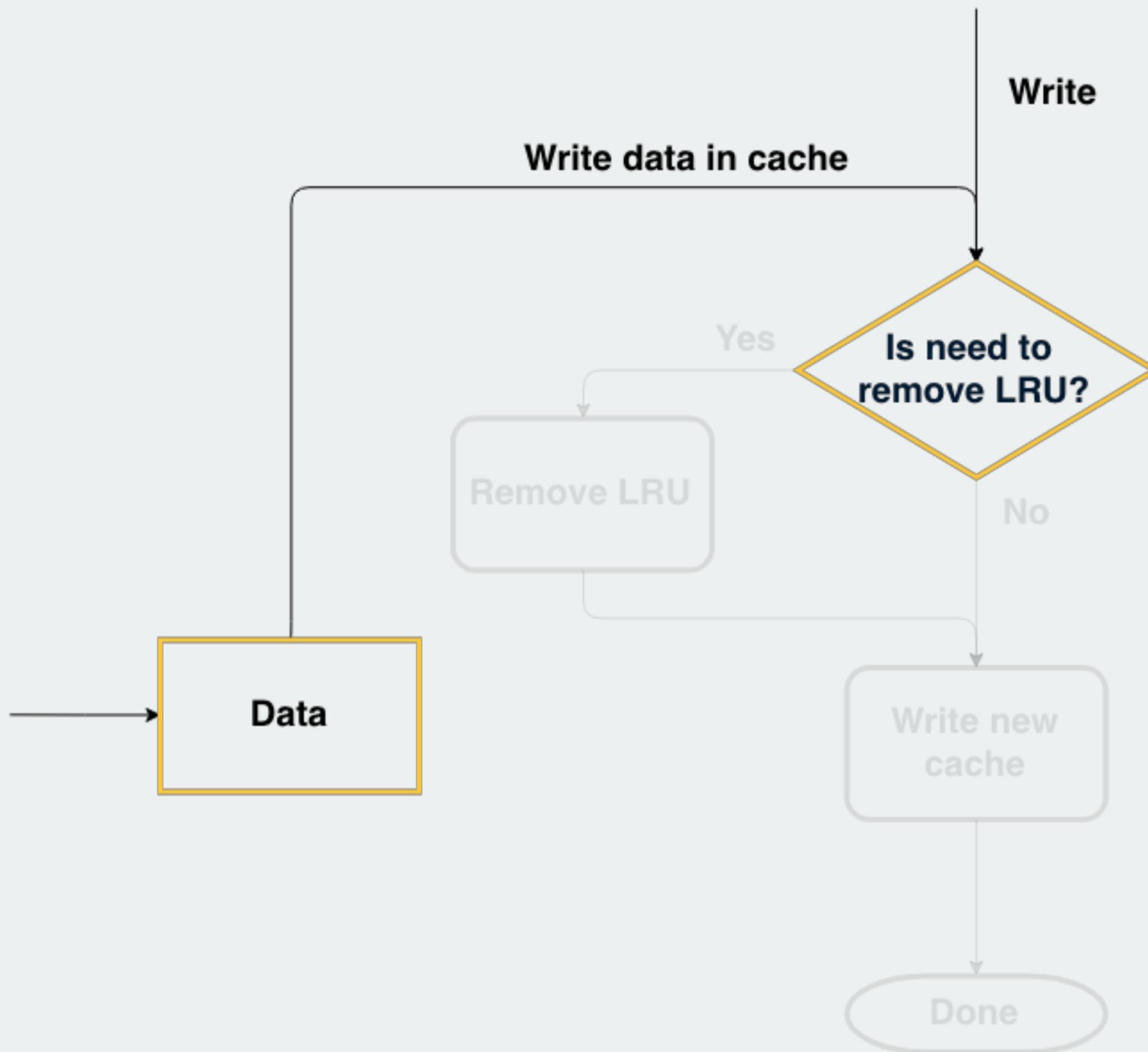
# LRU cache

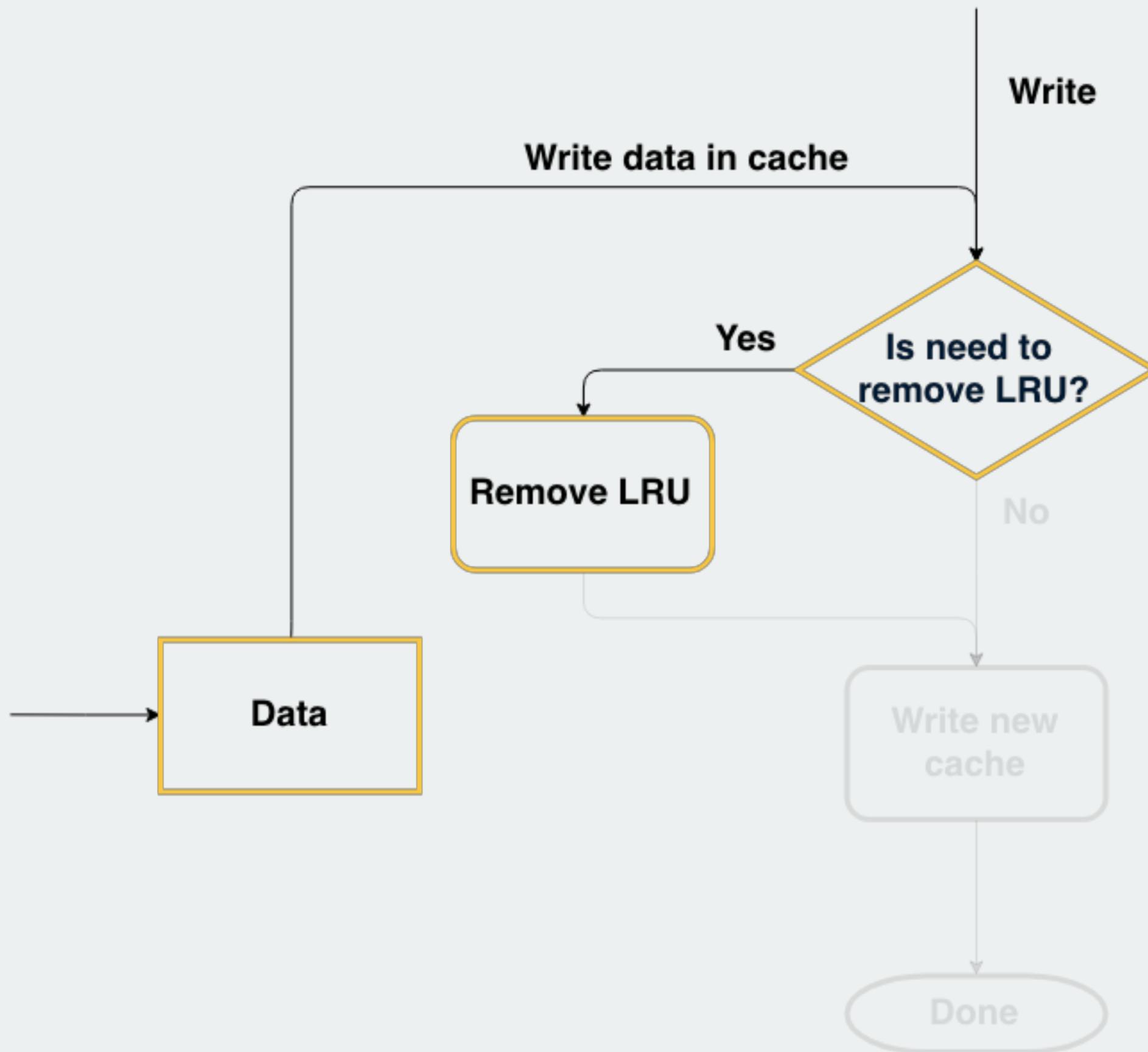


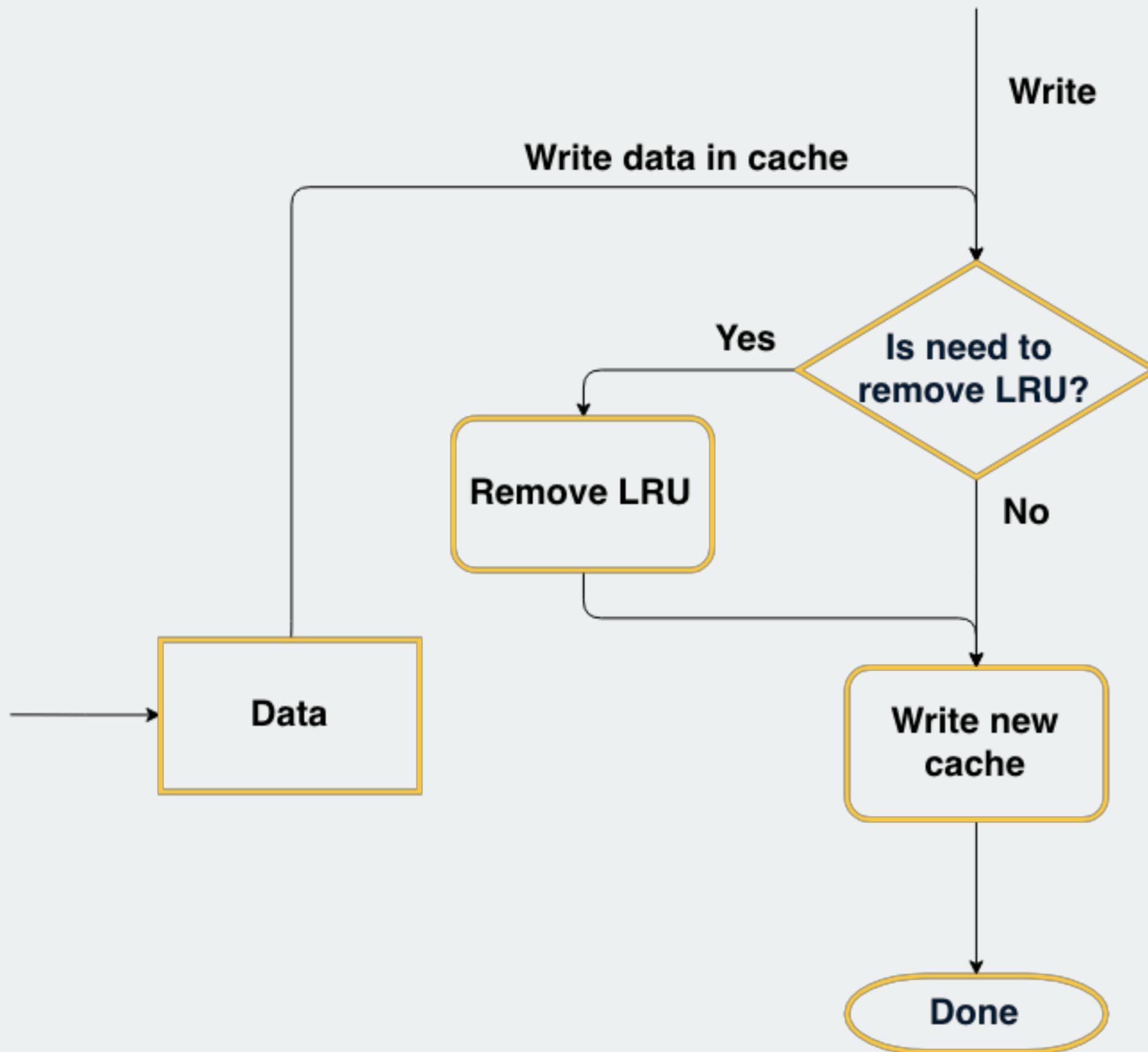














# Плюсы и минусы

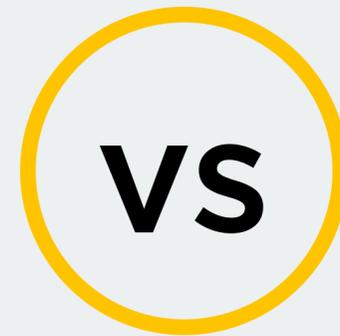
- **Зашифрованные данные  
не зависят от соединения**





# Плюсы и минусы

- **Закашированные данные не зависят от соединения**
- **Мгновенная доставка данных**





# Плюсы и минусы

- **Зашифрованные данные не зависят от соединения**
- **Мгновенная доставка данных**
- **Гибко настраиваемая инвалидация**





# Плюсы и минусы

- Зашифрованные данные не зависят от соединения
- Мгновенная доставка данных
- Гибко настраиваемая инвалидация



- Необходимо следить за инвалидацией



# Плюсы и минусы

- Закешированные данные не зависят от соединения
- Мгновенная доставка данных
- Гибко настраиваемая инвалидация

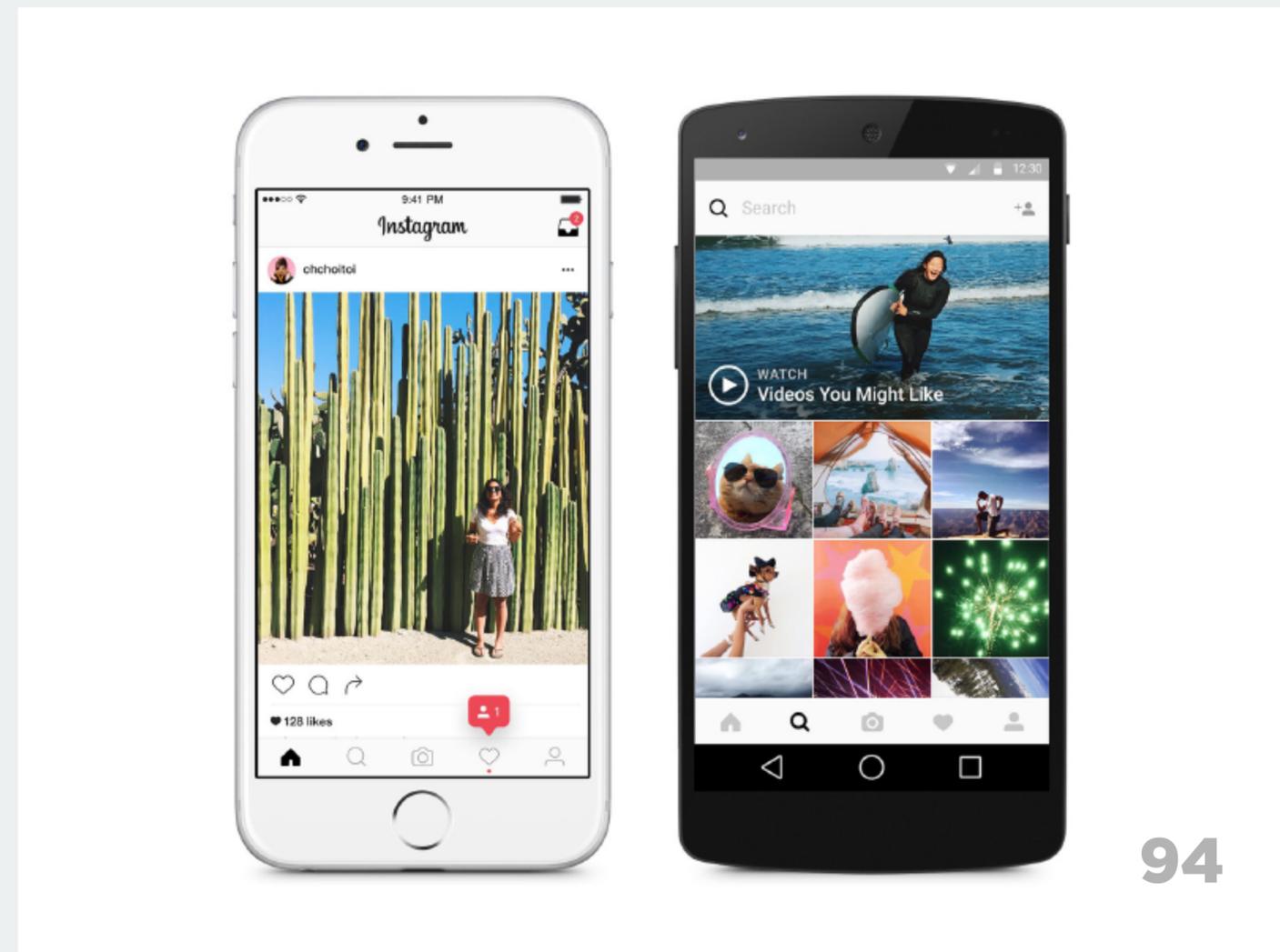
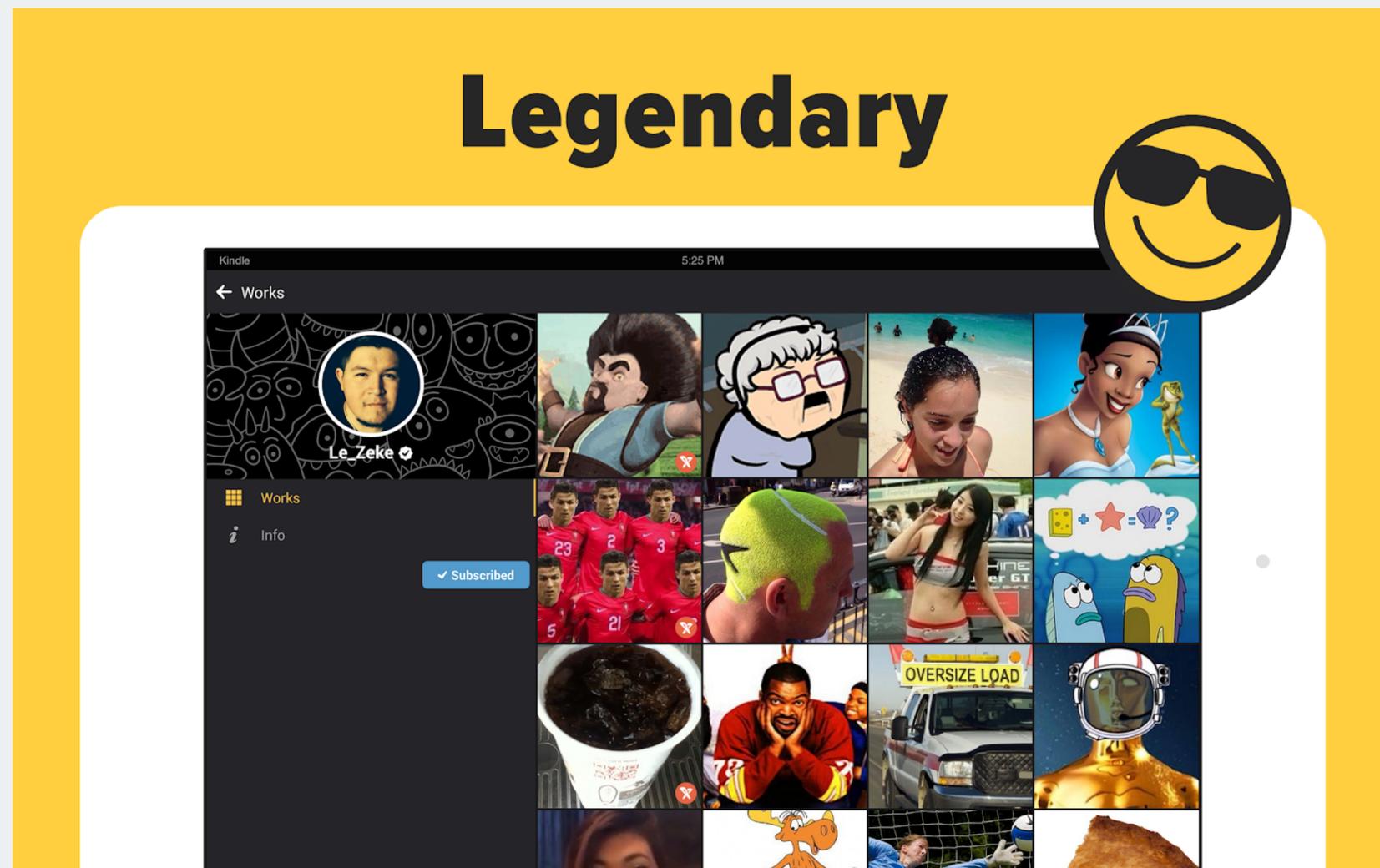


- Необходимо следить за инвалидацией
- Чаще всего сложная реализация



# Где использовать?

Приложения с тяжеловесным контентом.



# Властелин КЭШа:

## Возвращение рефракторинга



# Как всё начиналось?



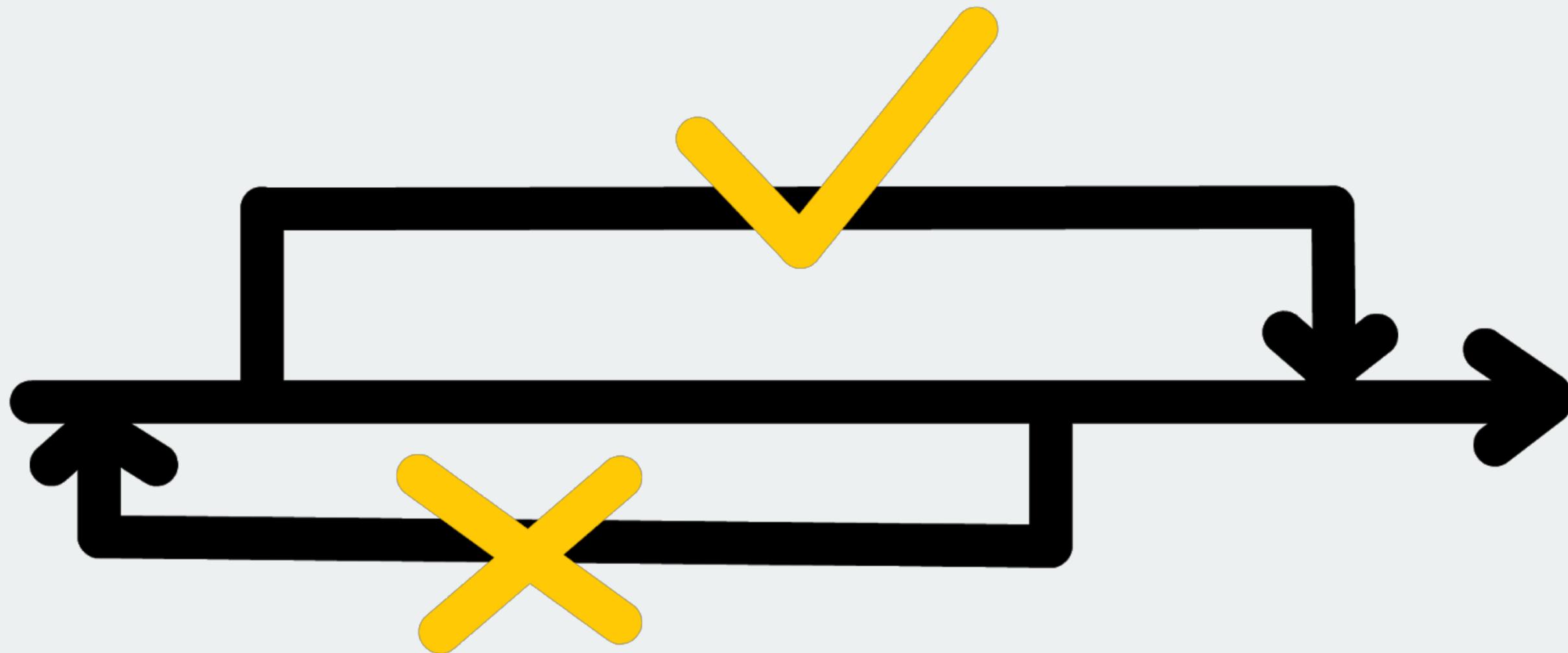
smartdev 11 мая в 11:17

**«Стартуем! Я сказала: стартуем!», или как мы реализовывали работу с faststart-видео под Android**

Блог компании FunCorp, Kotlin, Работа с видео, Разработка мобильных приложений, Разработка под Android



# Почему не стримы?



# Первая реализация DataSource

```
boolean hasStreamBytes = readBytes != RESULT_END_OF_INPUT && readBytes != 0;
if (hasStreamBytes) {
    mCacheFile.write(buffer);
}
if (bytesRead == mMediaCache.getFinalSize()) {
    closeData();
}
return readBytes;
} else if (mReadFromFile) {
    size = (int) Math.min(size, bytesRead - position);
}
if (mCacheFile.getFilePointer() != position) {
    mCacheFile.seek(position);
}
int read = mCacheFile.read(buffer, offset, size);
return read;
```



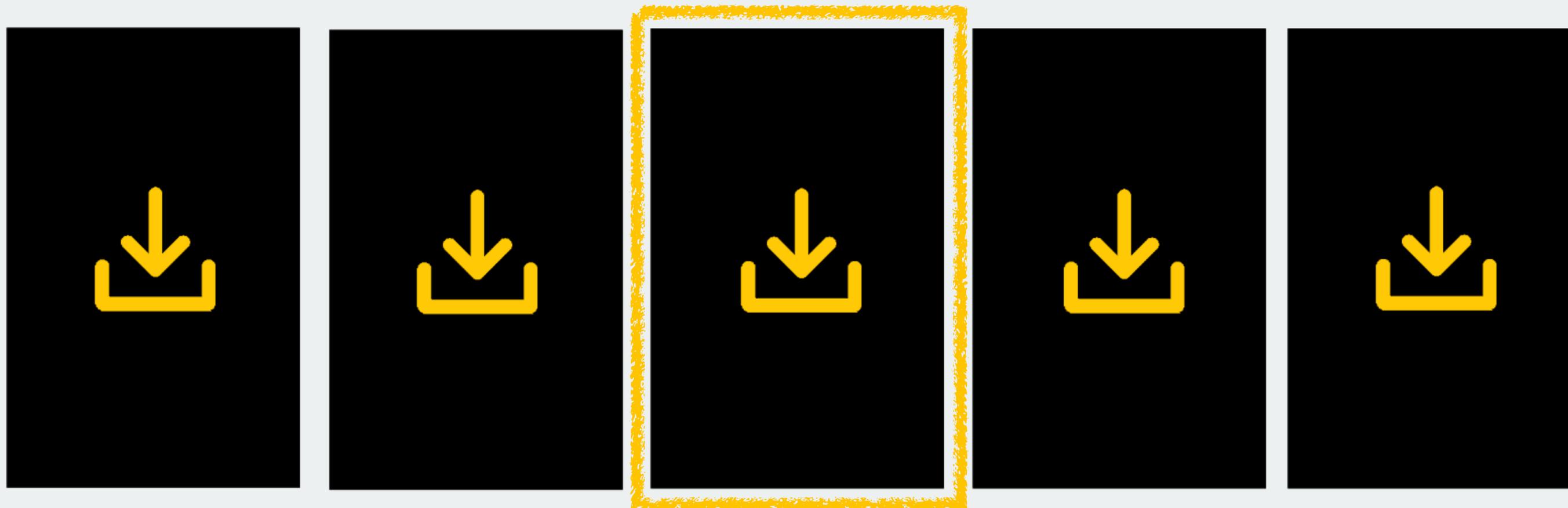
**С самого начала у меня была какая-то**



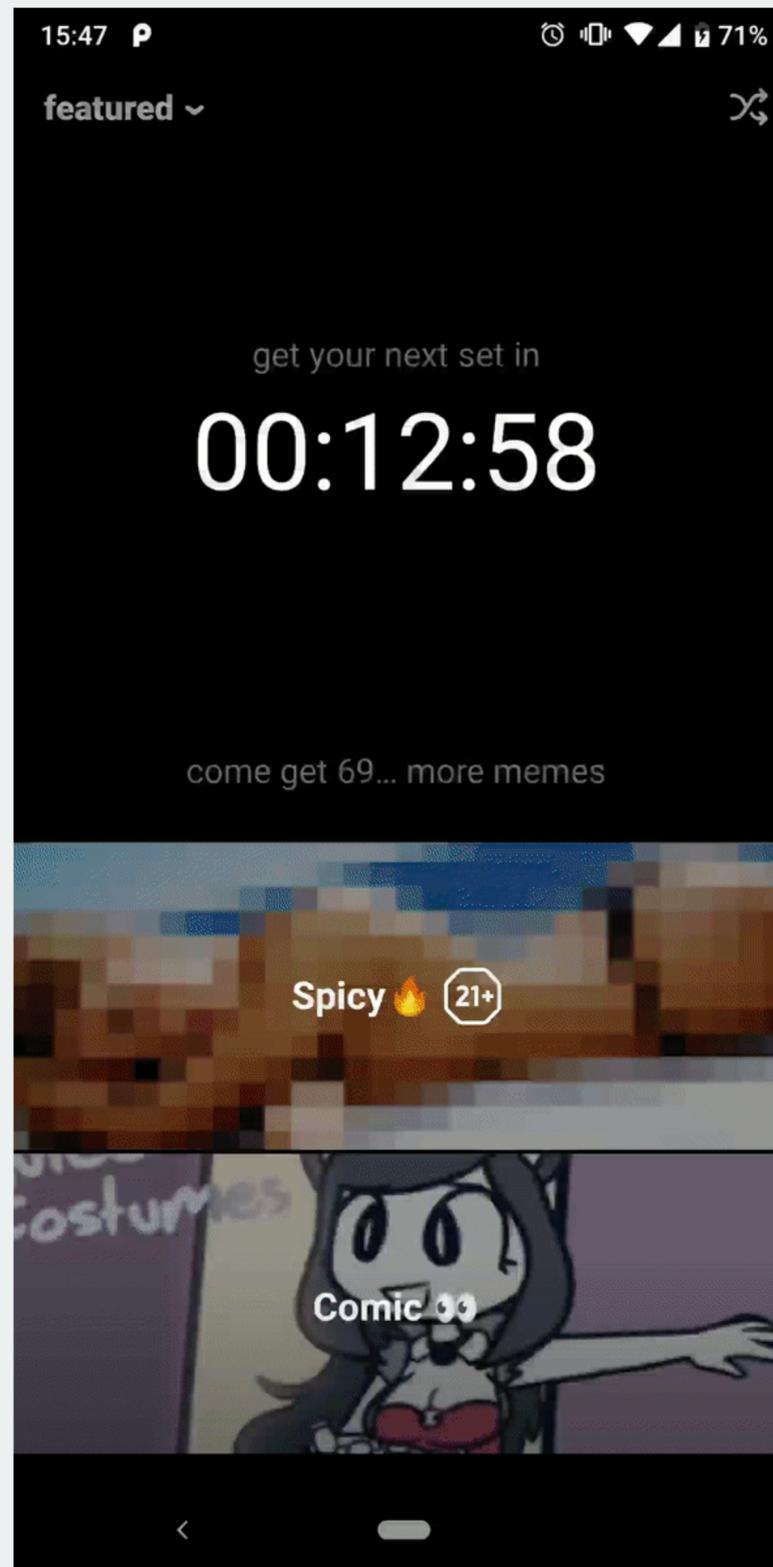
**и я ее придерживался!**



# Старый вариант префетчинга

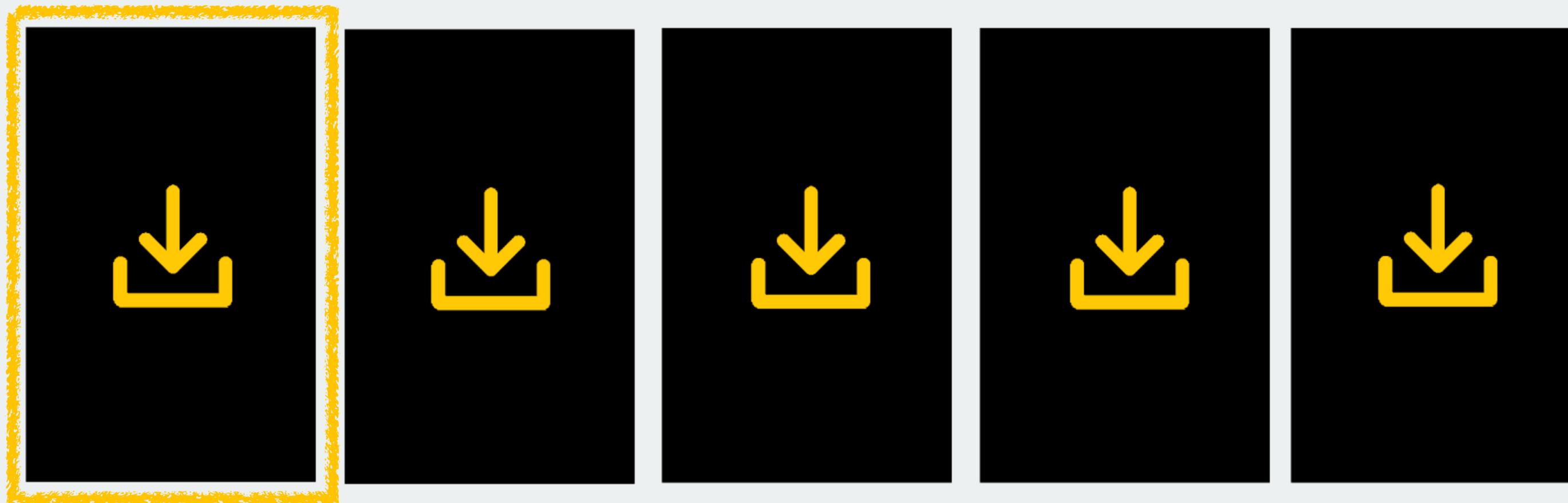


# Анализ ХИТмапа



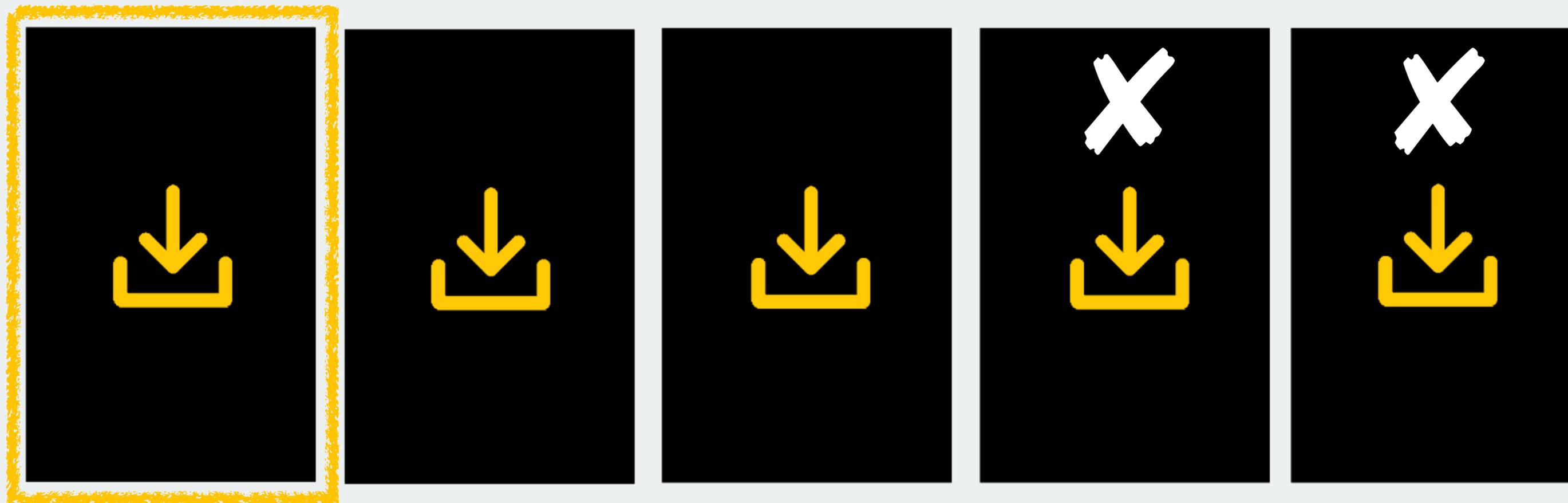


# Новый вариант префетчинга



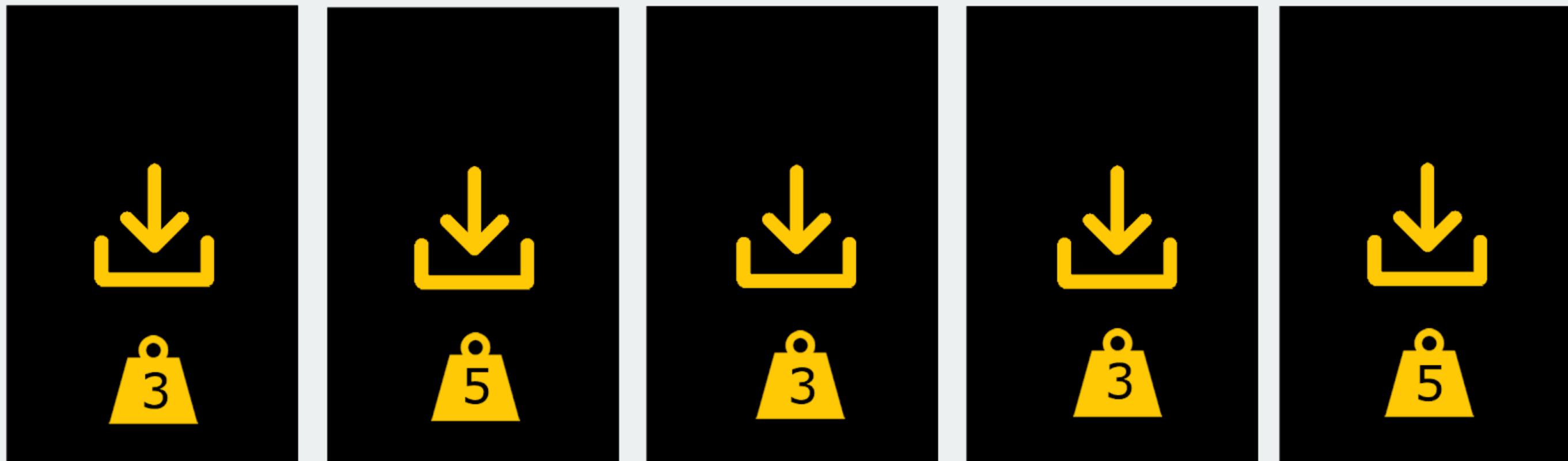


# Новый вариант префетчинга



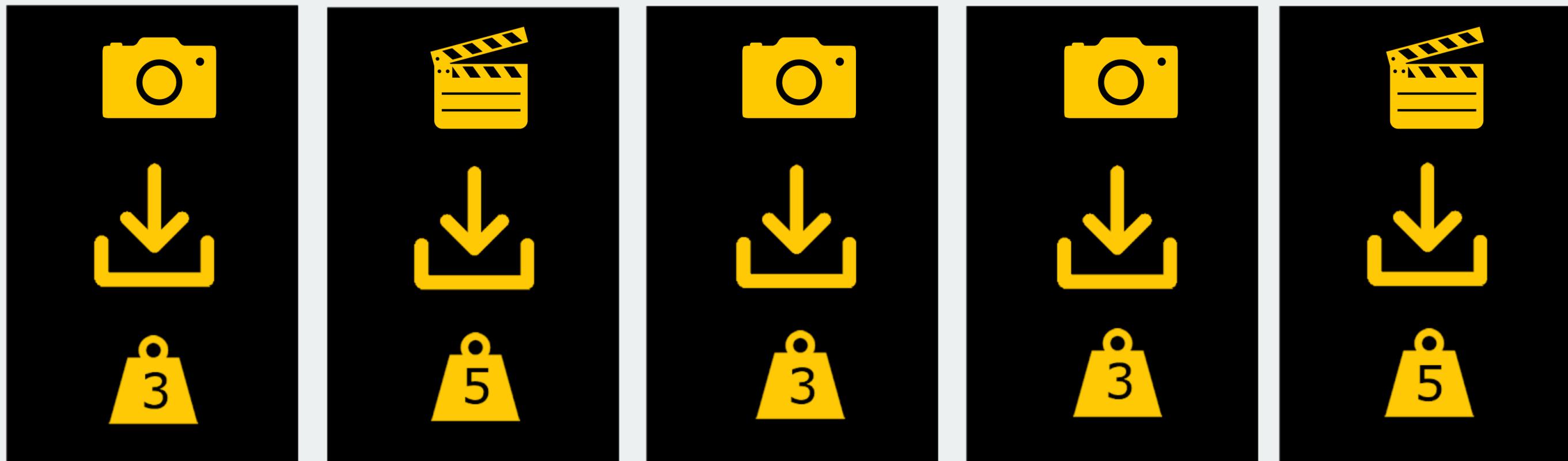


# Загрузка по весам





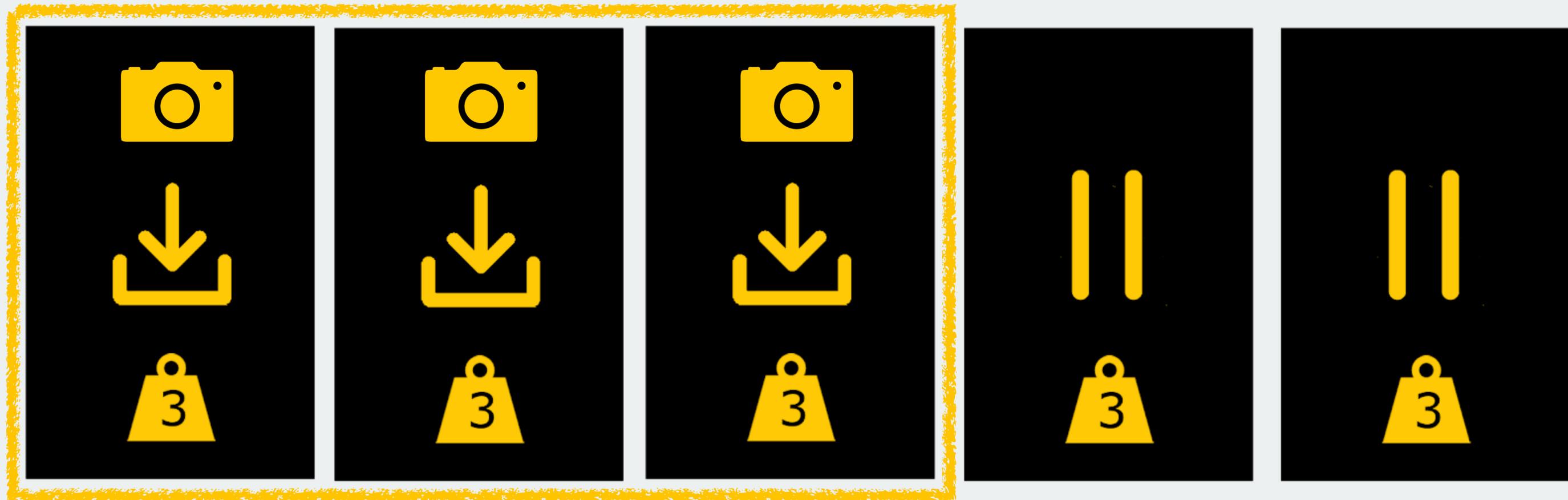
# Загрузка по весам



- Изображение - 3
- Видео - 5
- Максимальный вес - 11

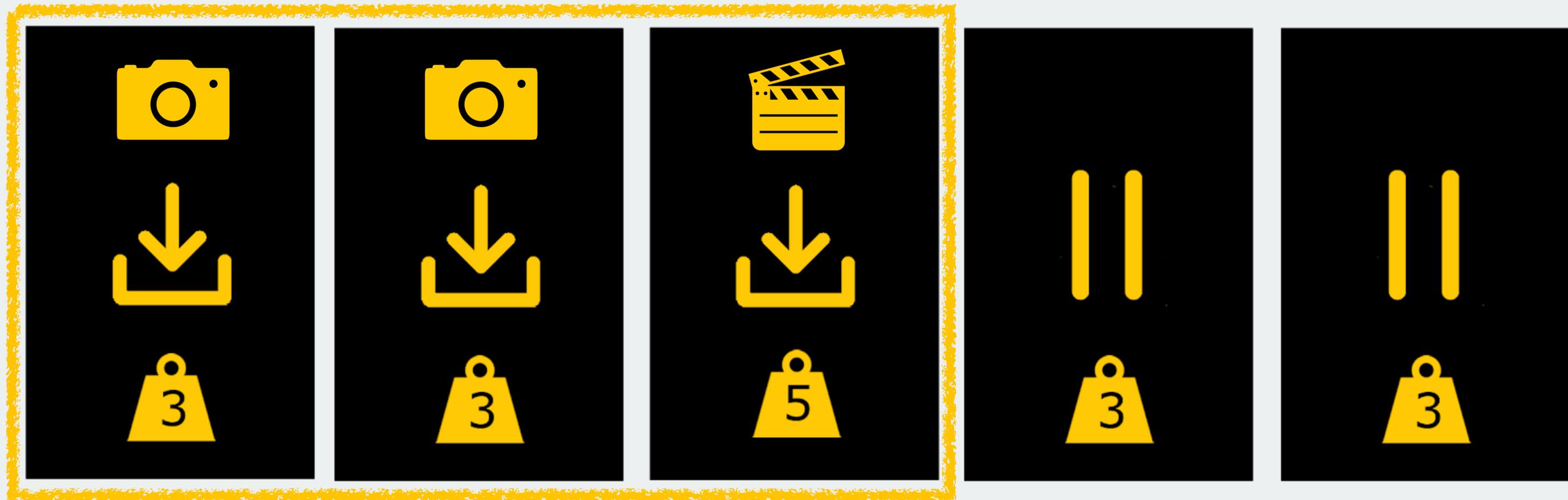


# Пример





# Пример

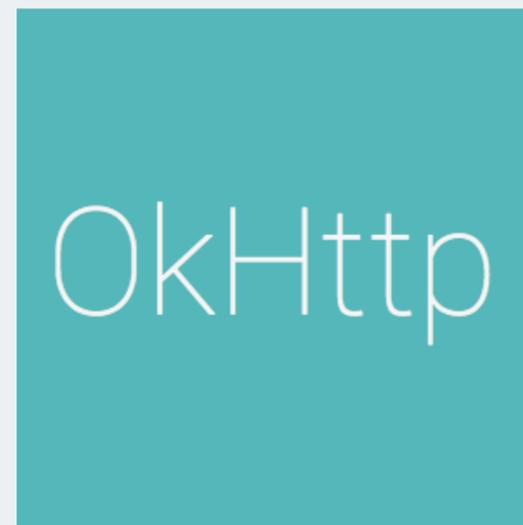




# Разные источники в старом подходе



glide





# Анализ нового подхода

- Избавление от фрагментации





# Анализ нового подхода

- Избавление от фрагментации
- Контролируемое дисковое пространство





# Анализ нового подхода

- Избавление от фрагментации
- Контролируемое дисковое пространство
- Переиспользование





# Анализ нового подхода

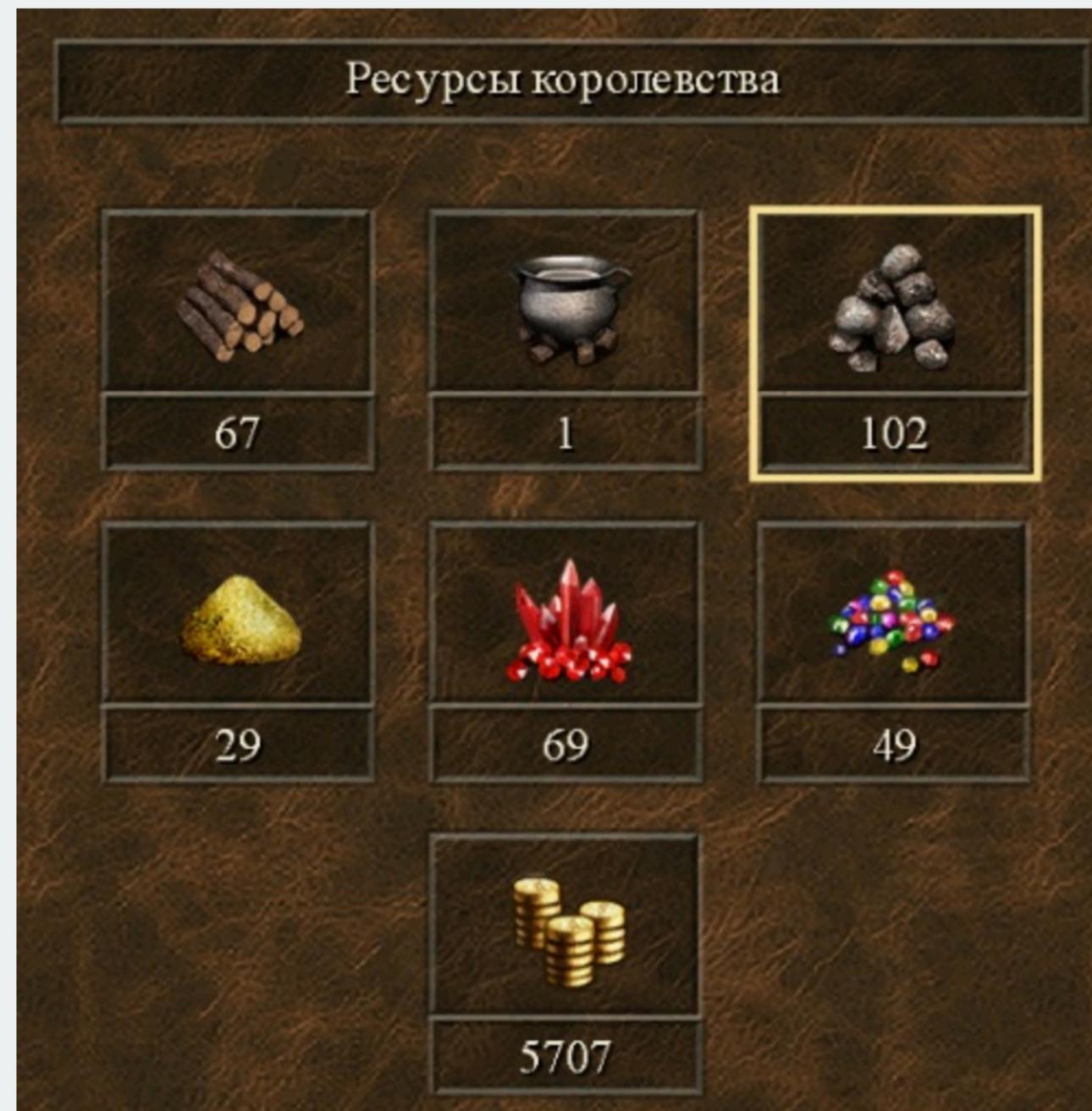
- Избавление от фрагментации
- Контролируемое дисковое пространство
- Переиспользование



- Дорогая реализация



# Ресурсов хватало



# Критерии для стратегии кэширования



- **Ограниченный размер кэша**

# **Критерии для стратегии кэширования**



- **Ограниченный размер кэша**
- **Не хардкорная в плане реализации**

# **Критерии для стратегии кэширования**



- **Ограниченный размер кэша**
- **Не хардкорная в плане реализации**
- **Должна легко интегрироваться в текущую систему загрузки**

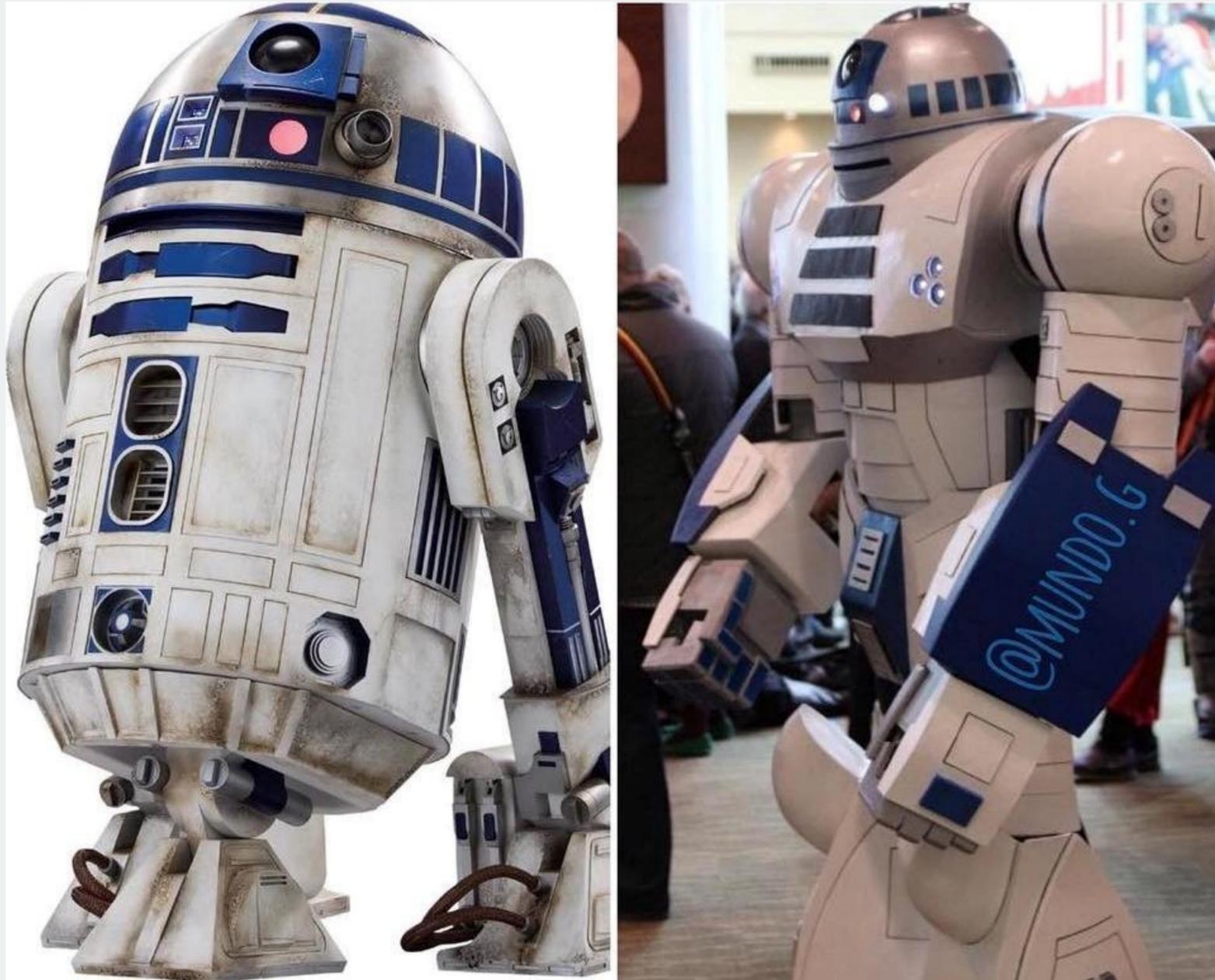
# **Критерии для стратегии кэширования**



- **Ограниченный размер кэша**
- **Не хардкорная в плане реализации**
- **Должна легко интегрироваться в  
текущую систему загрузки**
- **Конфигурируемая**



# Наша реализация



- Кэширование частично скачанных файлов



# Наша реализация



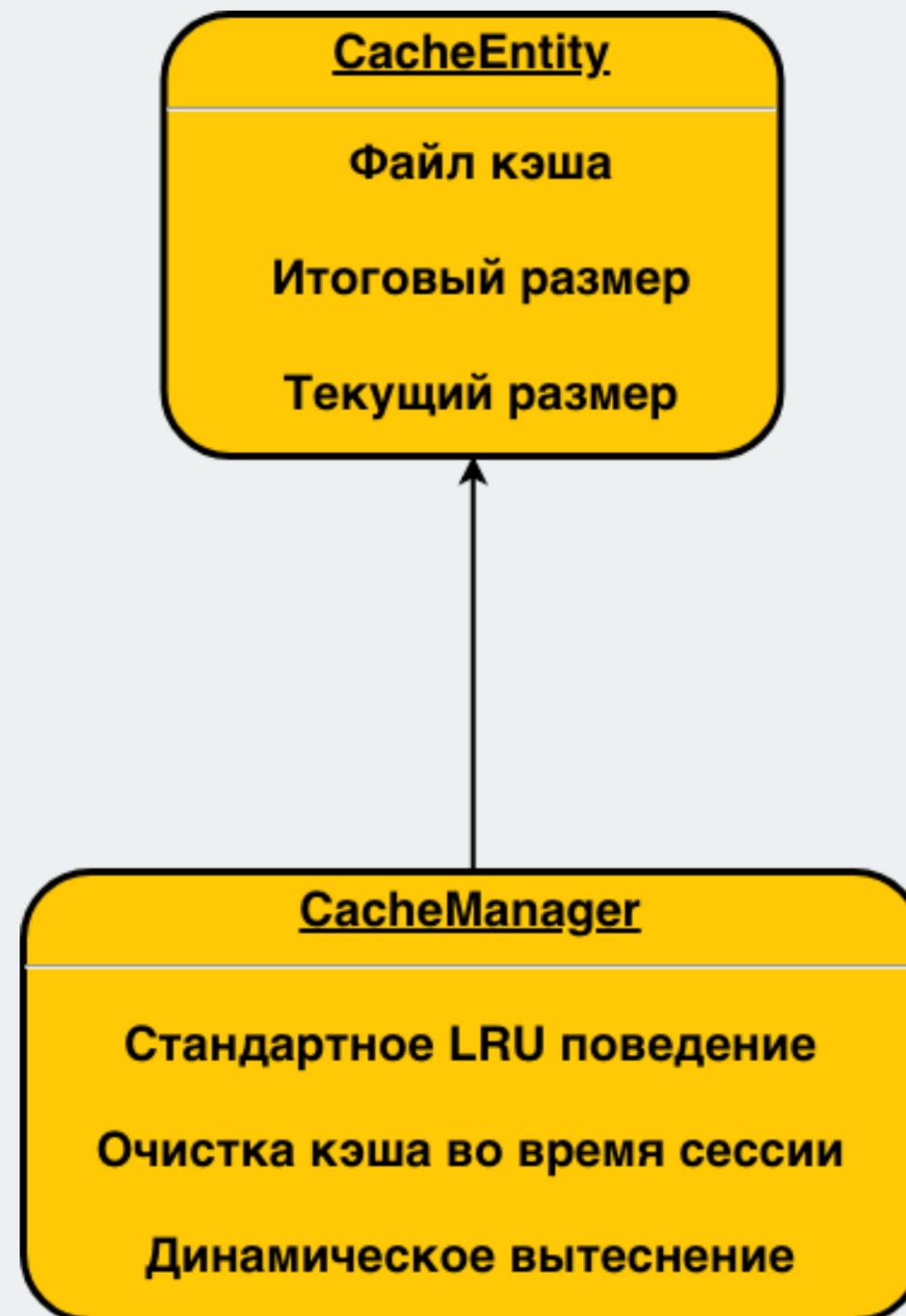
- Кэширование частично скачанных файлов
- Ограничение размера кэша в 100 Мб

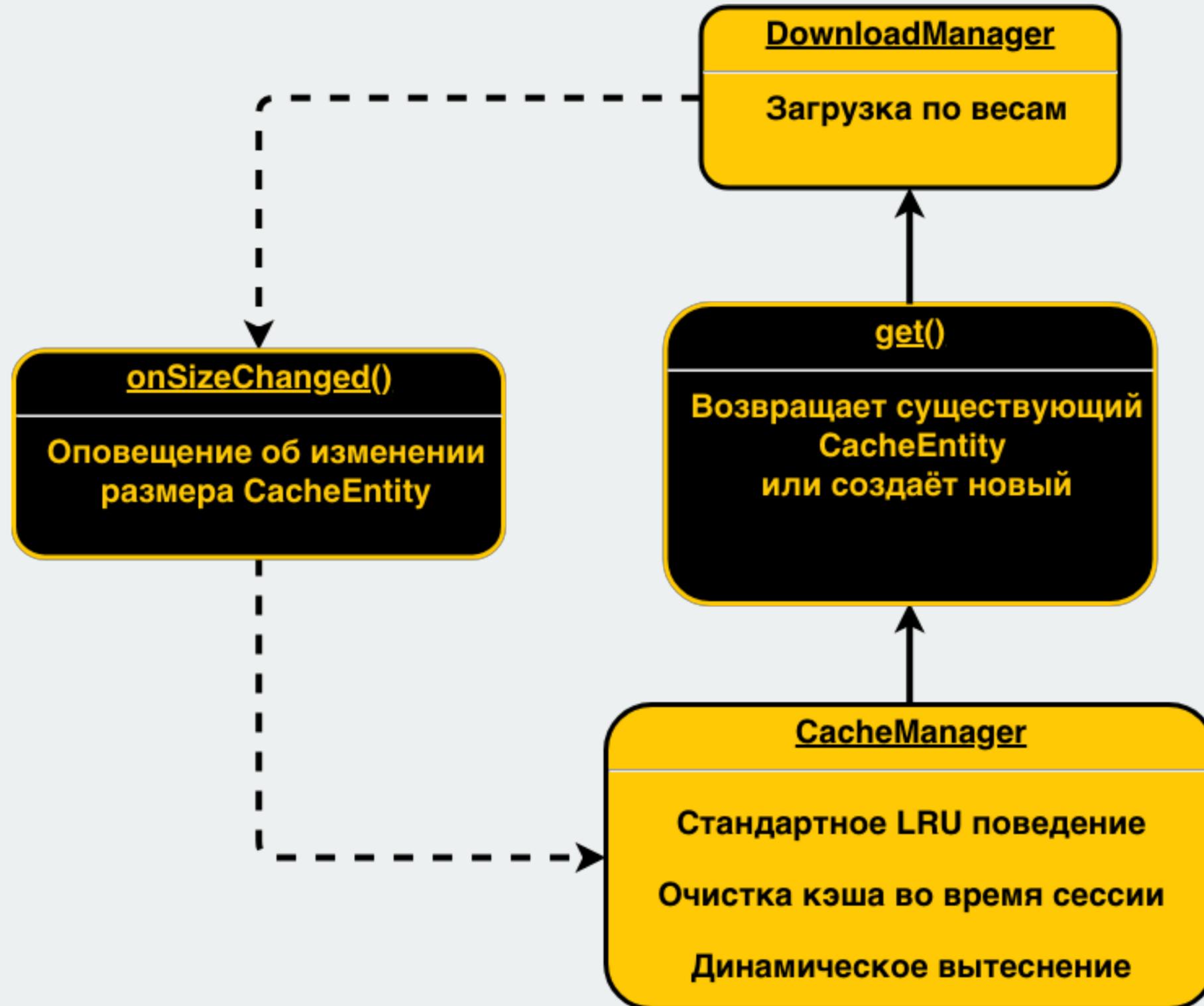


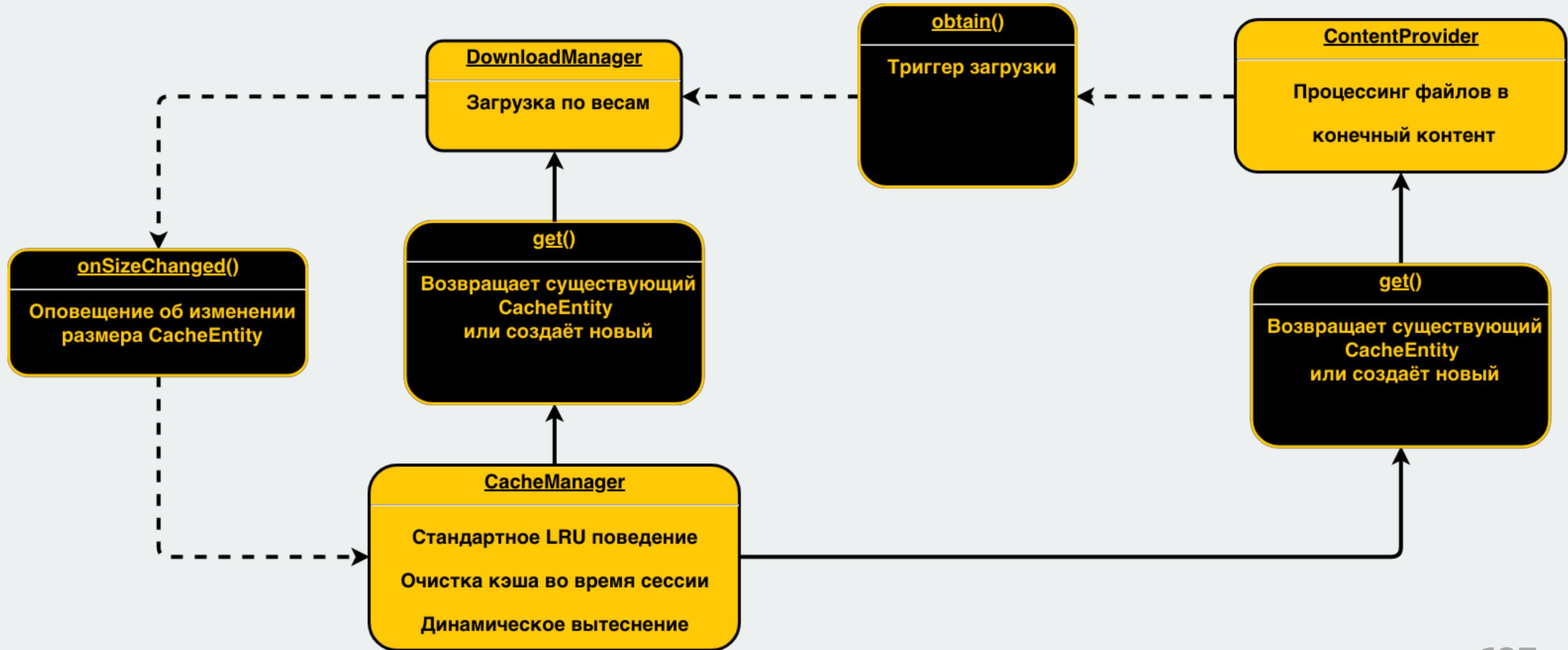
# Наша реализация

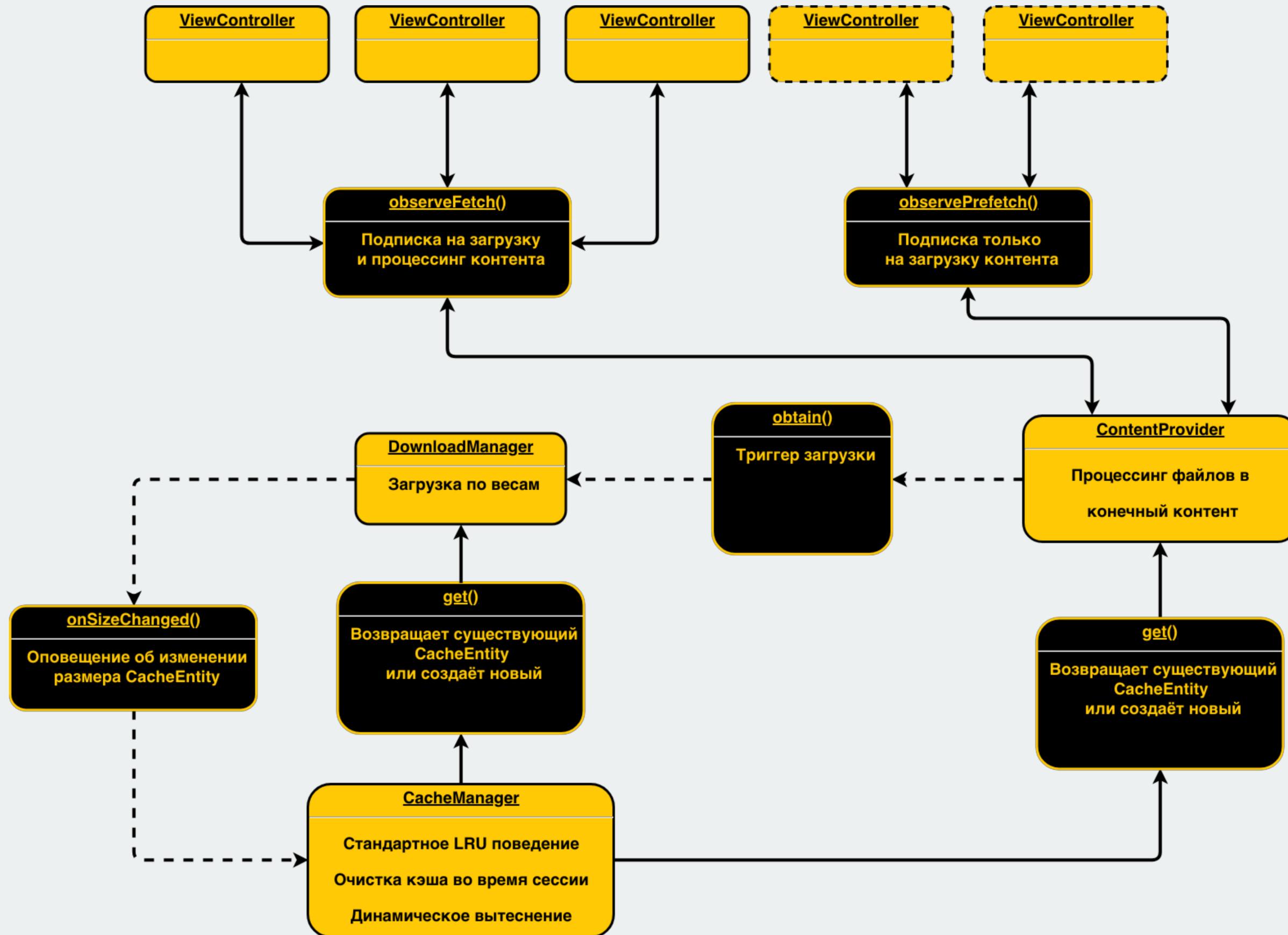


- Кэширование частично скачанных файлов
- Ограничение размера кэша в 100 Мб
- Очистка кэша после 5-ти минут бездействия в фоне



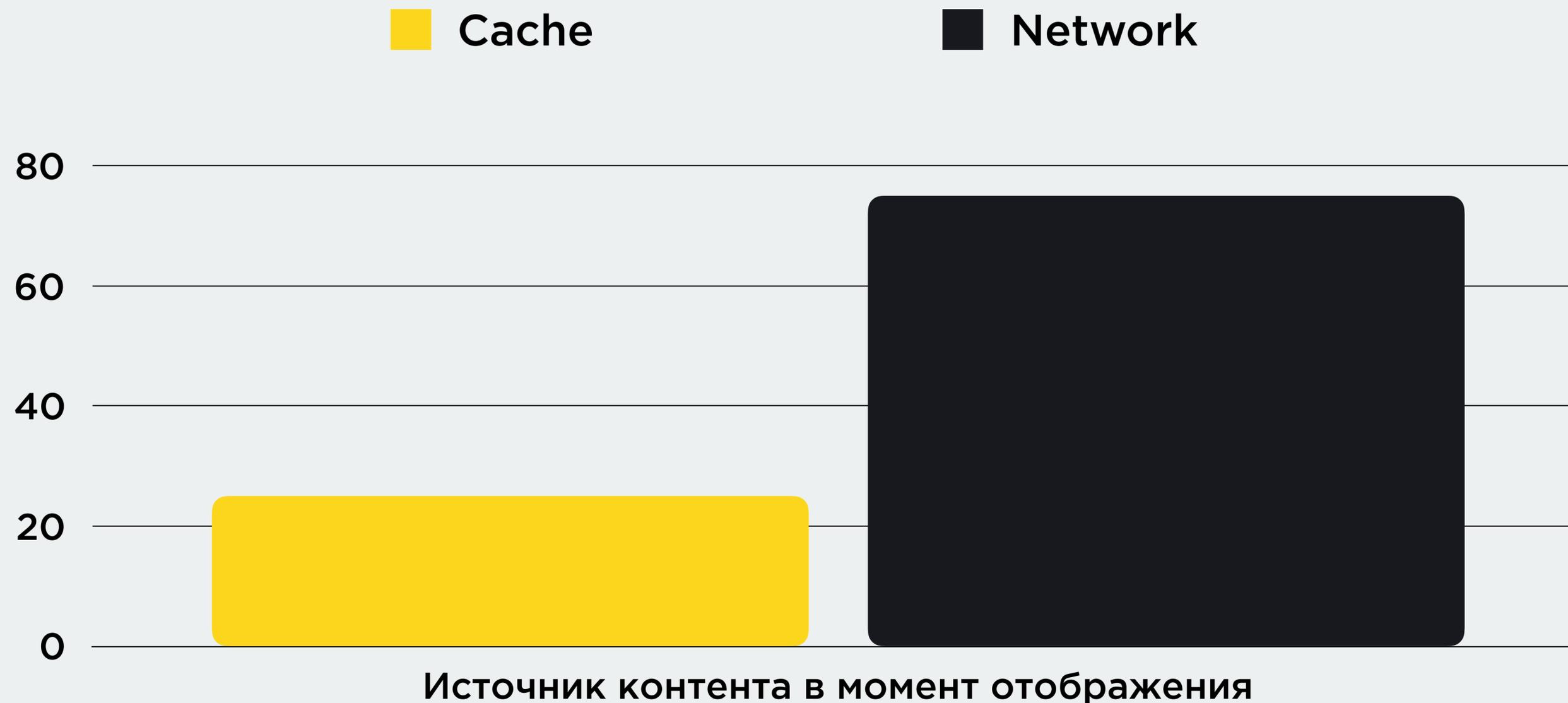






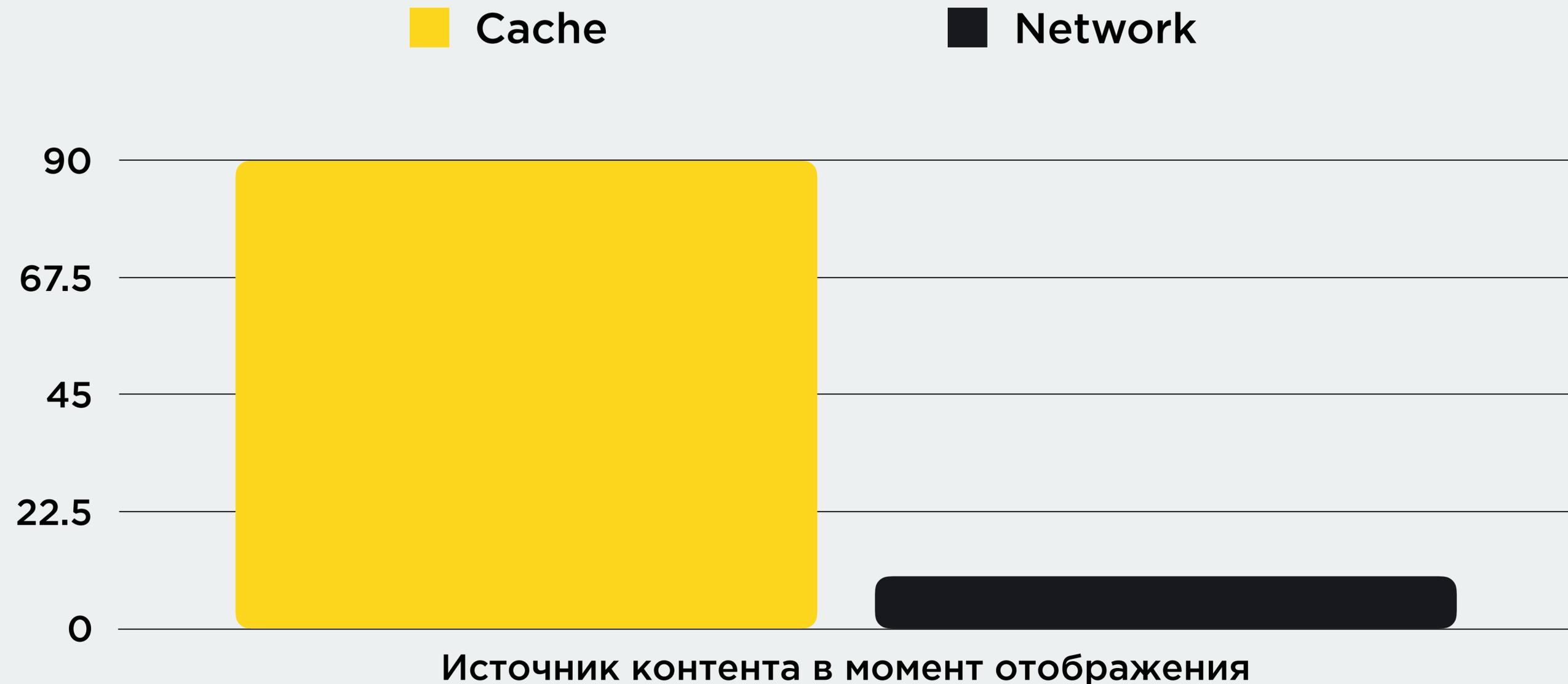


# Статистика до реализации





# Статистика после реализации

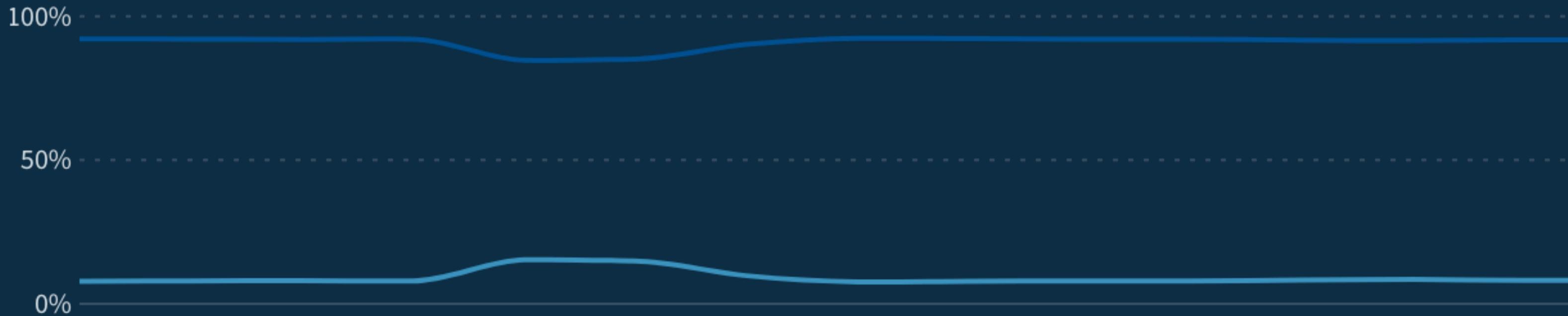


# Как это выглядит в Answers



From cache

CATEGORY ATTRIBUTE





# Статистика до реализации

**Progress Bar**

**1,5KK**

**Retry**

**1KK**

**Content**

**100KK**



# Progress Bar

**1,5KK**

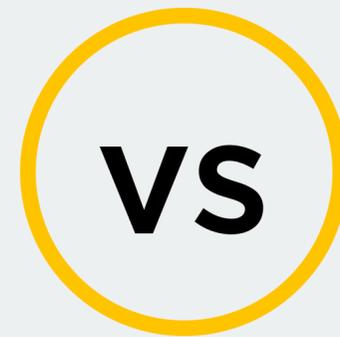


**0,5KK**



# Retry

**1KK**

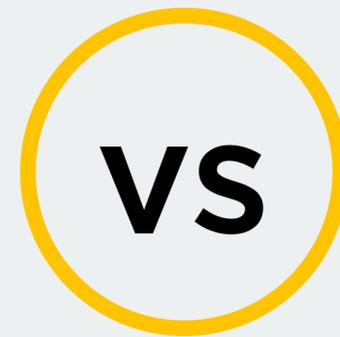


**0,4KK**



# Content

**100KK**



**130KK**



# Выводы

- **Убедитесь что вам действительно нужно кэширование**

# Из коробки



**OkHttp**

**cache-control**

**Glide**

**diskCacheStrategy**

**Picasso**

**networkPolicy**

**memoryPolicy**

**LruCache**

**android.util**

**DiskLruCache**

**<https://github.com/>**

**[JakeWharton/DiskLruCache](https://github.com/JakeWharton/DiskLruCache)**



# Выводы

- Убедитесь что вам действительно нужно кэширование
- Выберите правильную стратегию или коллаборацию стратегий



# Выводы

- Убедитесь что вам действительно нужно кэширование
- Выберите правильную стратегию или коллаборацию стратегий
- Не забывайте очищать/инвалидировать кэш



# Выводы

- Убедитесь что вам действительно нужно кэширование
- Выберите правильную стратегию или коллаборацию стратегий
- Не забывайте очищать/инвалидировать кэш
- Внедряйте итеративно



# Выводы

- Убедитесь что вам действительно нужно кэширование
- Выберите правильную стратегию или коллаборацию стратегий
- Не забывайте очищать/инвалидировать кэш
- Внедряйте итеративно
- Анализируйте эффективность использования



**smartdev.vdd@gmail.com**

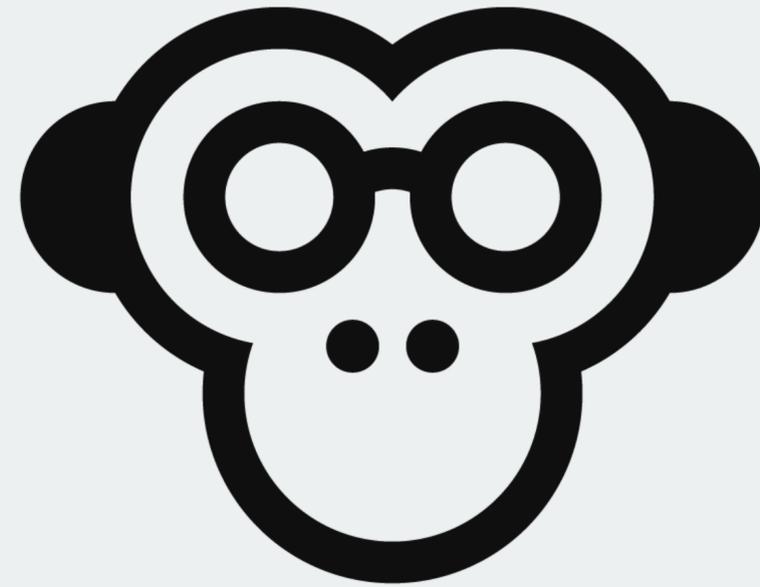


**@smartdev**



**@fraking\_vdg**





**FUNCCORP**

