@mail.ru
национальная почтовая служба

hl++

Быстрое развертывание шаблонов и статики в Mail.Ru

Кондратов Николай

Технический руководитель почтовой службы

О чем мы?

- система шаблонов
- организация работы верстальщиков
- процесс разработки и тестирования
- тестовая среда
- VCS
- развертывание

Хорошая система работы со статикой:

- удобство и скорость в работе
- версионность и бэкап
- параллельная работа верстальщиков
- независимая разработка фич
- параллельное тестирование
- независимое внедрение фич
- быстрое развертывание на серверах
- быстрый откат

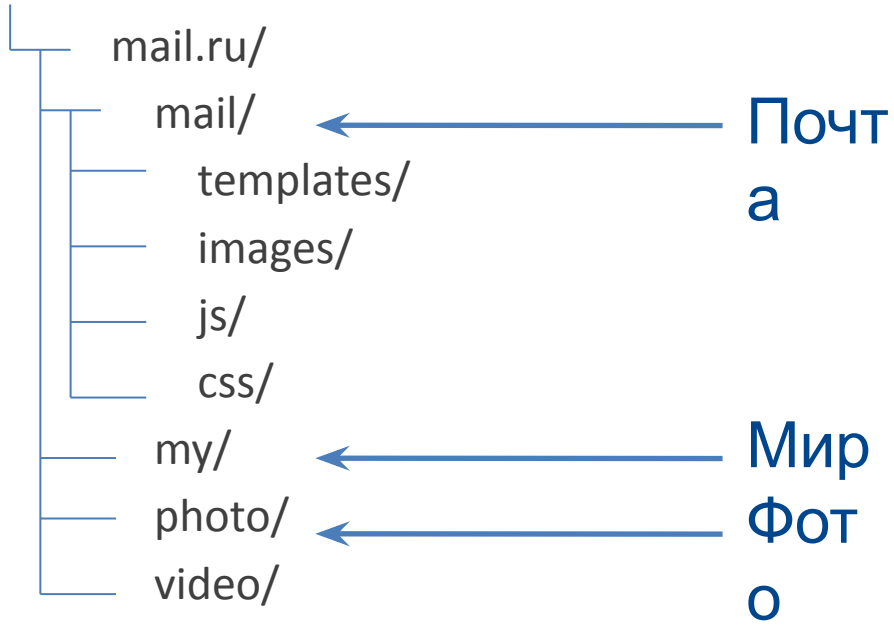
Система шаблонов

- HTML-текст с командами управления
- переменные, функции, инклуды, условия, etc.
- на фронтендах – компилированные
- одни шаблоны для Perl, C, Python

```
<html>
##SetVars(UserName=Вася)##
<!-- IF ActiveUser -->
<h1>Привет, ##UserName##!</h1>
<!-- /IF -->
</html>
```

Как это было: CVS

Схема репозитория



Что лежит:

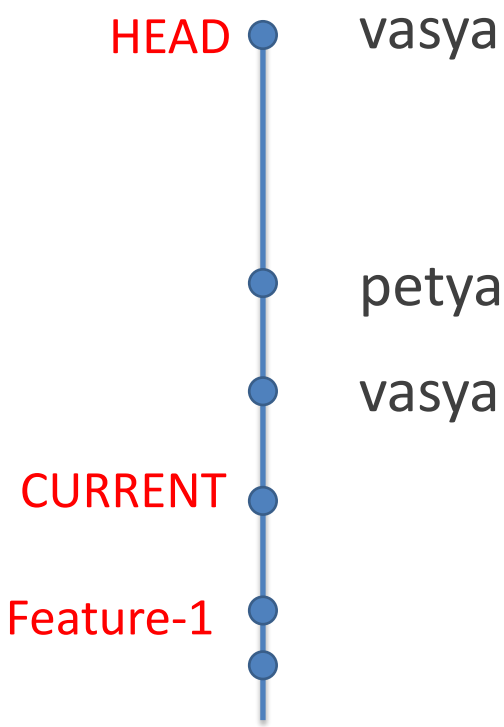
- шаблоны
- картинки
- swf
- бинарные файлы
- etc.

Как это было: работа верстальщика

- одна ветка в CVS
- теги для версионности

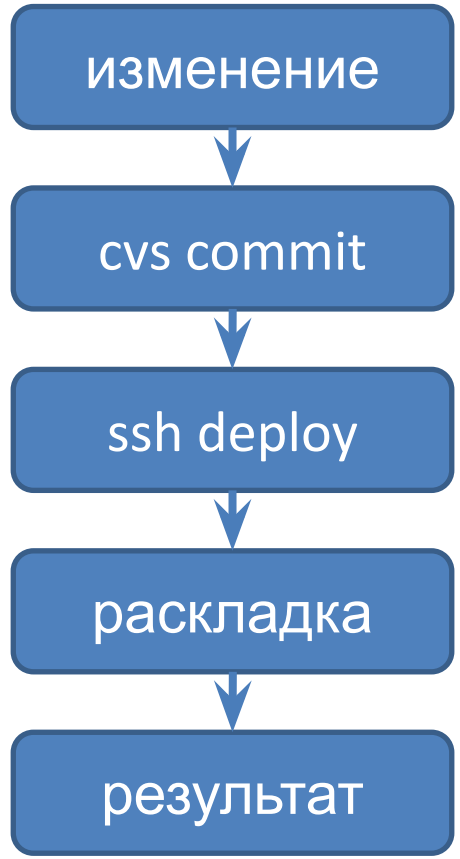
Проблемы

- сцепление версий



Как это было: работа верстальщика

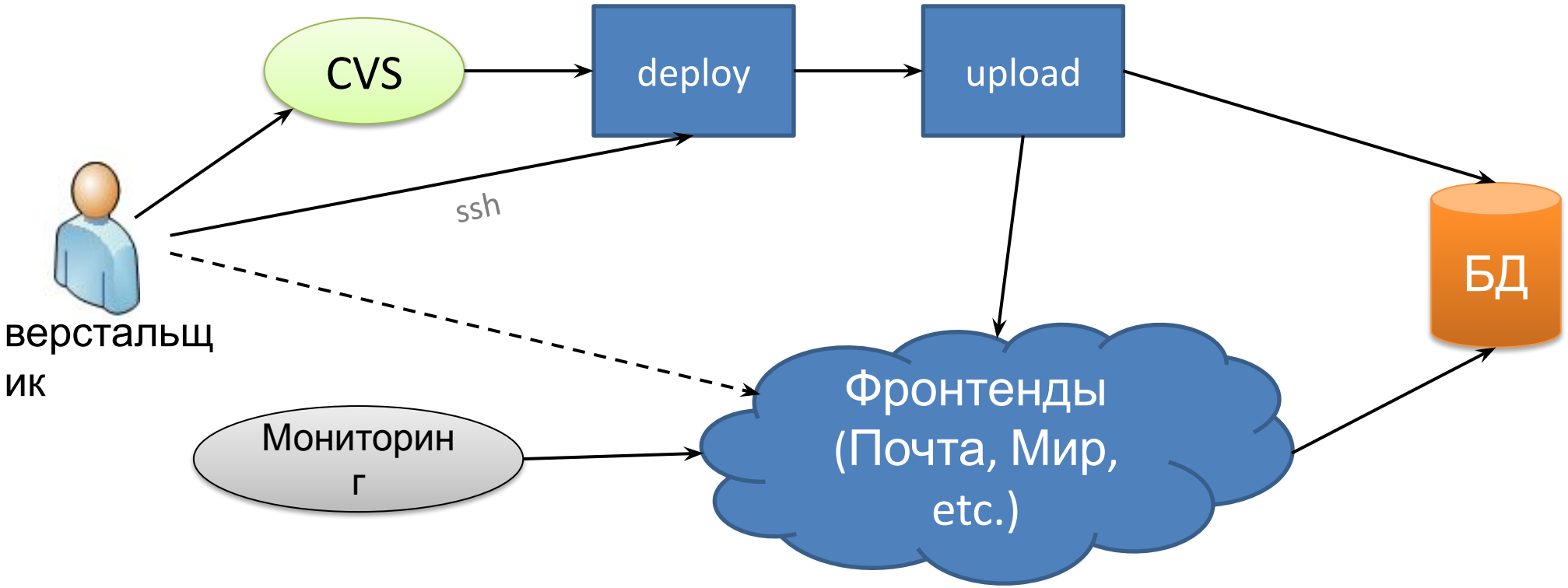
- любая раскладка – через репозиторий
- боевая и тестовая раскладка – по одной схеме



Проблемы

- долгое внесение изменений
- «мусорные» коммиты

Как это было: раскладка



Как это было: минусы

- долгий процесс разработки
- шаблоны всех проектов на всех фронтендах
- долгое распространение шаблонов

Как это было: минусы CVS

- размер репозитория – 2,8 Гб, 55 тыс. файлов
- медленный
- тяжелые ветки
- плохой мердж

Что сделали

- перевод репозитория CVS git
- разделение репозиториев
- использование веток
- реорганизация тестового окружения
- изменение схемы работы верстальщиков
- дополнительная логика при раскладке шаблонов

Ускорение разработки.
Как?

Переход на git – разделение репозиториев

- разделили репозитории (статика и шаблоны)
- git с шаблонами – 600 Мб (около 16 тыс. файлов)
- git со статикой – 3.2 Гб (около 38 тыс. файлов)

Статика

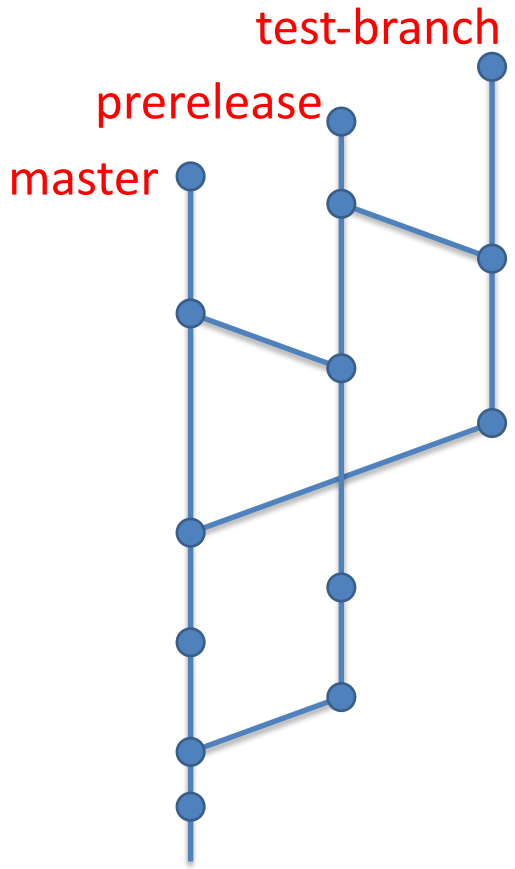
- картинки
- бинарные данные

Шаблоны

- шаблоны
- CSS
- JS

Переход на git – разделение на ветки

- разделили на ветки:
  - master
  - prerelease
  - ветки разработки



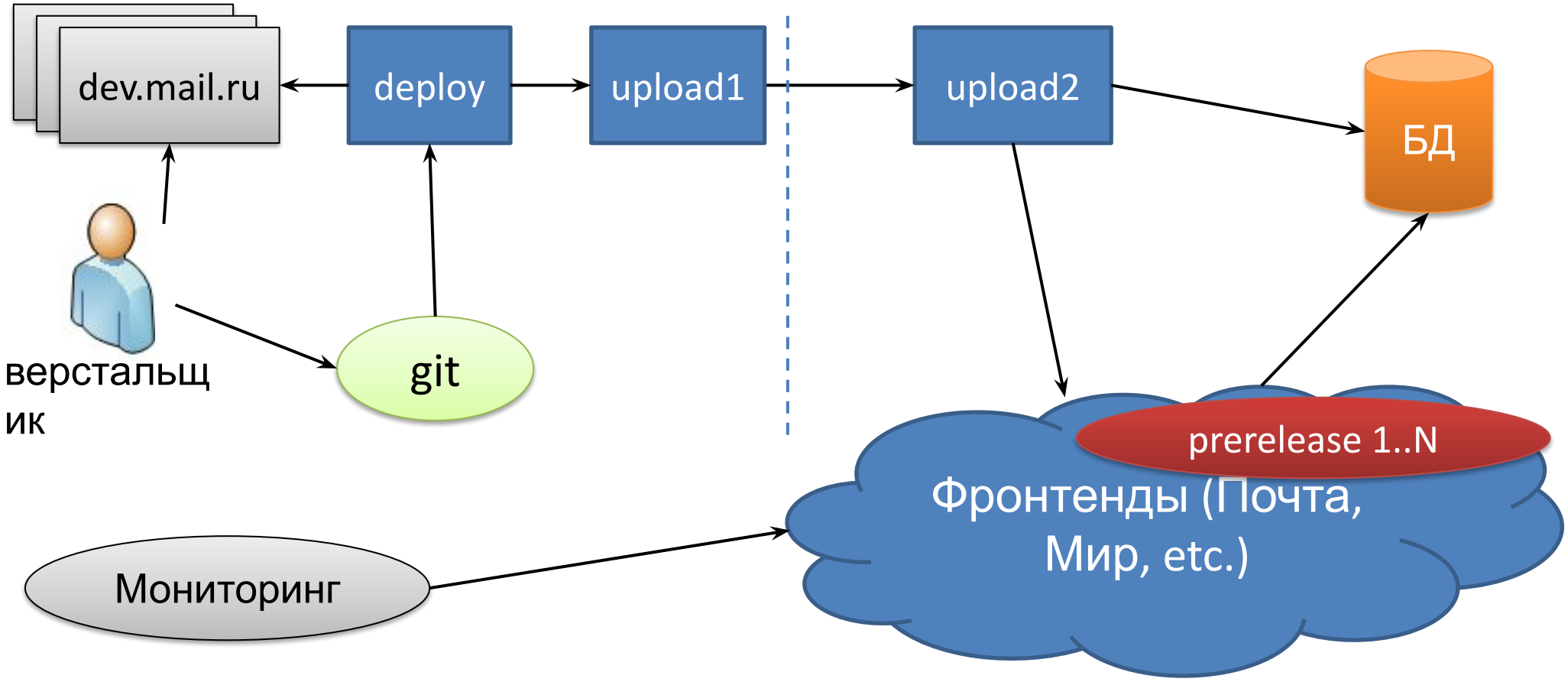
Переход на git - хуки

- hooks:
  - проверка синтаксиса шаблонов
  - обработка шаблонов
  - автоматическая раскладка
  - etc.

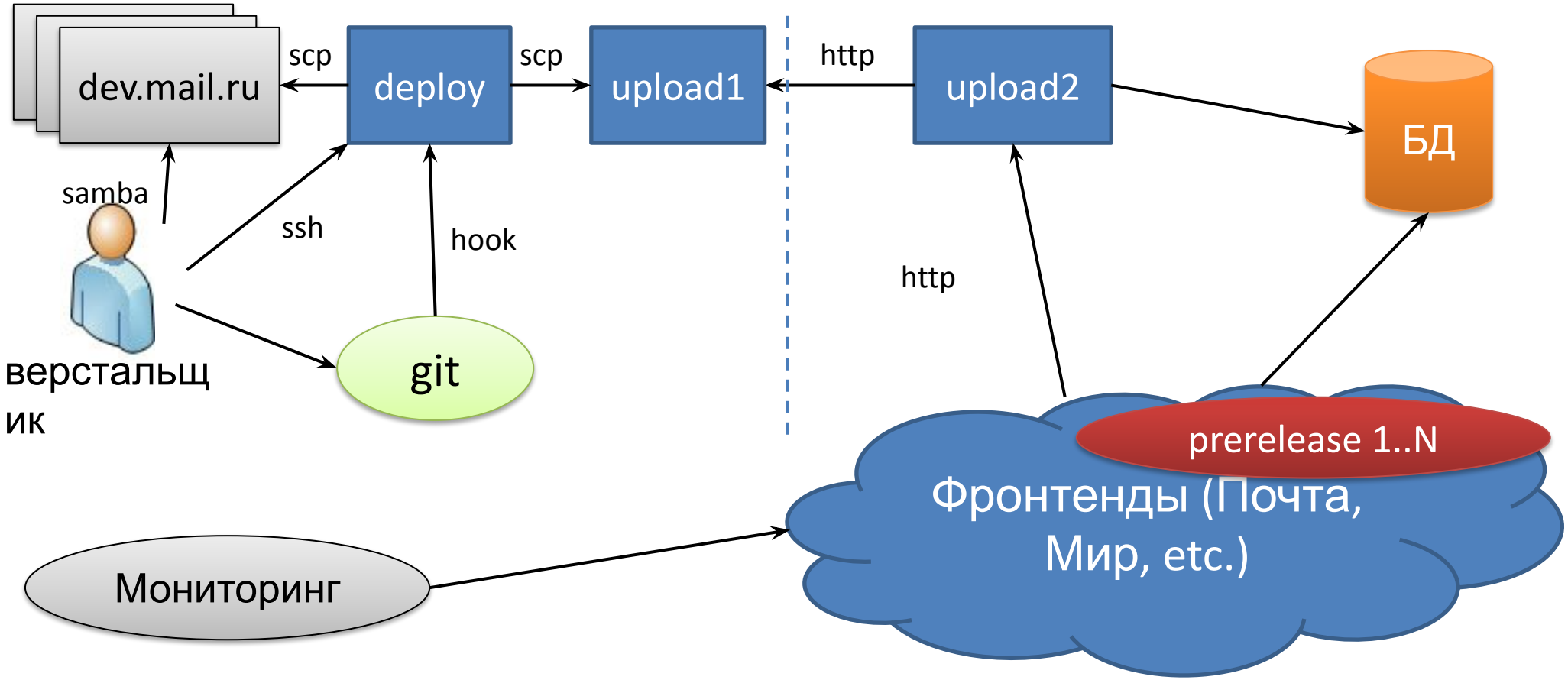
Процесс разработки

1. создаем ветку на основе master
2. вносим изменения
3. коммитим, тестируем
4. переносим изменения из ветки в prerelease
5. тестируем prerelease
6. переносим изменения из prerelease в master
7. раскладываем

Общая схема (потоки данных)



Общая схема (действия)

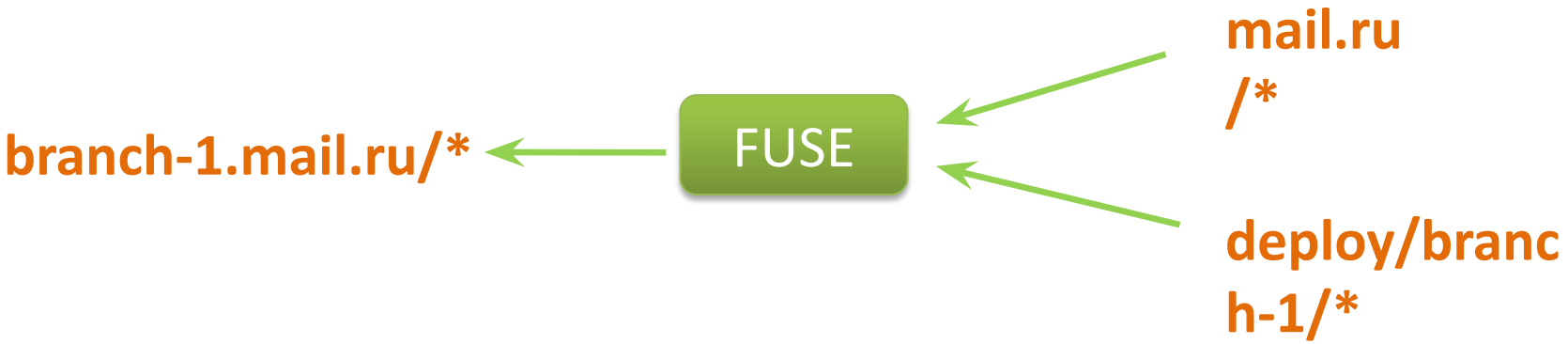


Тестовые серверы

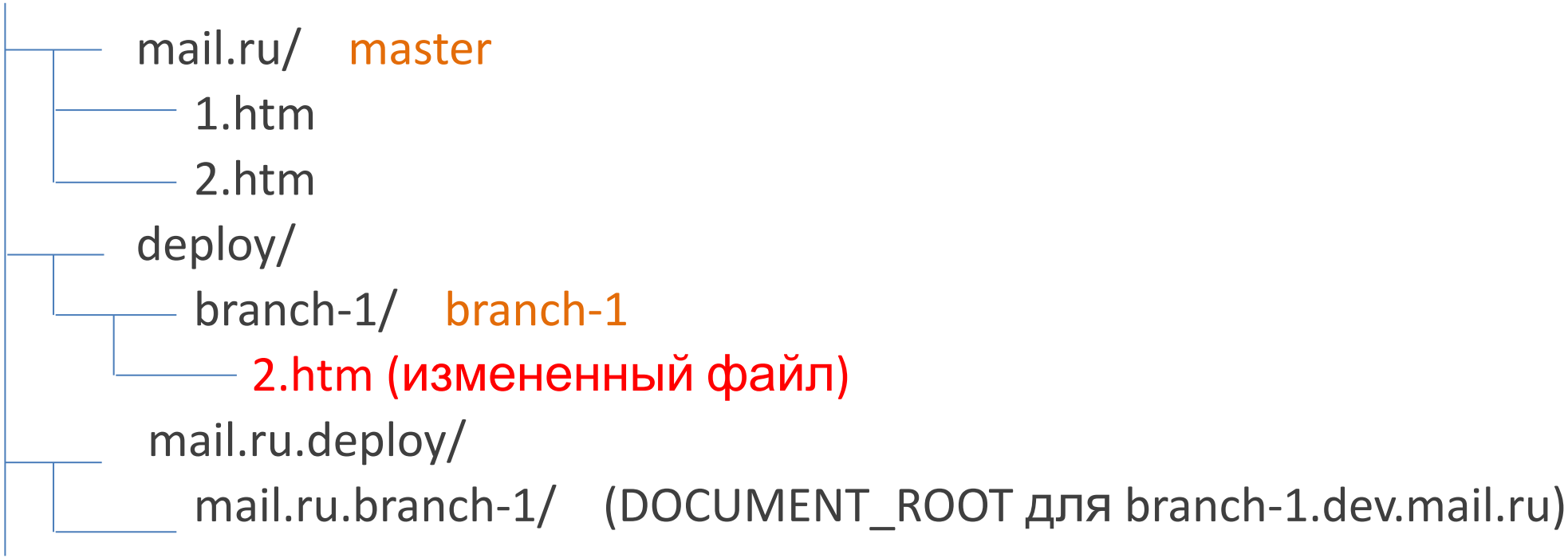
- DOCUMENT_ROOT сервера – директория с шаблонами
- dev.mail.ru – master
- branch-1.dev.mail.ru – ветка branch-1
- виртуальные машины с общим диском
- размер каждой директории – около 200 Мб
- количество веток – около 500
- для полного хранения – 100 Гб

FUSE (Filesystem on Userspace)

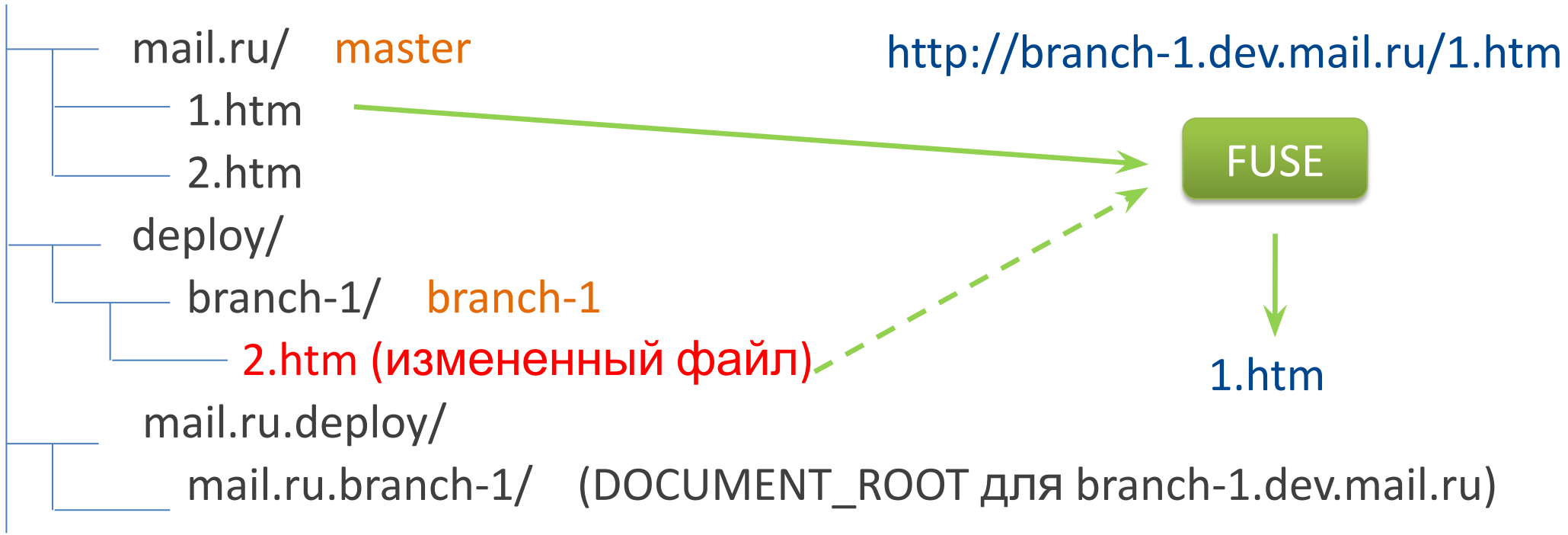
- храним только отличающиеся файлы
- отключаем компиляцию шаблонов
- недостающие файлы в ветке берем из master
- итоговый размер **5 Гб**



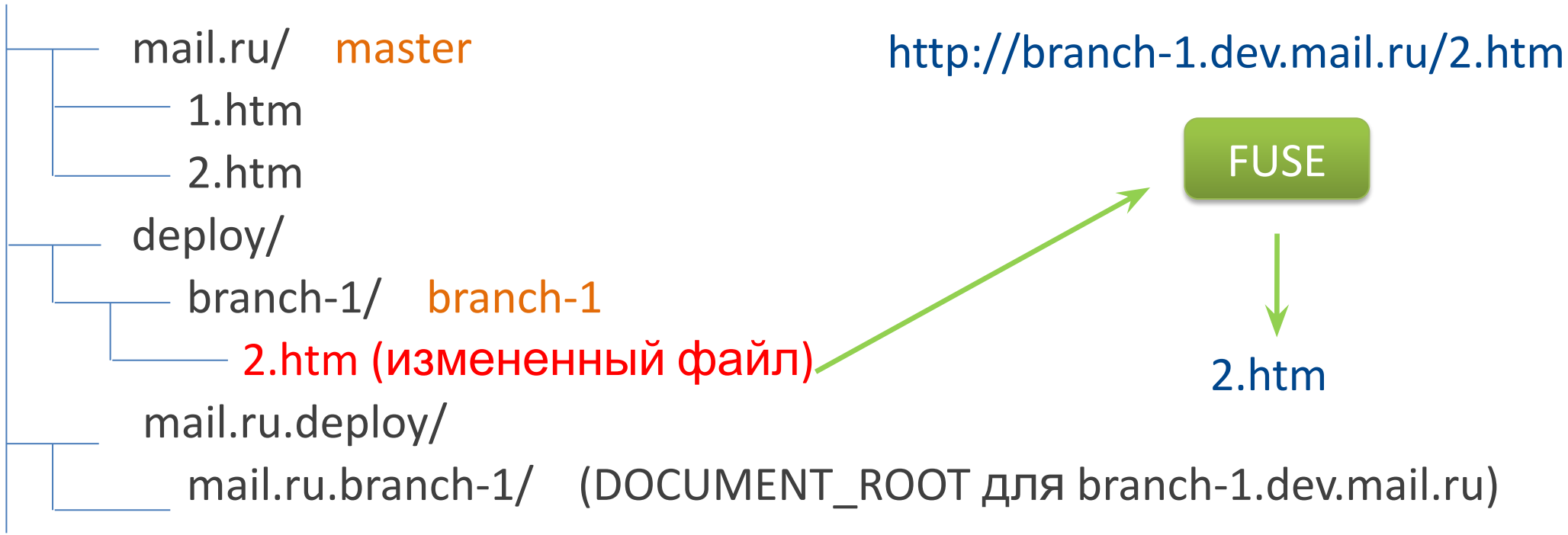
FUSE (Filesystem on Userspace)



FUSE (Filesystem on Userspace)

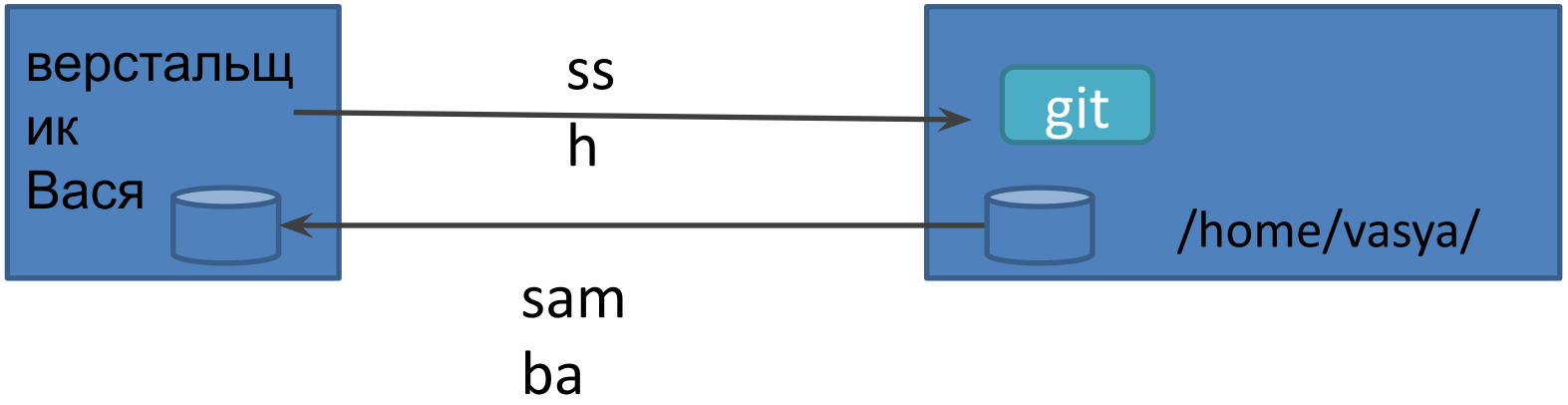


FUSE (Filesystem on Userspace)



Проблемы

- git + windows
- длинная цепочка действий для раскладки
- принцип «сохранил – увидел»
- решение – Samba + перенос репозитория



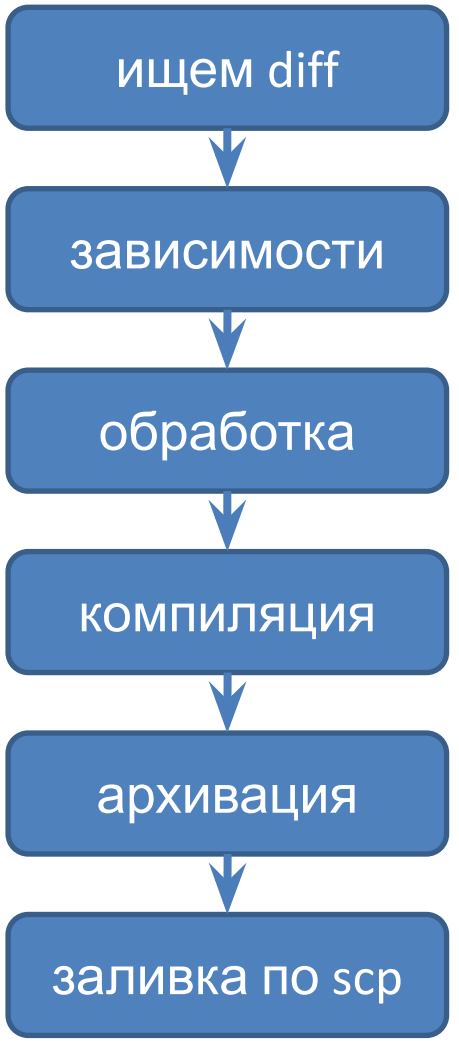
Samba

Процесс:

1. клонирование репозитория в /home/vasya
2. редактирование через Samba
3. проверка на vasya.dev.mail.ru
4. новая ветка (на основе master)
5. push
6. раскладка по хуку на тестовый сервер
7. мердж в prerelese
8. раскладка по хуку
9. мердж в master
10. ручная раскладка на живые

Раскладчик

- модульность
- проверка шаблонов
- различная обработка разных типов файлов
- компиляция шаблонов



Раскладчик – вкусности

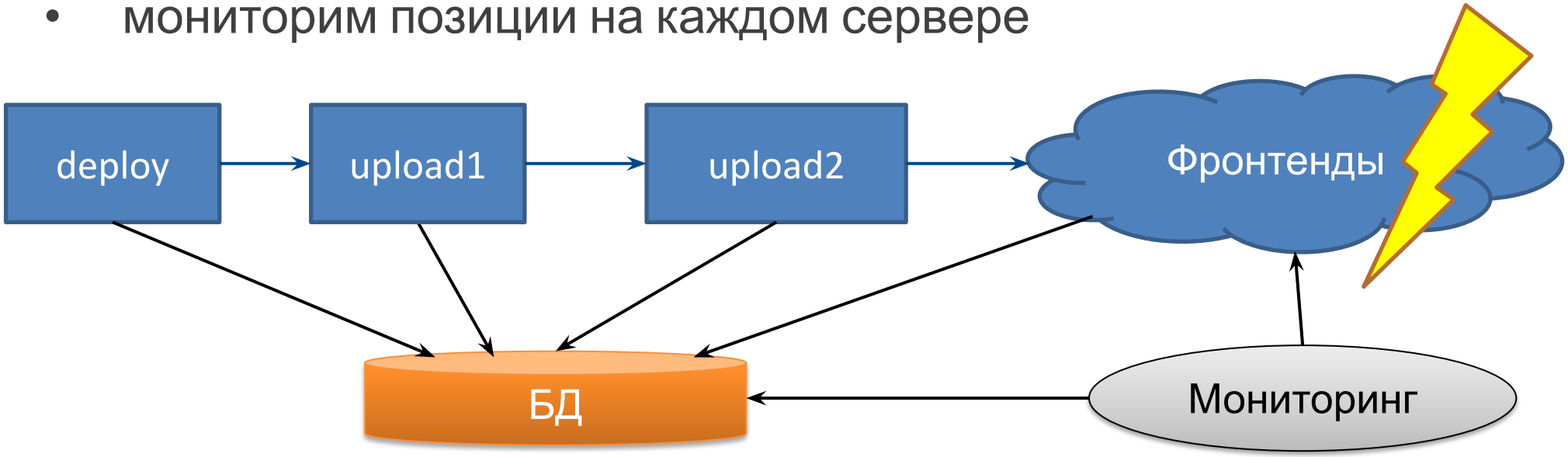
- условная обработка файла в зависимости от меток в нем
- JS
  - сборка
  - минимизация
  - именование файлов
- Шаблон
  - переводы строк
  - перекодировка
- и любые глупости

Раскладчик – пример метки

- метка в первой строке файла
- JS
  - `// @build`
  - `// @build-minify`
- Шаблон
  - `<!-- build-strip-newlines -->`

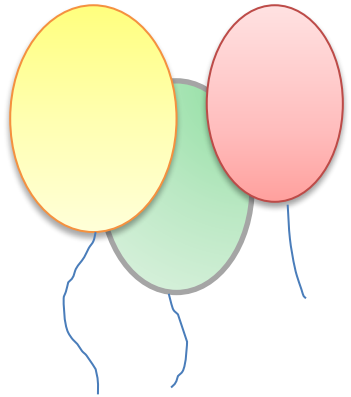
Мониторинг раскладки

- инкрементный номер каждой раскладки
- в БД – позиция на каждом сервере
- мониторим состояние раскладчика
- мониторим позиции на каждом сервере



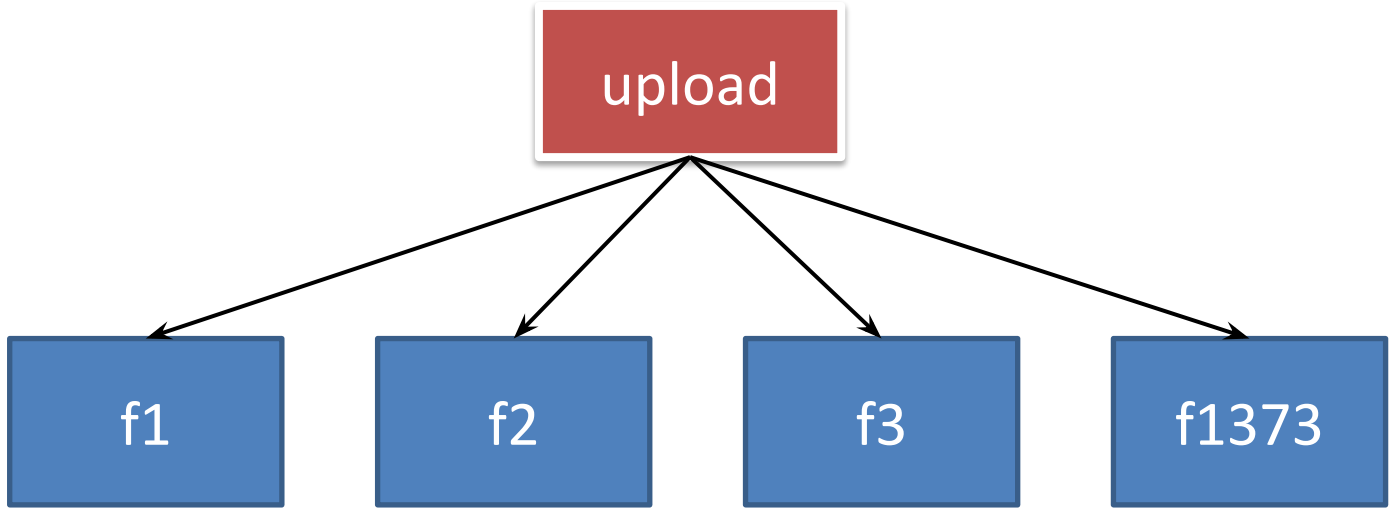
Результаты

- упростили жизнь верстальщикам
- возможность параллельной разработки фич
- ускорение отладки и тестирования
- улучшение работы с репозиторием



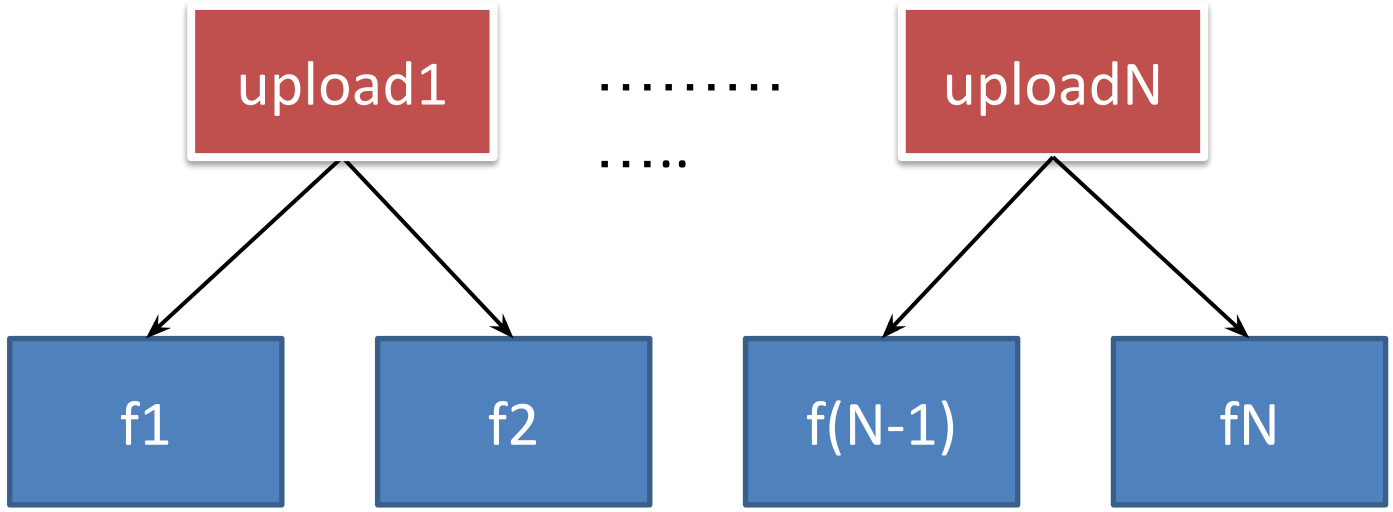
Но есть проблема…

- узкое место – получение архивов с шаблонами из одного источника



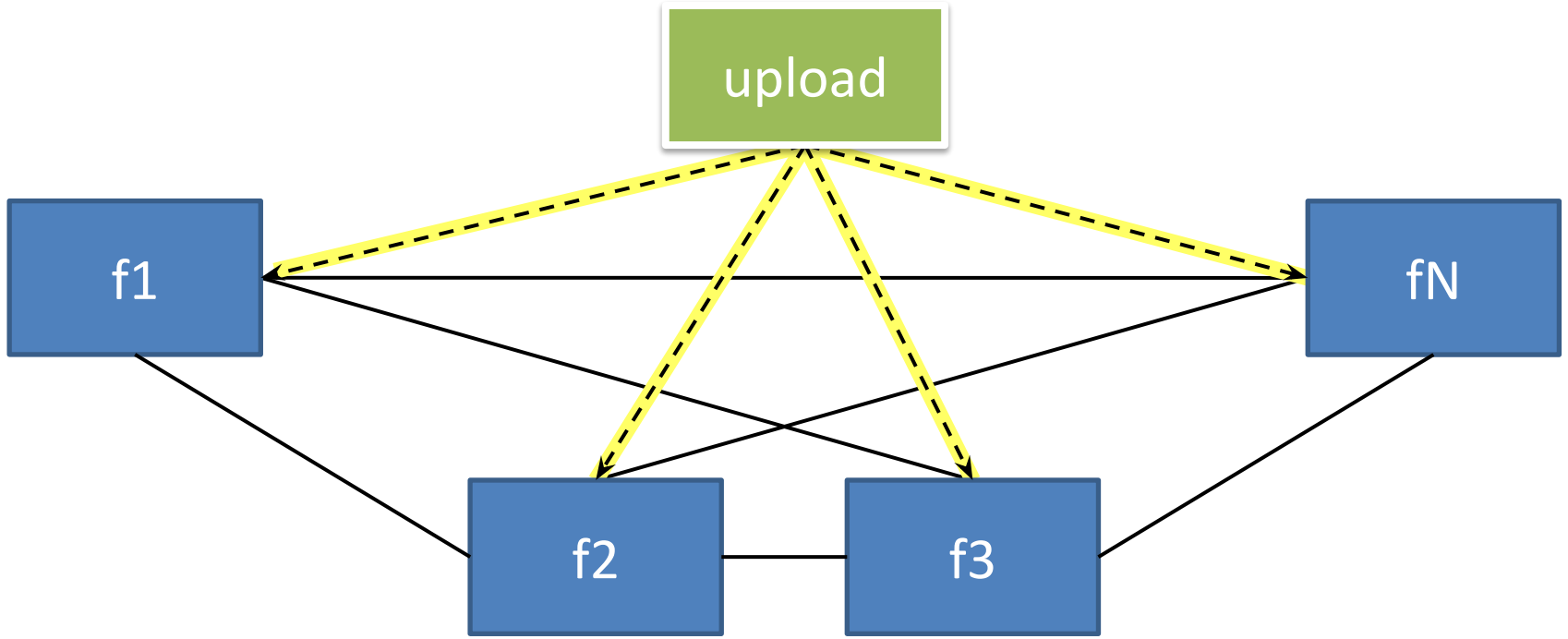
Простое решение… шардинг

- увеличить количество отдающих серверов



Хорошее решение: peer-to-peer

- с upload-сервера отдача в режиме супер-сидирования
- фронтенды обмениваются между собой



Спасибо. Вопросы?

kondratov@corp.mail.ru