Системи за управление на версии

Работа с git

Иван Георгиев, Христо Иванов, Христо Стефанов

Технологическо училище "Електронни системи",
Технически университет, София

28 февруари 2019 г.

Съдържание

Съдържание

Управляване на версии

- При работа в екип всеки програмист работи върху свое копие на програмния код
- Когато някой оправи проблем или добави нова функционалност, той трябва по някакъв начин да даде своите промени на другите хора от екипа за да може всички отново да работят върху един и същ програмен код
- Казано по друг начин, програмистът трябва да сподели своята *версия* на програмния код (наречена още *ревизия*) с останалите хора от екипа
- Когато разпространяването на ревизии се прави ръчно чрез копиране на файлове, често се допускат грешки (напр. забравен файл, презаписване на файл с вече направени промени от някой друг). Също така, причината за новата ревизия се налага да се предава устно.

Управляване на версии

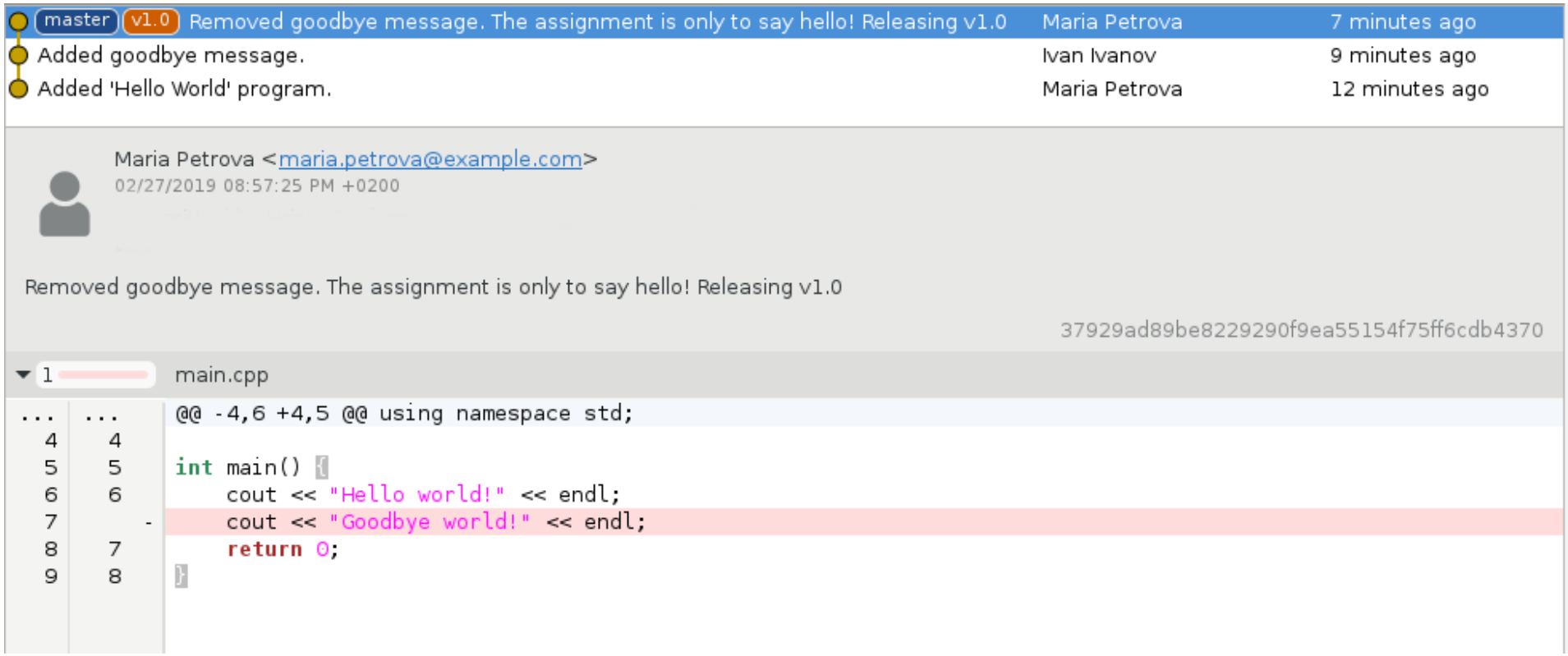
- При разработването на софтуерен продукт, често се налага едновременното поддържане на няколко *публични версии* (напр. Windows 7, Windows 8.1, Windows 10)
- За да може да бъде оправен проблем или да се добави функционалност в някоя публична версия е нужно да се намери копие на програмния код за нея и да се направят промени. Това води до създаването както на нова публична версия, така и до ново копие на програмния код ($1.0 \rightarrow 1.1$)
- Става изключително важно да се знае във всяка една публична версия какви промени са направени и кой е съответният програмен код, за да може софтуерният продукт да бъде поддържан

Съдържание

Системи за управление на версии

- *Системите за управление на версии* целят да улеснят разпространяването на ревизии на програмния код в екипа и да поддържат точна история на промените в проекта
  - За *публична версия* (или *версия*) се приема някоя ревизия на програмния код, която се разпространява до крайни потребители
  - други имена: система за управление на програмен код, система за контрол на ревизии/версии, copc-контрол система
- Най-често системите за управление на версии работят като съхраняват всяка една ревизия, заедно с описание на промените в нея, в *хранилище* (*repository*)

Системи за управление на версии

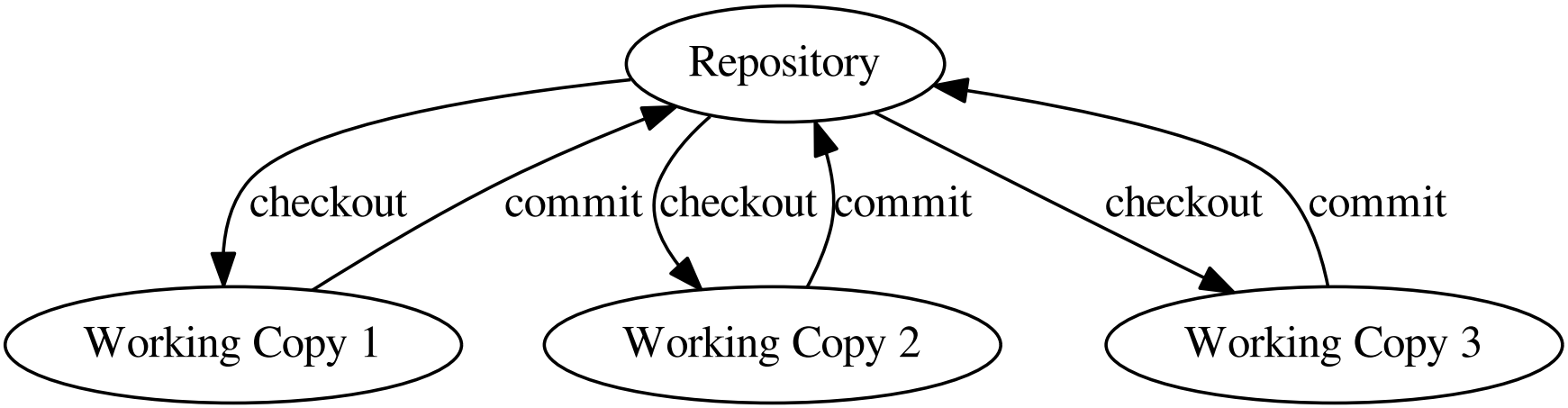


Фигура 1: История на промените в хранилище

Работа със системи за управление на версии

- Всеки от екипа разполага с копие на програмния код от някоя ревизия (най-често последната), което се нарича *работно копие*
- Докато се работи по дадена задача се променя кода в работното копие
- Когато задачата се счете за свършена в работното копие, трябва направените промени от работното копие да се добавят (*commit*) в хранилището като нова ревизия
- След като ревизията е добавена, другите хора от екипа могат да изтеглят (*checkout*) последната ревизия от хранилището в тяхното работно копие

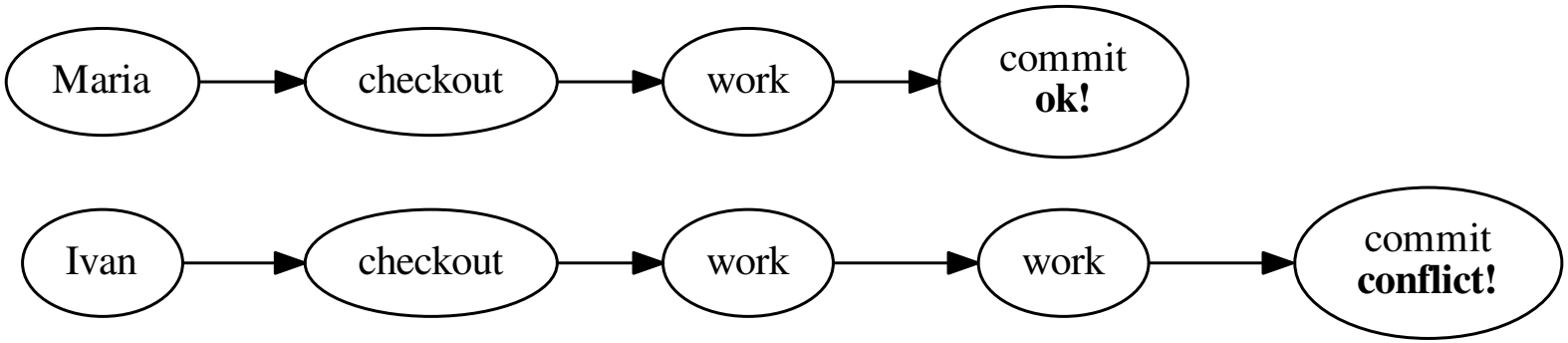
Работа със системи за управление на версии



Фигура 2: Работа с хранилище

Конфликти

- Когато се добавя нова ревизия в хранилището е възможно да възникне *конфликт* (*conflict*) - някой друг от хората в екипа е добавил нова ревизия в хранилището, за която човека, който добавя своята ревизия, не знае



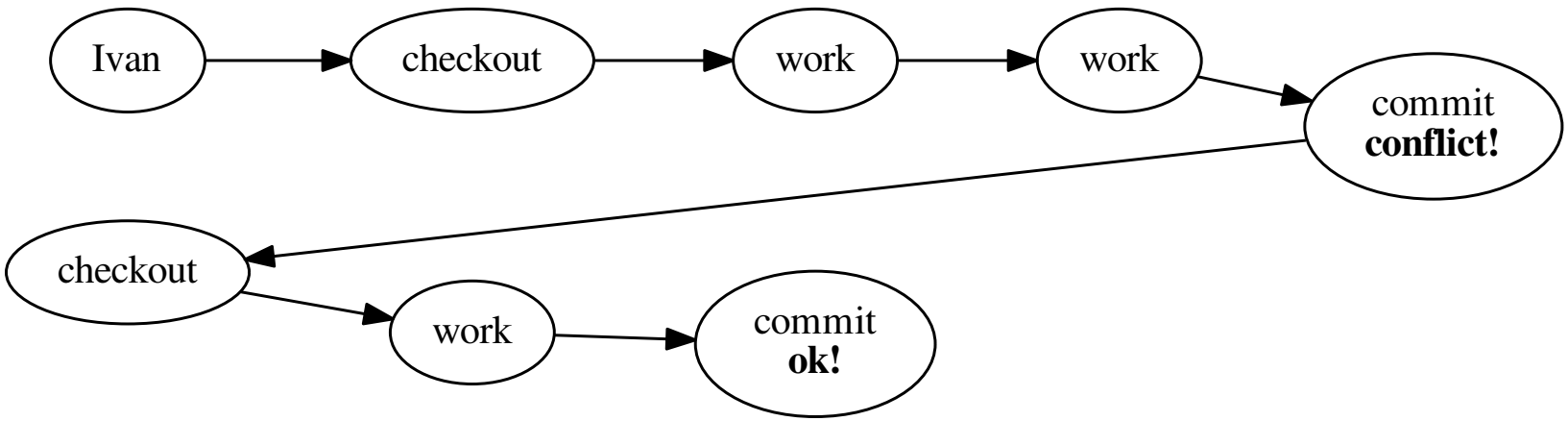
Фигура 3: Конфликт при добавяне на ревизия в хранилище

Конфликти

- Когато е работил върху своите промени, Иван е работил спрямо определена ревизия на кода (той е *базирал* своите промени върху нея)
- При добавяне на своята ревизия на кода в хранилището, той би изтрил промените на Мария (възниква конфликт)
- За да се избегне това, Иван трябва да базира промените си върху ревизията на Мария
  - Иван трябва да се запознае с направените промени в ревизията на Мария и ако е нужно, да направи допълнителни промени
- След като промените са базирани върху последната ревизия в хранилището (тази на Мария), Иван може да добави своята ревизия без да се загубят промени от други ревизии
- Почти всички съвременни системи за управление на версии позволяват автоматично базиране на промените спрямо друга ревизия, стига да не е нужна допълнителна намеса на програмиста

Разрешаване на конфликти

- Разрешаването на конфликти се състои в това да се изтегли (checkout) последната ревизия от хранилището и да се пренесат промените от старото работно копие (преди конфликта) върху новото. След това се прави нов опит за добавяне на ревизия в хранилището.



Фигура 4: Разрешаване на конфликт при добавяне на ревизия в хранилище

Съдържание

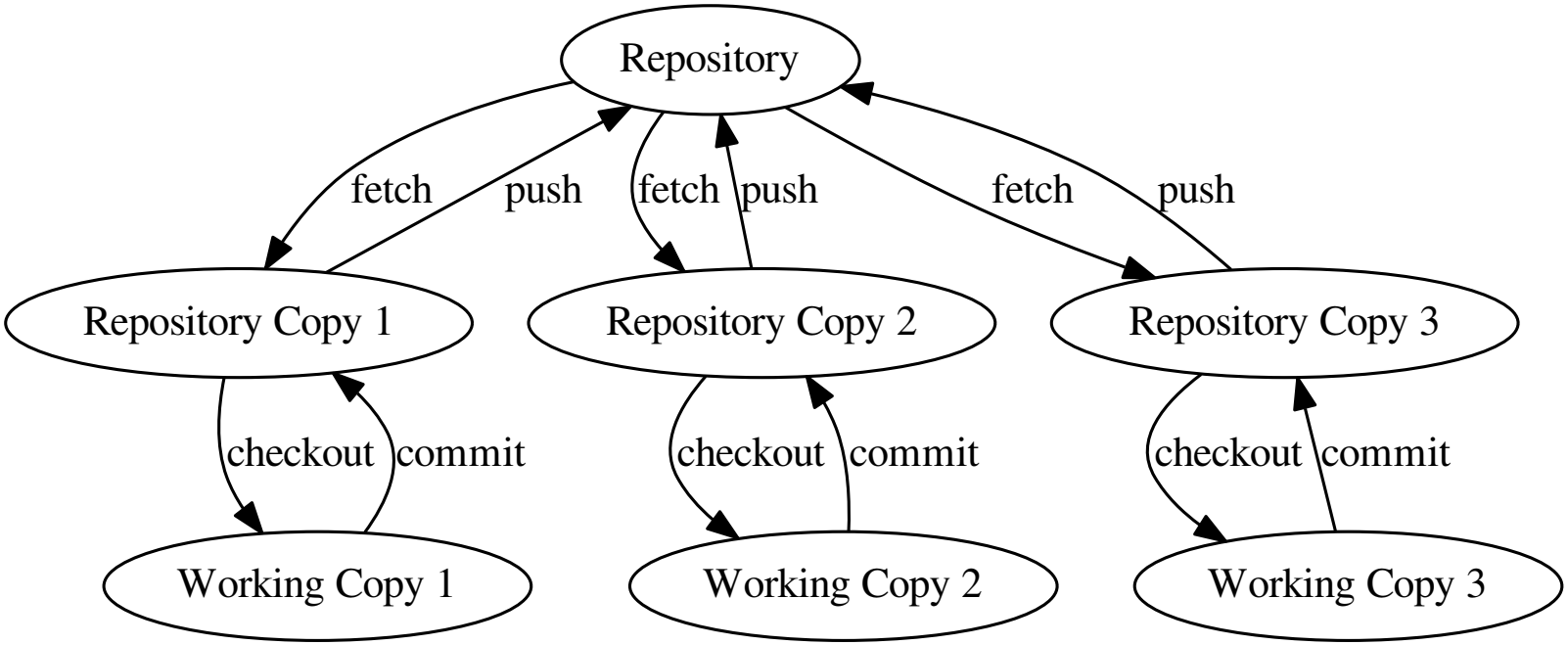
Децентрализирани системи за управление на версии

- При *централизираните* системи за управление на версии има едно хранилище. Техният недостатък е, че при отпадане на връзка към хранилището, не могат да се добавят нови версии - т.е. възпрепятства се работата по проекта
- При *децентрализираните* системи за управление на версии, освен работно копие на програмния код, при всеки се съхранява и копие на хранилището, наречено *локално хранилище*.
- Това позволява винаги да могат да се добавят промени в локалното хранилище, независимо от наличието на връзка с оригиналното хранилище (*отдалечено хранилище*). От друга страна това налага да се синхронизират локалните с отдалечените хранилища

Работа с децентрализирана система за управление на версии

- Работата с децентрализирана система за управление на версии добавя две нови действия, нужни за синхронизирането на локалното хранилище с отдалеченото.
- Донасяне (fetch) на ревизии от отдалеченото хранилище.
- Качване (push) на ревизии от локалното хранилище в отдалеченото.

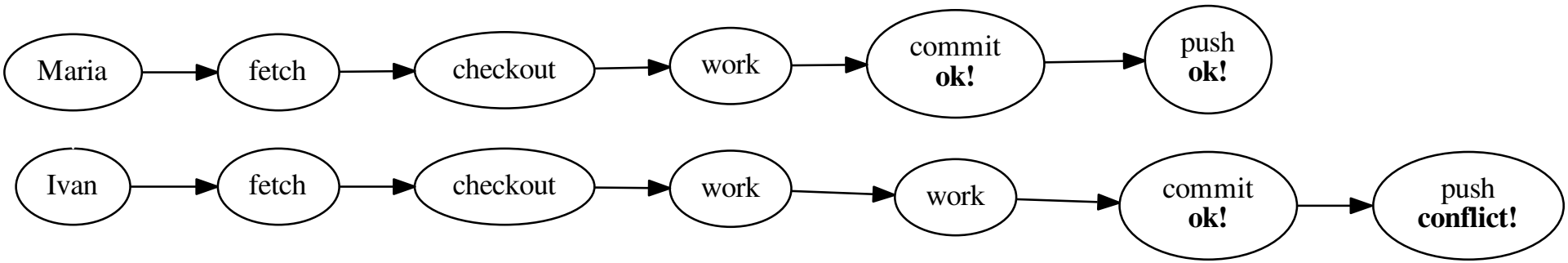
Работа с децентрализирана система за управление на версии



Фигура 5: Работа с децентрализирана система за управление на версии

Конфликти при децентрализирана система за управление на версии

- При синхронизация на локално хранилище с отдалечено е възможно да възникне конфликт - в отдалеченото хранилище има по-нови ревизии, за които локалното не знае



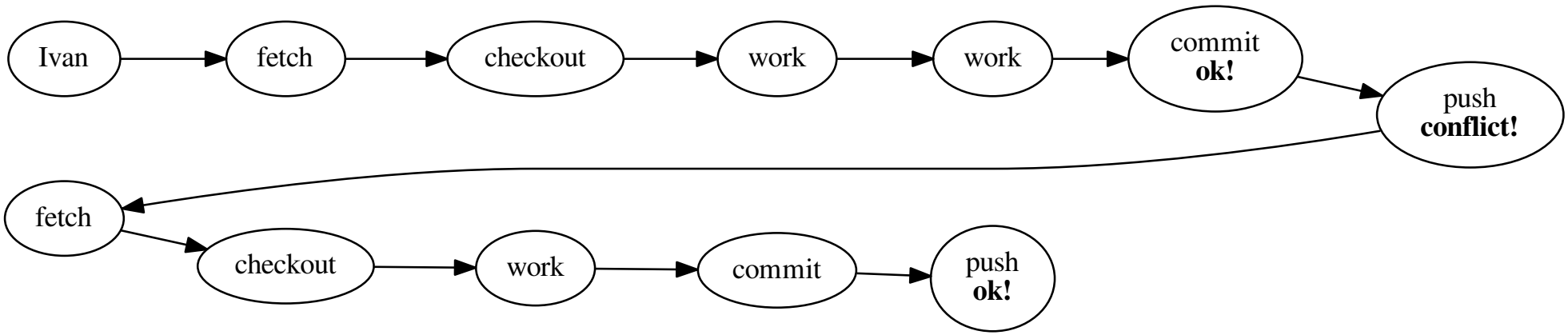
Фигура 6: Конфликт при качване на ревизии от локално в отдалечено хранилище

Конфликти при децентрализирана система за управление на версии

- Когато Иван е работил по задачи и е добавял ревизии, той ги е добавял спрямо някоя ревизия. Неговата *последователност от ревизии* е базирана върху тази ревизия
- При добавяне на своята последователност от ревизии в отдалеченото хранилище, той би изтрил промените на Мария (възниква конфликт)
- За да се избегне това, Иван трябва да обедини промените от своята последователност от ревизии заедно с тези на Мария в нова ревизия
  - Иван трябва да се запознае с направените промени в последователността от ревизии на Мария и ако е нужно, да направи допълнителни промени
- След като Иван обедини последователностите от ревизии в нова ревизия, той може да я качи без да се загубят промени от други ревизии.

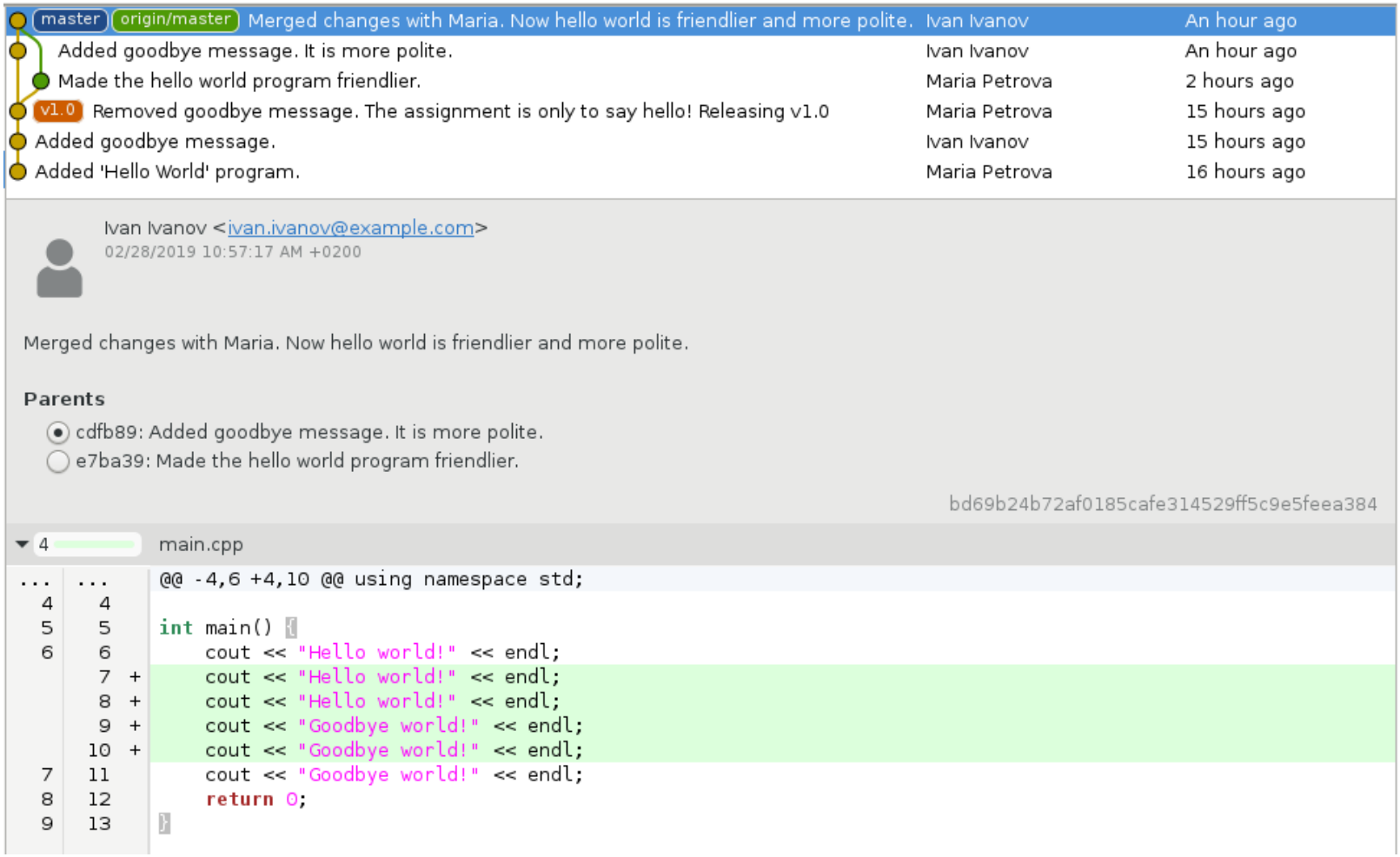
Конфликти при децентрализирана система за управление на версии

- Повечето децентрализирани системи за управление на версии предлагат автоматизиране на процеса, ако не е нужна намеса на програмиста



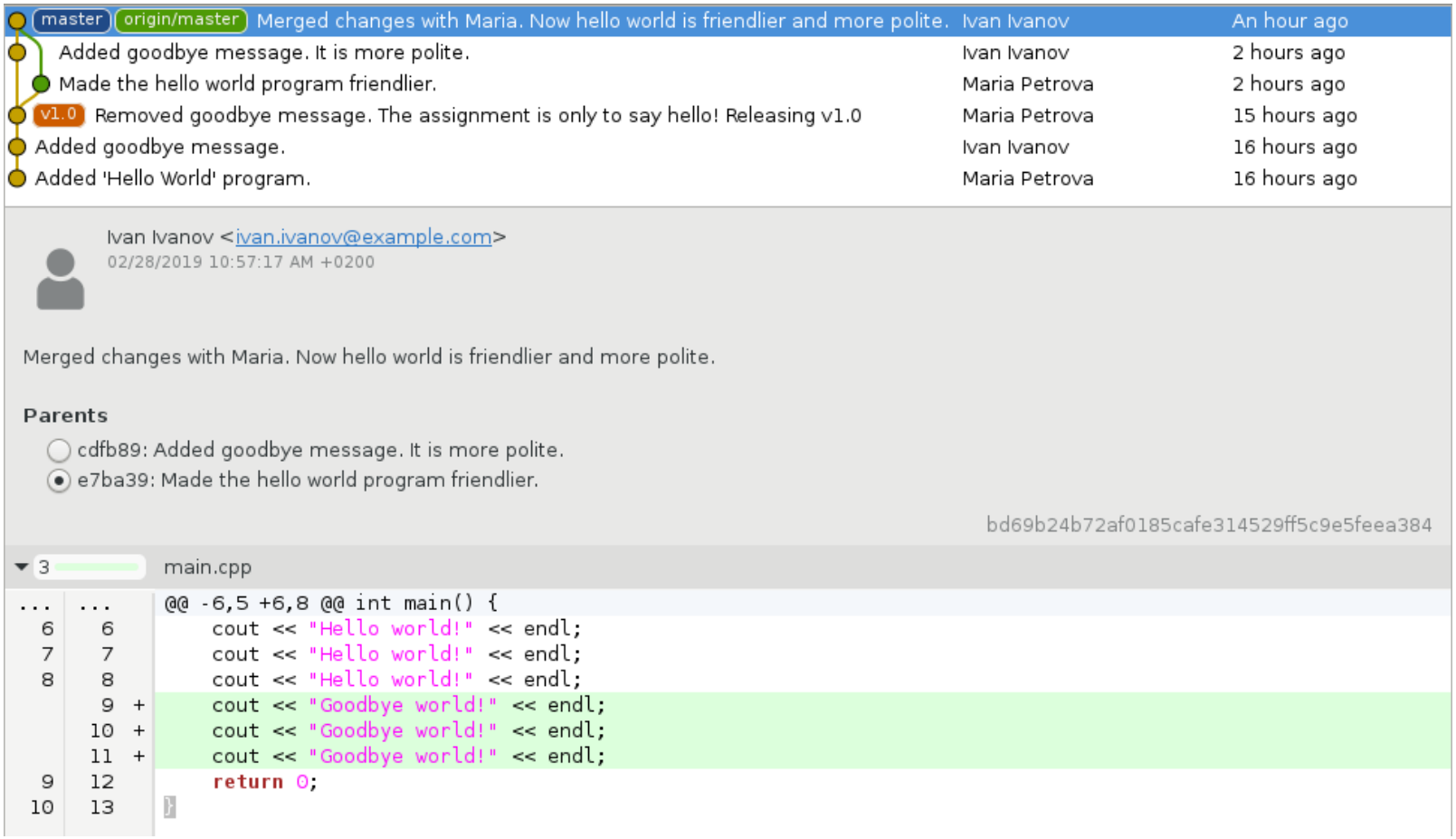
Фигура 7: Разрешаване на конфликт при качване на промени от локално в отдалечено хранилище

Конфликти при децентрализирана система за управление на версии



Фигура 8: История в хранилището след разрешаване на конфликт

Конфликти при децентрализирана система за управление на версии



Фигура 9: История в хранилището след разрешаване на конфликт

Съдържание

Git

- *Git* е децентрализирана система за управление на версии с отворен код, която работи под Windows, Linux, MacOS и други.
- В днешно време е една от най-разпространените.
- Сайтът Github предоставя безплатното създаване на отдалечени хранилища.

Работа с Git

- `git clone URL` - създава локално хранилище в текущата директория, което е копие на някое отдалечено
- `git add FILE` - добавя промените от работното копие на файла спрямо последната ревизия от локалното хранилище в *списък от промени, чакащи нова ревизия* (*staging area*)
- `git commit` - добавя нова ревизия в локалното хранилище с промените от списъка с промени
- `git push` - качва ревизиите от локалното хранилище към отдалеченото
- `git fetch` - донася (изтегля) ревизиите от отдалеченото хранилище в локалното
- `git checkout [REVISION]` - обновява работното копие с кода от определена ревизия, ако е посочена, или от последната в локалното хранилище
- `git pull` - прави `git fetch` и `git checkout` в една стъпка

Работа с Git

- `git status` - показва списък с променените файлове спрямо последната ревизия в локалното хранилище, както и съдържанието на списъка от промени, чакащи нова ревизия
- `git diff` - показва променените редове за всеки файл, който *не е включен* в списъка с чакащи промени, спрямо последната ревизия в локалното хранилище
- `git diff --cached` - показва променените редове за всеки файл, който *е включен* в списъка с чакащи промени, спрямо последната ревизия в локалното хранилище