Разработка на софтуер

Лекция 10 – Интеграция

Милен Спасов

Какво е интеграция

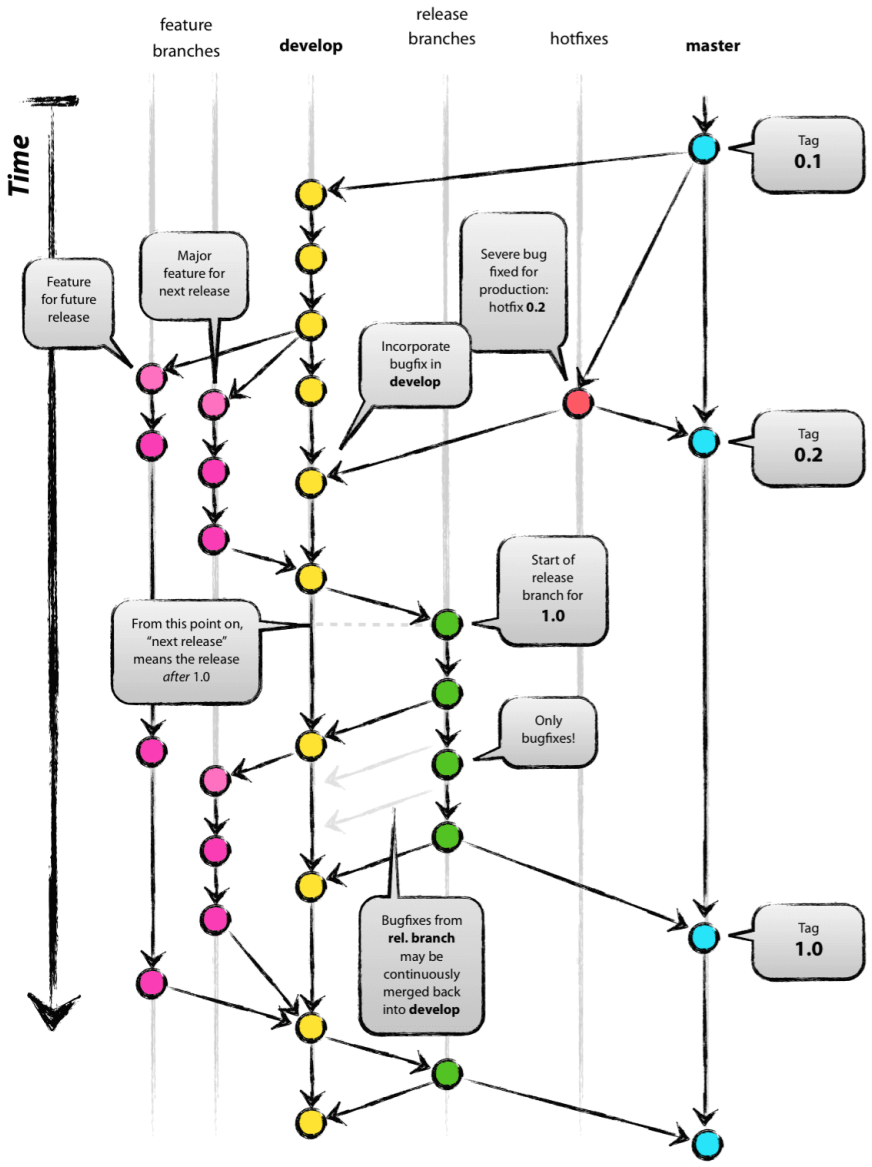
- Добавяне на промени (нови функционалности, бъгове, външни системи) към съществуващия код на системата

- Два основни типа интеграция:
  - На нови функционалности
  - На външни системи (най-често чрез REST API и обмен на данни в JSON или XML формат)

- Какво води до нужда от интеграция:
  - Разработка на нови функционалности
  - Преминаване от стара към нова версип на дадена система
  - Използване на външни услуги
  - Използване на външни системи
  - Преминаване от on premise към cloud
  - Всяка друга промяна на codebase-а на проекта

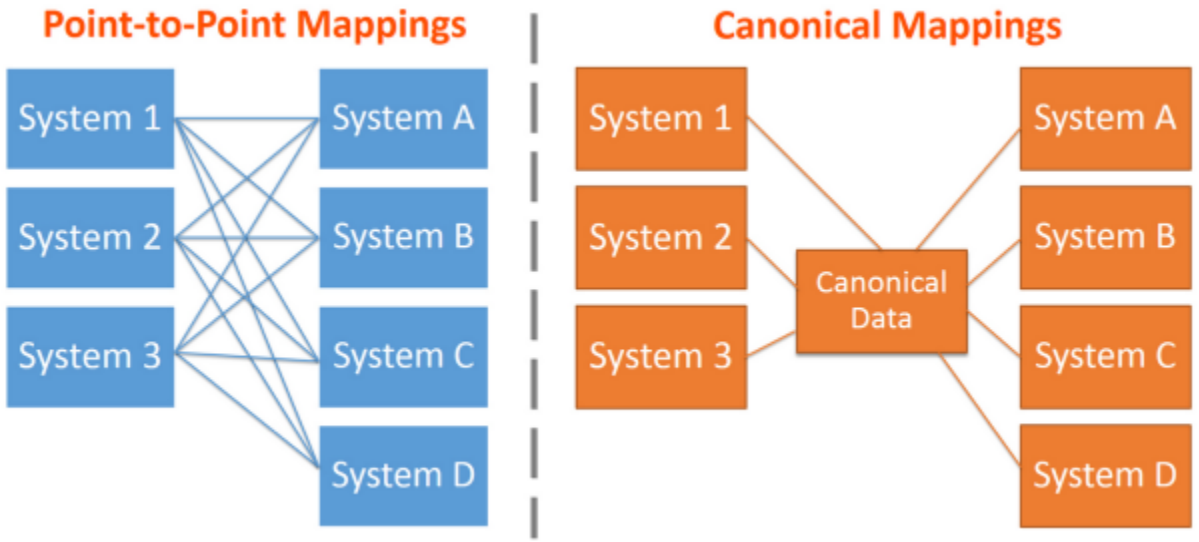
Интеграция на нови функционалности

- Интеграцията е всяко merge-ване към основния бранч (dev) или приемане на pull request
- Обичайно се интегрират feature branch-ове към основния
- Преди интеграцията се извършва код ревю и се проверява дали новата функционалност работи и дали има ефект върху други функционалности
- Всяка промяна води до нуждата от интеграция, независимо колко голяма е



Canonical data model design pattern

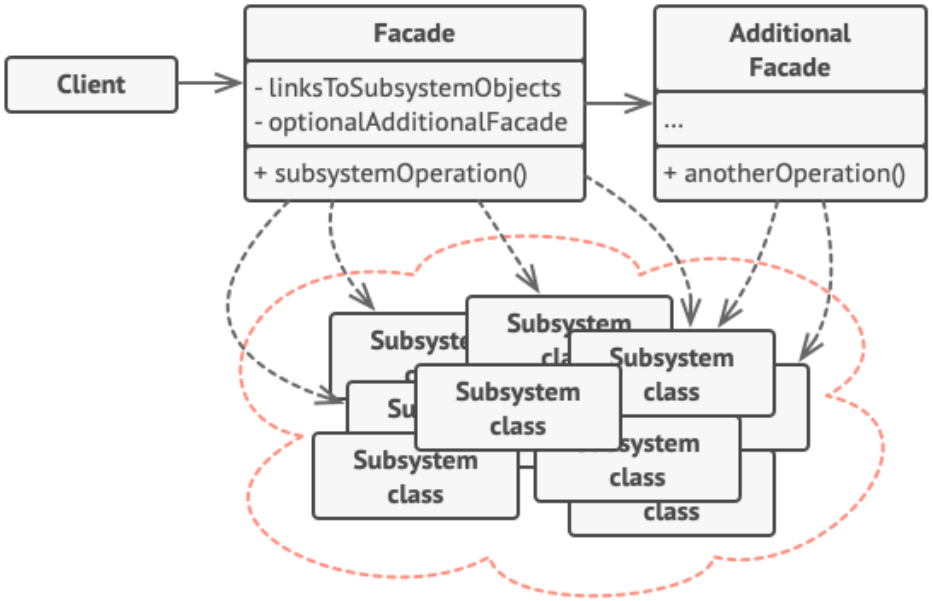
- Предимства
  - Ясна дефиниция на данните и зависимостите между тях
  - Опростена архитектура и намален брой връзки между модулите
  - Създаване на общ модел на данните за цялата система
  - Single source of truth по отношение на дните
  - Възможна разработка на адаптори за превод на данните в различни формати и за различни системи



Façade design pattern

- Предимства
  - Скриване на сложността на подсистемата от нейните клиенти
  - Сигурност и малка възможност за грешки, защото клиентите използват само минималните необходими интерфейси
  - Повишаване на абстракцията и енкапсулацията на комплексна логика, която не е необходима на клиентите

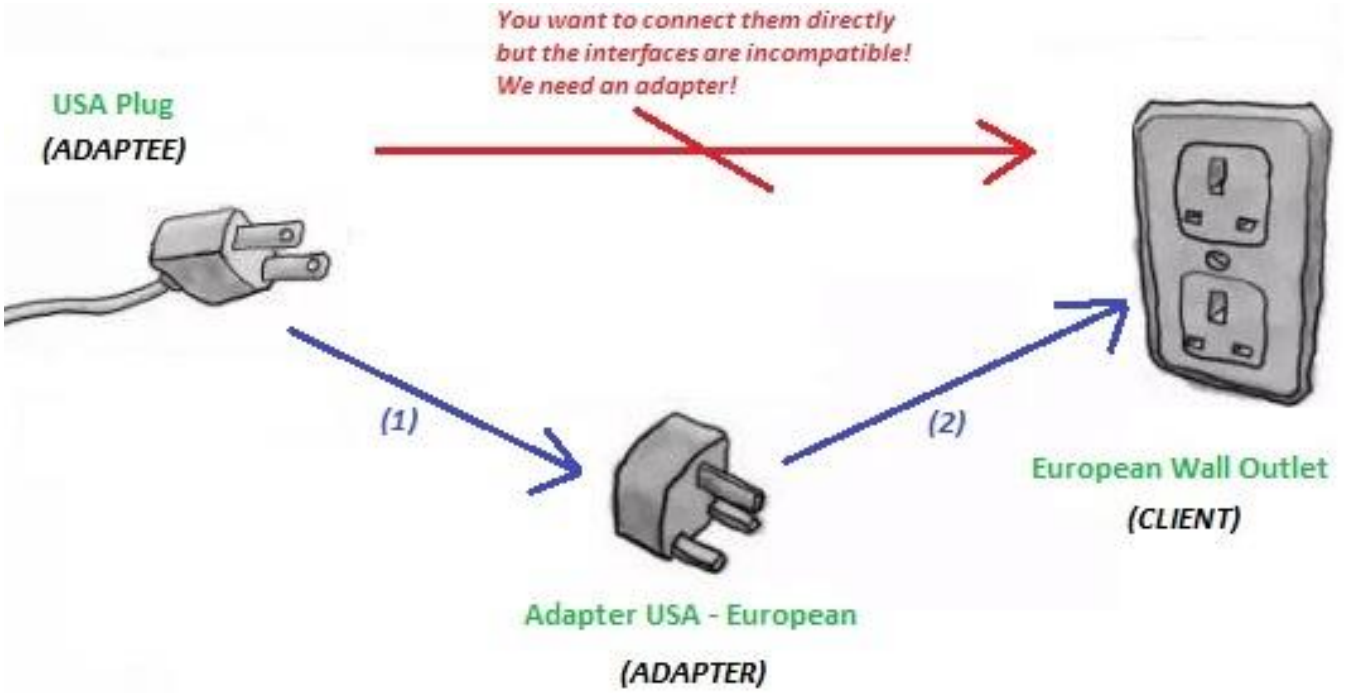
  - https://refactoring.guru/design-patterns/facade



Adapter design pattern

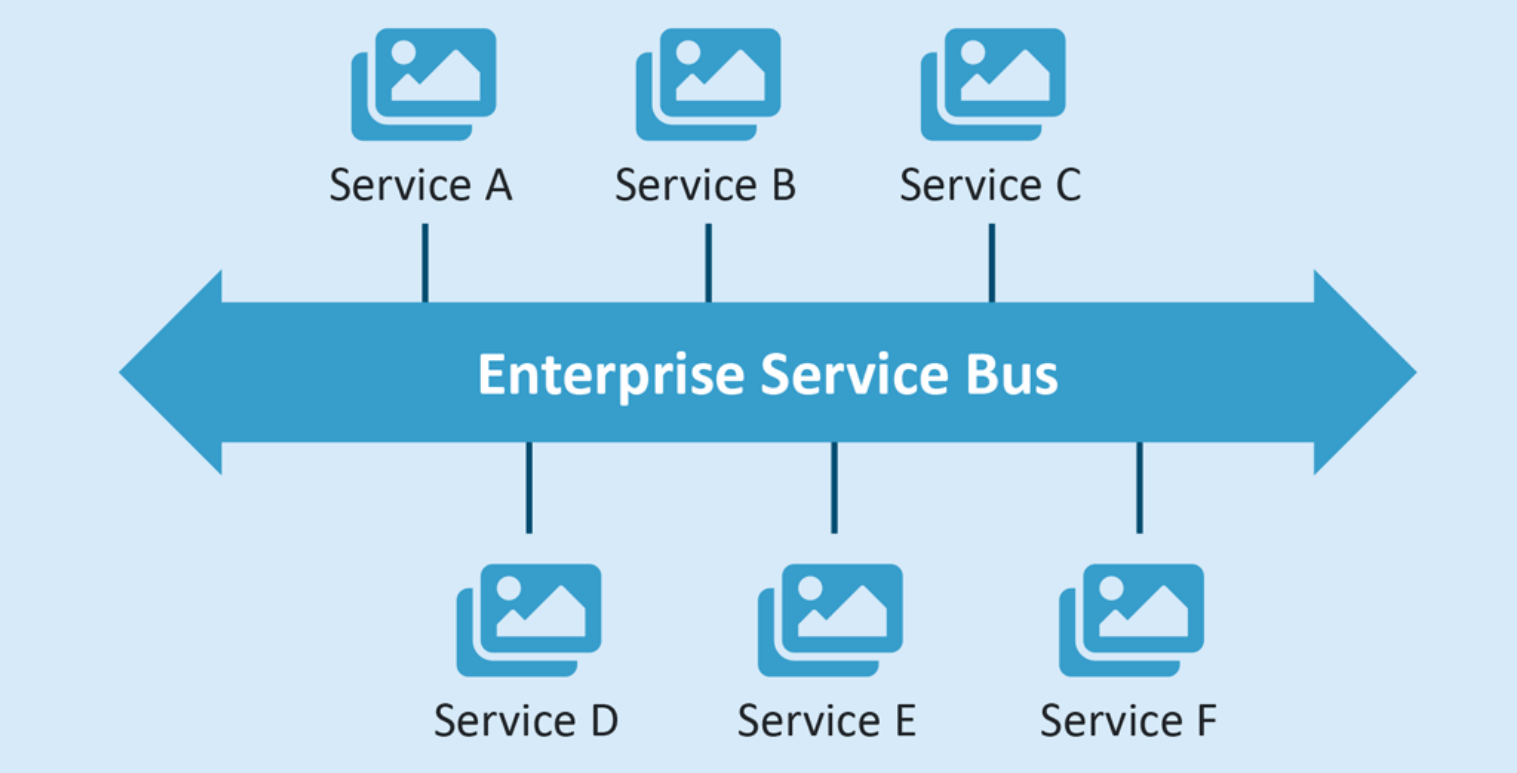
- Предимства
  - Превеждане на интерфейса между две системи, които без адаптер не могат да си комуникират
  - Не е нужна промяна в която и да е от двете системи
  - Изключително полезен pattern при системи, в които има много legacy код или при интеграция на външни системи, които не можем да променим

  - https://refactoring.guru/design-patterns/adapter



Message bus design pattern

- Предимства
  - Утвърден протокол за комуникация между софтуерни модули или цели системи
  - Механизми за защита и надеждна работа (напр. повтаряне на съобщение, приемане само от един или от много получатели, регистриране на грешки и др.)
  - Повишаване на абстракцията и разкачане на модулите един от друг
  - По-лесна поддръжка, промени и скалируемост на системата



Проблеми при интеграция

- Липсата на добра архитектура води до несъвместимост на отделни модули, компоненти или подсистеми
- Множество хора, които работят по различни задачи трябва да са наясно къде има допирни точки между тях
- Възможност за регресия или дефекти след интеграция, които не са очаквани или са в привидно несвързани модули, което води до нуждата от интеграционни тестове

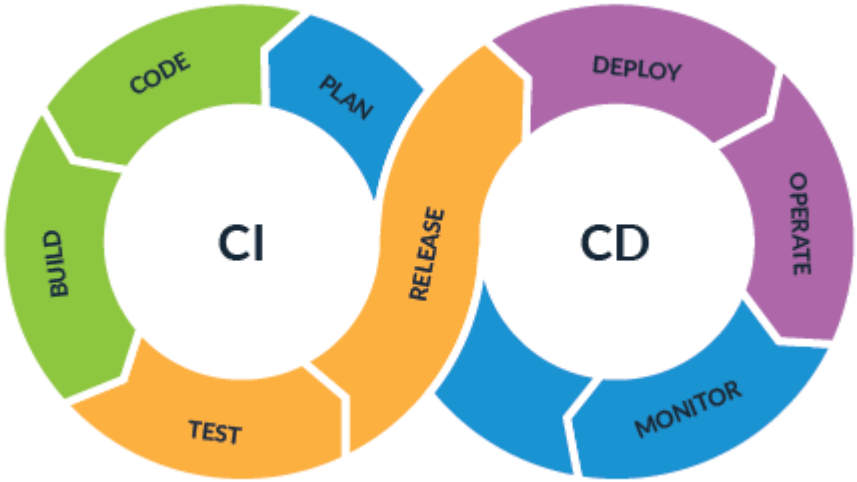
Big Bang Integration

- Късна интеграция на много промени
- Работа в изолация на хората от екипа
- Несъвместимост на разработените модули
- Късно откриване на грешки

Решение

- Разделяне на работата на по-малки части
- Честа интеграция на промените
- Автоматизирани тестове

Continuous integration



- Автоматизирано създаване на новите софтуерни версии при определени правила
- Автоматично изпълняване на тестове и информиране при възникнали грешки
- Continuous integration стъпки
  - Интеграция на промяна (нов commit в определен бранч)
  - Build-ване на текущата версия на проекта с тази промяна
  - Създаване на артефакт от билда (които после може да бъде deploy-нат)
  - Изпълнение на автоматизирани тестове
  - Нотификация при успешен или неуспешен резултат


Thank
You
köszönöm
Kiitos
Asante
Shukria
Vinaka
Maake
Dhanyavadagalu
감사합니다
Dankscheen
спасибо
Kam Sah Hammida
اركشكرا
Manana
Dankon
Matondo
Dank Je
Mauruuru
Biyan
Diolch i Chi
Terima Kasih
Taiku
Blagodaram
Ngiyabonga
Dziekuje
Tack
Chokrane
Grazie
Juspaxar
Arigato
Mochchakkeram
Gracias
Khap
Tingki
நன்றி
Bedankt
D'akujem
धन्यवाद
cảm ơn bạn
Paldies
Gratias Tibi
Ua Tsaug Rau Koj
Grazas
Di Ou Mèsi
Kia Ora
Kop Khun
Obrigado
Děkuji
Nuwun
Nirringrazzjak
ありがとう
Suksama
Rahmat
Matur
谢谢
хвала
Hvala
Welalin
Djiere Dieuf
Misaotra
Danke
Merci
Go Raibh Maith Agat
Tuke
Eskerrik Asko
Salamat
ขอขอบคุณคุณ
Najis