
# Architektura CRCE

KIV/SAR - semestrální práce

student: Radek VAIS, Zdeněk VALEŠ
mail: vaisr@students.zcu.cz, valesz@students.zcu.cz
datum: 1.1.2019

# 1 Zadání

CRCE[1] je komponentívé úložiště vyvíjené v rámci výzkumu na Katedře informatiky a výpočetní techniky na Západočeské univerzitě v Plzni. Jeho hlavní vlastností je podpora kontroly kompatibility verzí různých OSGi komponent nebo webových api.

## 1.1 Motivace

Vývoj projektu probíhá již několik let a vystřídalo se na něm mnoho vývojářů, což způsobilo zanesení větších či menších architektonických dluhů. Projekt již můžeme klasifikovat jako velký. Existuje potřeba, aby střídající se vývojáři (např. v rámci bakalářských prací) snadno a rychle zprovoznili základní instalaci a získali tak prostředí, pro rychlý vývoj nových částí.

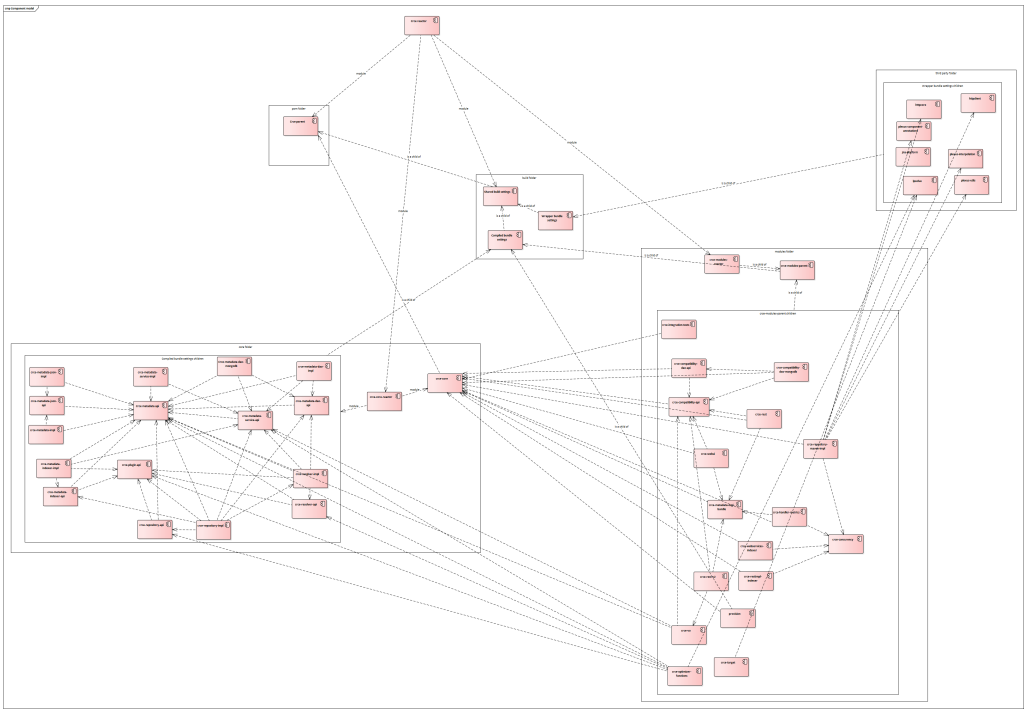
## 1.2 Cíle projektu

- Unifikovat proces sestavení aplikace a spuštění.
- Zdokumentovat současnou architekturu aplikace.
- Analyzovat závislosti modulů v projektu a ověřit, zda některé moduly mají být součástí jádra.
- Připravit Docker image pro vývojáře.

[1] Component Repository supporting Compatibility Evaluation

# 2 Analýza architektury

Celý projekt je rozdělen do čtyř hlavních modulů: sdílené nastavení sestavení, jádro, moduly a knihovny třetích stran. Ne zcela jasné je začlenění modulů pro nasazení mezi moduly CRCE. závislostí modulů na jádře pozorujeme dva typy deklarování závislostí. Prvním způsobem je závislost na agregačním modulu jádra, druhým je jmenovitá závislost na komponentách jádra. Tyto skutečnosti jsou patrné na celkovém diagramu na Obrázku 2.

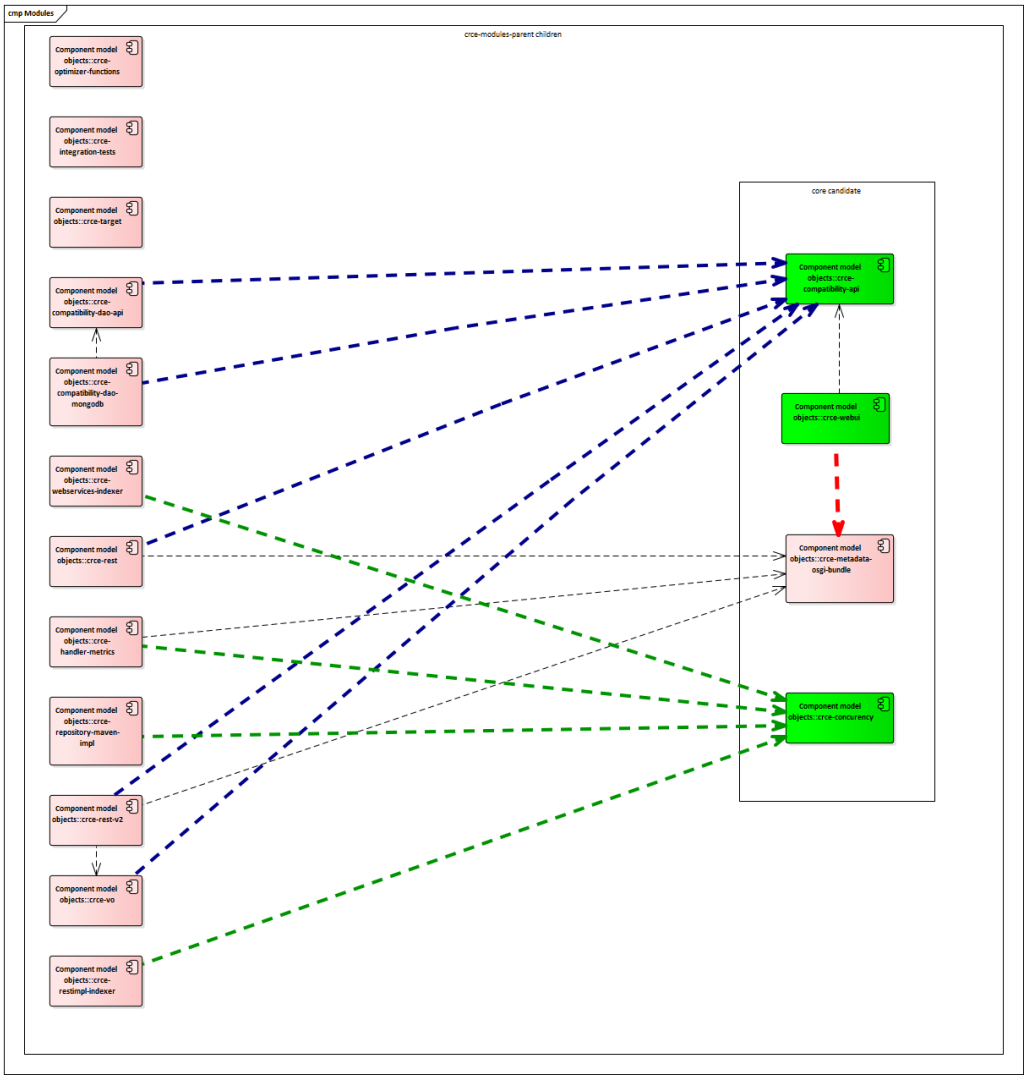


Obrázek 1: Diagram závislostí všech maven komponent projektu CRCE před provedenými úpravami.

Během analýzy modulu pro moduly CRCE jsme identifikovali několik kandidátů na přesun do jádra. Za kandidáty jsme volili takové komponenty, které využívá velká část ostatních modulů a zároveň tyto moduly mají závislosti do jádra. Tomuto pravidlu odporuje komponenta webového rozhraní, bez kterého úložiště nelze použít. Přesouvané komponenty jsou: crce-compatibility-api, crce-webui, crce-concurency (graficky znázorněno na Obrázku 2.

Po identifikace kandidátů jsme z přesunu vyřadili komponentu metadata-osgi-bundle, protože není dostatečně obecná. Komponenta indexuje pouze OSGi komponenty. Bohužel přesouvaná komponenta webui obsahuje nevhodnou závislost na této komponentě. Tato závislost jde proti snadné záměně

komponenty pro indexaci vstupů úložiště. Navrhli jsme tedy přidání metody do rozhraní indexeru pro zjištění jaké elementy indexer zpracovává.




Obrázek 2: Diagram původních závislostí uvnitř mudulu moduly CRCE. Zeleně jsou označeny přesouvané komponenty. Červeně je označena nevhodná závilsost.

# 3 Provedené změny

- Oddělení modulů pro spuštění CRCE do nového modulu deploy.

Vytvoření komponenty default-modules, která zachycuje verze spouštěných modulů.

- Přesun komponent crce-compatibility-api, crce-webui, crce-concurency do jádra CRCE.

  Vytvoření testů vlastností a defaultních rozhraní.

  Rozšíření rozhraní indexerů.

- Dokerizace základní instance CRCE.

  Konfigurace DB connection pomocí proměnných prostředí (oddělení DB).

# 4 Uživatelská příručka

Provedené změny ovlivnily základní používání projektu při vývoji.

Pro sestavení na platformě Linux lze použít bash script build.bash, který automaticky spustí jednotlivé kroky sestavení pomocí programu Maven.

Nastavení connection stringu je možné pomocí proměnné prostředí `mongo_connection`. Příkladem takového parametru je: *mongodb://localhost:27017*.

# 5 Závěr

V rámci projektu jsme naplnili cíl usnadnit novým vývojářům spuštění výchozí aplikace, především přípravou docker image. Nepodařilo se nám zjednodušit proces sestavení celé aplikace na jeden Maven příkaz nicméně pro prostředí Linux jsme vytvořili script, který úkony nutné pro sestavení provede automaticky. Dále jsme přesunuli několik modulů do jádra CRCE z důvodu jejich provázanosti s ostatními nebo nutnosti pro používání.