CDH Usage Pattern Analysis

Wiandt Bernát
2018. 03. 14.

Bevezető

A Cloudera cég fő terméke a CDH disztribúció, amely nagy adatmennyiségeket hatékonyan kezelő open source szoftverkomponenseket (big data eszközök) integrál egy egységes egészbe. A komponensek közül több hasonló funkcionalitású van, például fájlrendszerek, adatfeldolgozó komponensek. A cég felhasználói mind a saját céljaiknak megfelelően használják a CDH disztribúció komponenseit.

A kutatás egyik célja meghatározni, hogy mely komponenseket használnak együtt. Ez az információ segíthet a fejlesztési folyamat során történő döntéshozatalban. Például egyes komponensek CDH disztribúcióból történő eltávolítása megalapozott lehet, ha a használati statisztikák alapján a felhasználók elenyésző kisebbsége veszi igénybe az adott komponenst, cégen belül mégis erőforrást kell biztosítani a komponens karbantartására. Komponensek módosításakor vagy új funkcionalitás bevezetésekor is hasznos információ lehet, hogy az adott komponens mely másik komponensekkel kerül interakcióba a tipikus felhasználási esetekben.

A kutatás másik célja, hogy meghatározzuk az egyes komponensek népszerűségét és ezzel segítsük a megfelelő fejlesztői erőforrás allokálást cégen belül. Egy komponens népszerűségét több adatforrás felhasználásával is lehet közelíteni, azonban mindegyik adatforrás korlátozott ebben a tekintetben. Ezért több adatforrás együttes felhasználása a cél, hogy teljesebb képet kapjunk az egyes komponensekről.

Jira - az első adatforrás

Hibajegyek

A kutatáshoz felhasznált egyik adatforrás a nyílt forráskódú szoftverek fejlesztéséhez széles körben elterjedt Jira szoftver volt. A Jira egy hibabejelentő

és problémakövető szoftver, amelyben a fejlesztők ún. issue-kat (hibajegy) rögzíthetnek, amelyek lehetnek hibák, feladatok vagy fejlesztési kérések. Feltételezésünk szerint az Apache projektekhez használt Jira rendszerben rögzített hibajegyekhez rendelt adatokból következtetni tudunk majd az egyes komponensek életciklusára és népszerűségére. A hibajegyek mezői a következők lehetnek:

- **Type** - a hibajegy típusa, ez lehet task (feladat), bug (hiba) vagy new feature (új funkció), stb.
- **Priority** – az egyes hibajegyek közötti fontossági sorrend kialakítására szolgál
- **Status** – a hibajegy jelenlegi állapotát adja meg
- **Resolution** - itt jelezhető, ha a hiba megoldódott, ekkor zárható a hibajegy
- **People** - a közösség azon tagjai tartoznak ide, akik a hibajegyhez kapcsolhatók, mint a hiba bejelentője és a megoldója
- **Activity** - a Jira naplót vezet a hibák megoldásáról, a megoldás során felvetődött kérdésekről, és a munka folyamatáról

A hibajegy állapotai változnak, miközben az emberek megoldják. Ezeknek az idejét a Jira rendszer menti, így olyan információkhoz juthatunk, mint a létrehozási idő (Creation), vagy ha valaki elkezdi megoldani a problémát, akkor a státusza folyamatban van-ra (In progress) változik, és végül a megoldásnak az idejét (Resolved) is jegyzi a rendszer.

Adatok összegyűjtése

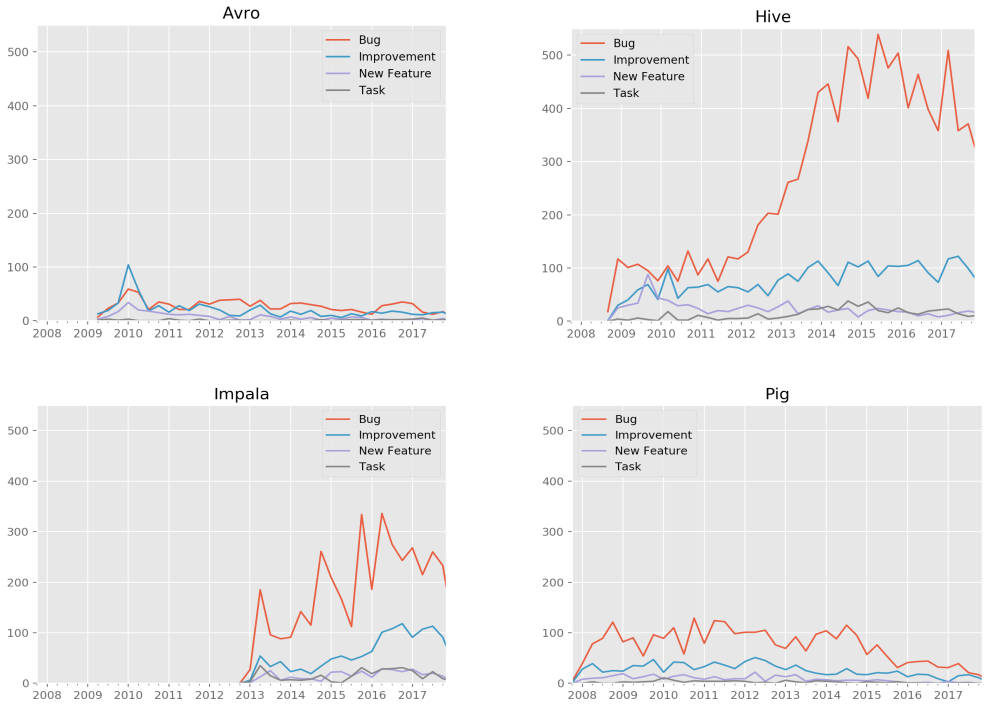
Az előző alfejezetben bemutatott Jira adatforrásból a félév során adatokat gyűjtöttünk. Ehhez a rendelkezésre álló REST API-t vettük igénybe. A lekérdezéseket JQL kifejezésekkel lehet megadni, amelyre a rendszer JSON formátumban ad választ.

A félév során a CDH különböző komponenseinél negyedévente létrehozott Jira hibajegyek számosságát vizsgáltuk. A hibajegyek lekérdezéséhez a hallgató egy Python modult készített, amelyben felhasználta a pip (python package manager)-ből elérhető *jira* modult. A letöltött hibajegyeket egy Pandas dataframe-ként tároltuk. Ebben negyedévente összegeztük a vizsgált komponenseknél létrejött

hibajegyek számát A hibajegyeket típusuknak megfelelően különböző kategóriákba sorultuk. Bug, Improvement, New Feature és Task.

Adatok megjelenítése

Négy, a CDH disztribúcióban fontos komponenst vizsgáltunk: Pig, Hive, Impala és Avro. Az eredményeket a következő grafikonokon jelenítettük meg, ügyelve arra, hogy ugyanolyan méretei legyenek mind az X, mind az Y tengelyeknek, így a grafikonok összehasonlíthatóak legyenek.



Látható, hogy mindegyik komponens életciklusának elején jellemzően emelkedtek a hibajegy mennyiségek. Utána viszont látható különbségek vannak a komponensek között.

Az egyes komponensek hibajegyeinek időbeli eloszlása lehetőséget ad különböző hipotézisek megfogalmazására. Az Avro hibajegyeinek száma sohasem emelkedett 100 fölé a 3 hónapos periódusokban, 2017 végére a számuk elenyésző. Ez alapján feltételezhetnénk, hogy az Avro-val ma már nincs sok megoldandó probléma. A Hive hibajegyek a kezdeti emelkedés után két éven keresztül tartották a 100 körüli értéket, majd meredeken nőni kezdett a Bug-ok száma 3 éven keresztül 4-500 körül mozgott, viszont 2017 végére lecsengés tapasztalható. A Hive komponens körül magasabb fokú aktivitást tapasztalunk,

mint az Avro esetében, ezáltal feltételezhetjük hogy a Hive egy népszerűbb komponens. Az Impala a legfiatalabb komponens, 4 éves, a hibajegyek száma folyamatosan nő, ebből látszik, hogy a fejlesztésére igény van, ez is egy népszerű komponens. A Pig Bug típusú hibajegyeinek a száma kb. 100 körül mozgott az idő során, azonban 2014 végén egy lecsengés látható, aminek ugyan voltak fölívelései, de 2017 végére 10-nél kevesebb hibajegy/hó volt a jellemző. Ez alapján feltételezhetjük, hogy a Pig egy fejlett komponens, elérte azt az állapotot amikor új funkcionalitás már nem vagy csak nagyon ritkán kerül bele és stabil építőköve a CDH disztribúciónak. Azonban egy másik hipotézis is megfogalmazható a hibajegyek számának alapján: a Pig egy elavult komponens, kevesen használják és ezért kevés új hibajegy is születik, a fejlesztői erőforrásokat más komponensre csoportosítják.
Ahhoz, hogy az alternatív hipotézisek közül nagyobb biztonsággal tudjuk kiválasztani a valóságot leírót, több adatforrást érdemes megvizsgálni.

Archív levelezési lista - a második adatforrás

Levelezési listák megismerése

A félév során használt második adatforrás a levelezési listák archívuma volt. Minden Apache projektnek van levelező listája, a legtöbb projektnek három: commits, dev, user listák. A commits listán értesítést kapnak a tagok, amikor változásokat véglegesítenek a verziókövető rendszerben. A dev listán a fejlesztők vitathatják meg a folyamatban lévő munkát és döntéseket hozhatnak. A user listán a projektekkel kapcsolatos általános kérdésekre válaszolnak a fejlesztők. A közösség archív leveleit a http://mail-archives.apache.org/mod_mbox/ honlapon keresztül lehet elérni.

Adatok összegyűjtése

Az adatelemzéshez a user listák tartalmát használtuk fel. Az archivált listák honlapjáról mind a négy projekthez (Avro, Hive, Impala, Pig) mBox formátumban letöltöttük a havi leveleket. Ezután a Python MboxExtractor moduljával a saját formátumomban tároltuk el. A hibajegyekhez hasonlóan a listára beérkező levelek számát is negyedéves bontásban vizsgáltuk.

Adatok megjelenítése

Az eredményeket a következő grafikonokon jelenítettük meg, ügyelve arra, hogy ugyanolyan skálán legyen értelmezve az x és az y tengelyek, így a grafikonok összehasonlíthatóak lesznek.

Ha összevetjük ezeket a Jira diagramjaival, akkor megfigyelhetjük, hogy mindegyik levelezési lista később indult el, mint ahogy az első hibajegy létrejött, például az Impala-nál 3 év telt el a kettő között. Emellett a levelezési listák indulásáról, már nem mondható el maradéktalanul az, hogy emelkedtek a mennyiségek, a Pig grafikonján látszik, hogy 3-4 évig nem volt levél a listán.

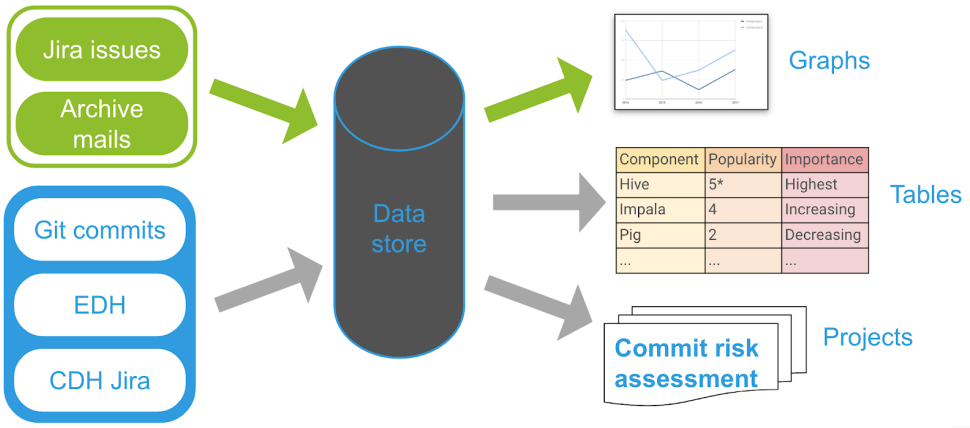
Az Avro levelei hegy formát mutatnak, volt egy felívelésük, majd az utóbbi években csökkent a számuk, ebből akár arra is lehet következtetni, hogy csökkent a népszerűsége, de természetesen egy adatforrás alapján nem lehet biztosat mondani. A Hive levelek folyamatosan ingadoznak, 2016 közepétől elkezdett csökkenni a számuk, ez a hibajegyekkel ellentétes, ott csak 2017 végén indult meg a csökkenés, tehát a Hive-nál nagyon fontos több adatforrás beemelése.

Az Impala levelezési listája nemrég indult el, folyamatosan nő a levelek száma, ennyi információ alapján nem lehet megállapítani az Impala népszerűségét, ehhez is több adatforrás szükséges. A Pig-re pedig az jellemző, hogy egy hirtelen

növekedés után egy gyors csökkenés következett be, 2017 végére már szinte nincs is email a levelezési listán.

További célok

A kutatás fő célja egy olyan rendszer kialakítása lenne, amely a fejlesztők számára fontos információkat tudna szogáltatni a disztribúció komponenseiről, illetve egy másik PARIPA programon belül futó projekt bemenetéül is szolgálhatna. Az ábrán zöld színnel szerepel a félév során elkészített rész és kék színnel az elkészítendő részek.



Az adatforrásokat tekintve a jelenlegi bázist bővíteni kell, így a nagyobb adatmennyiség birtokában pontosabban, megbízhatóbban lehet majd meghatározni a komponensek népszerűségi értékeit. Három plusz adatforrás hozzáadását tervezzük:

- Git kommitok: a komponenseket ért változások,
- EDH: a Cloudera használat elemző szoftvere, a komponensek igénybevételéről tudhatunk meg több információt,
- CDH Jira: a Cloudera belső Jira rendszere, amely külső felhasználók számára nem elérhető.