



Jak dobrać młotek do
gwoźdźnia?

Liczby

5 lat

13 osób (teraz 8)

171 różnych repozytoriów

18 agentów budujących

3937 merge requestów

48min od stworzenia MR do pierwszego review (mediana)

16h od stworzenia MR do zmergowania (mediana)

2018

Początek Intuition-1

Będziemy robić
software!

... tylko co dalej?

Czego potrzebujemy?

- System budowania
 - C++
 - Python
- Repozytorium
- Continuous integration
- Code review

Repozytorium

- GitLab
- Wersja self-hosted
- Nie ma ryzyka potencjalnego wycieku kodu/danych
- Nie ma ograniczeń na użytkowników/repozytoria
- Prostsza integracja z narzędziami wewnątrz firmy (np. web hooki)
- W 2024
 - 868 projektów
 - 91 użytkowników
 - 155GB

Continouous integration

- Jenkins
 - Znany i lubiany
 - Brak ograniczeń licencyjnych
 - Pipeline'y
 - Jenkins Pipeline Library
- Jenkins Configuration as a Code
 - Bo maszyny lubią padać
 - W tekście lepiej widać zmiany
 - Wiadomo co jest celowe a co przypadkowe
- CI = źródło prawdy działa/nie-działa

Code Review

- Reviewable jest fajne :)
- Review GitLaba (w 2018) jest kiepskie
- "Ile może zająć napisać czegoś do CodeReview?"
 - "Potrzymaj mi piwo"
 - <https://github.com/ChainsawDevelopment/CodeSaw>
- Główne funkcje
 - Śledzenie zmian między kolejnymi rewizjami
 - Rygorystyczne śledzenie stanu uwag
 - Integracja z GitLabem

» 🔒 ↺ ⏮ ⏭ pyproject.toml ⬇

1 2


Publish changes Choose file...

@3,3 - 16,16 (13,13 LINES)

```
3 # Because building requires many runtime changes, we don't use hatch to build packages.
4 # Wheel building is done in build_wheel.py script.
5 #
6 # To build python package locally, run:
7 # hatch run python build_wheel.py <platform_name> dist
8 #
9 # To publish python packages from the dist folder, run:
```

```
3 # Because building requires many runtime changes, we don't use hatch to build packages.
4 # Wheel building is done in build_wheel.py script.
5 #
6 # To build python package locally, run:
7 # hatch run python build_wheel.py <platform_name> dist
8 #
9 # To publish python packages from the dist folder, run:
```

Comments




Jakub Olesz

✓ 8.12.2023, 08:35:53

from

Resolved



Marek Antoniak

8.12.2023, 09:09:56

👍

Start new discussion...

Add comment Cancel

☐ Comment ☒ To fix ☐ Good work! Grab potato!

```
10 # hatch publish -r kplabs
11
12 [project]
13 name = 'kp-qemu'
14
15 [tool.hatch.envs.default]
```

```
10 # hatch publish -r kplabs
11
12 [project]
13 name = 'kp-qemu'
14
15 [tool.hatch.envs.default]
```

Jak ~~zmusić~~ przekonać ludzi do robienia CRek?

Niech ktoś



Jak ~~zmusić~~ przekonać ludzi do robienia CRek?

- 😞 Niech ktoś!
 - Wrzucasz merge requesta i czekasz
 - Nikt się czuje wystarczająco odpowiedzialny żeby zrobić CR
 - Wszyscy są "zbyt zajęci"
- 😞 Kółeczka wzajemnej adoracji
 - Brak propagacji informacji
 - Brak zewnętrznej perspektywy
 - "Hackowanie systemu"

Arek

- <https://github.com/ChainsawDevelopment/Arek>
- Automat
- Wysła wiadomości na Rocket.Chata
- Raz dziennie (7:00) przypominający o code review
- Automatycznie wyznacza dwie osoby do CR
- Automatycznie wybiera osobę do testowania i mergowania
- Równomiernie rozkłada CRki między cały zespół
- Przypomina o zbyt długo czekających MRach
- Przypomina o statusach na Jirze



Arek @rocket.cat 7:01

intuition-1 / obc / obc

- [i1o-18 frame combinator \(2 file\(s\), I1O-18\)](#)
 - Review in progress but issue has status In Progress? - Marcin Drobik
 - Created 7 days ago with status In Progress - Marcin Drobik
 - Review fixes. - Mariusz Lenczyk

Tycho

- [Add leopard to CI \(3 file\(s\), no id\)](#)
 - Review needed, ONLY 3 file(s)! - Maciej Nowak Marek Antoniak
 - Fixes requested by reviewers. - Paweł Klisz
- [Added xsct 2023_2 \(1 file\(s\), no id\)](#)
 - Approved but still not merged - Marek Antoniak

Oryx Jenkins Pipeline Library

- [optional master branch added to fetchTags \(1 file\(s\), no id\)](#)
 - Missing ticket status - Paweł Klisz
 - Review needed, ONLY 1 file(s)! - Mariusz Lenczyk Tomasz Martyniak

Pisanie kodu

- Jaki system budowania?
 - C++
 - Skąd brać toolchain?
 - Python
- Command-line first
 - nie ma problemów z budowaniem na CI
 - ale musi też dobrze się integrować z różnymi IDE

System budowania (C++)

- CMake
 - Wieloplatformowy
 - Wsparcie IDE
 - Ninja (zdecydowanie szybsza niż Make!)
- C++17
- arm-none-eabi-gcc
- clang-format

Skąd brać toolchain?

- Lista i 'zbuduj sobie sam'
- Raz poskładany ZIP i leży 'na sieci'
- Automat do sklejania

Toolchain Builder

- Skrypt budujący ZIPa z toolchainem
 - CMake, Ninja, GCC, QEMU
 - Pobierz -> rozpakuj -> działa
- Budowane na CI
- Automatyczna dystrybucja do wszystkich agentów
- Historia zmian

System budowania (Python)

- Python zintegrowany w CMakeLiście
 - Automatyczne tworzenie virtualenva w folderze budowania
 - Automatyczne instalowanie zależności z `requirements.txt`
- Wymuszamy type-hintsy
 - Sprawdzane przy pomocy mypy
- Flake8 + rodzina jako linter

Jedno
repozytorium czy
więcej?

Dlaczego dzielnie kodu jest istotne?

- Teraz mamy jeden projekt
- ... ale w przyszłości będzie więcej
- ... chcemy zbudować sobie bazę klocków na przyszłość



Source: <https://trollopolis.fandom.com/wiki/Copypasta>

Współdzielenie kodu (C++)

- C++ nie ma dominującego menadżera pakietów
- Nie chcemy wpakować się w problemy od samego początku

Współdzielenie kodu (C++)

- Wybieramy rozwiązanie technicznie proste i zobaczymy co będzie
- `FetchContent`
 - Pobieranie źródeł z GitLaba na podstawie commita
- Kilka paczek (2018)
- Cała reszta w monorepo
 - ale pamiętajmy o dobrej separacji

Współdzielenie kodu (Python)

- Budowanie z repozytorium I1 OBC
- Nexus jako serwer paczek
- Wersjonowanie - ``setuptools_scm``
 - Tagi ``release-x.y.z`` oznaczające stabilne wersje
 - Automatyczne generowanie wersji ``dev`` licząc commity od ostatniej stabilnej
- CMake opisujący proces budowania paczek pythonowych

A kto to panu
tak... ustawił?

Infrastruktura

- Skalowanie wszerek
 - dużo słabszych maszyn zamiast jednej wielkiej
- Szybkie przywrócenie do działania zamiast 99.999...% niezawodności
- Infrastructure as a Code
 - Ansible
 - Code review
- Dodanie nowego agenta od Jenkinsa w < 1h

2020

Prace przyspieszają

- Arek - dwa razy dziennie
 - 7:00 i 14:00 - pełny zrzut czekających merge requestów
- Arek - inkrementacyjny
 - Skanuje GitLaba co kilka minut i wrzuca informacje o nowych MR **natychmiast**
- ``!arek`` - odsyła informacje przefiltrowane dla konkretnej osoby
 - Dzięki self-hosted używanie webhooków jest proste

Create to first review

bucket		Total Counts
[0, 2.280577)		2400
[2.280577, 4.561154)		304
[4.561154, 6.841731)		206
[6.841731, 9.122308)		269
[9.122308, 11.40288)		260
[11.40288, 13.68346)		170
[13.68346, 15.96404)		106
[15.96404, 18.24462)		22
[18.24462, 20.52519)		22
[20.52519, 22.80577)		26
[22.80577, 25.08635)		26
[25.08635, 27.36692)		29
[27.36692, 29.64750)		34
[29.64750, 31.92808)		10
[31.92808, 34.20865)		4
[34.20865, 36.48923)		4
[36.48923, 38.76981)	·	7
[38.76981, 41.05038)		10
[41.05038, 43.33096)		4
[43.33096, 45.61154)	·	6
[45.61154, 47.89212)		0
[47.89212, 50.17269)		0
[50.17269, 52.45327)		1
[52.45327, 54.73385)		3
[54.73385, 57.01442)		1
[57.01442, 59.29500)	·	5

Outliers (>60 hours) 25

Median 0.79

90th percentile 12.61

Reviewed in less than 2 hours 60.07%

> 

Rev.	2023 (count)	2023 (%)	2023 (avg +/-%)	2024 (count)	2024 (%)	2024 (avg +/-%)
jo	233	9.56	-13.99	36	14.01	12.06
ko	307	12.59	13.33	0	0.00	-100.00
ma	378	15.50	39.54	34	13.23	5.84
md	260	10.66	-4.02	25	9.73	-22.18
ml	222	9.11	-18.05	28	10.89	-12.84
mn	312	12.80	15.18	33	12.84	2.72
pk	270	11.07	-0.33	41	15.95	27.63
pr	189	7.75	-30.23	37	14.40	15.18
tm	267	10.95	-1.44	23	8.95	-28.40
Total	2438	100.00	270.89	257	100.00	32.12

Więcej projektów

- Toolchain builder
 - Nowe toolchainy
 - Kod paczkujący CMake/Ninja/QEMU wspólny dla wszystkich toolchainów
 - Dodanie nowego toolchaina jest proste
- Jenkins
 - Dodawanie projektów w JCasC jest proste
 - Mała zmiana: dodajemy całe grupy z GitLaba

Więcej projektów

- FetchContent
 - Zaczynają się problemy
 - Słabe discoverability
 - Problem z wersjami (używamy hashy a nie tagów)
- `FetchContent` nie ma przyszłości
 - Wiemy, że trzeba będzie zbudować inne rozwiązanie
 - Po dwóch latach lepiej rozumiemy potrzebny model paczkowania

2021

Kolejka mergowania

- Cały zespół się rozpędził
 - Przy rozbudowanych podstawach łatwo dodawać nowe funkcje
 - Skutek: dużo małych merge requestów
- Wymagamy merge'a typu fast-forward
 - `git bisect` jest prostszy
- Build na CI trwa ~40 minut (długie testy sprzętowe)
- Wyścig kto pierwszy zmerguje
- Pozostali: rebase -> build -> merge
 - ...a ktoś znowu może być szybszy

- **Maciej Nowak** added **Merge** label 3 days ago
- **Hulk** added 4 commits 3 days ago
 - [7ea26e8a...94a53dc2](#) - 2 commits from branch **master**
 - [e44a2fc9](#) - Uplink mode / downlink mode confusion in gnuradio gs
 - [a1704d7e](#) - Until now, when we sent 5 radio frames at once, GNU Radio treated each

[Compare with previous version](#)
- **Hulk** enabled an automatic merge when the pipeline for [a1704d7e](#) succeeds 3 days ago
- **Hulk** merged 3 days ago
- **Hulk** mentioned in commit [f843a36b](#) 3 days ago

Kolejka mergowania

- Tool który co kilka minut skanuje GitLaba
- Szuka merge requestów z etykietą ``Merge``
- Ustawia znalezione merge requesty w kolejce
- Dla pierwszego w kolejce:
 - Rebase (jeśli trzeba)
 - Dobrze podzielony kod daje mało konfliktów
 - Czeki na code review i CI
 - CodeSaw potrafi rozpoznać nieistotne zmiany
 - Merge
- Z punktu widzenia developera:
 - ustaw etykietę ``Merge``
 - "W końcu się zmerguje"

Labeled with "Merge" to actual merge		Total Counts	
bucket			
[0	, 0.577306)	=====	1029
[0.577306,	1.154611)	=====	87
[1.154611,	1.731917)	=====	49
[1.731917,	2.309222)	=====	27
[2.309222,	2.886528)	=====	15
[2.886528,	3.463833)	=====	10
[3.463833,	4.041139)	=====	14
[4.041139,	4.618444)	=====	4
[4.618444,	5.195750)	=====	6
[5.195750,	5.773056)	=====	5
[5.773056,	6.350361)	=====	3
[6.350361,	6.927667)	=====	0
[6.927667,	7.504972)	=====	2
[7.504972,	8.082278)	=====	2
[8.082278,	8.659583)	=====	5
[8.659583,	9.236889)	=====	6
[9.236889,	9.814194)	=====	2
[9.814194,	10.39150)	=====	7
[10.39150,	10.96881)	=====	5
[10.96881,	11.54611)	=====	2
[11.54611,	12.12342)	=====	2
[12.12342,	12.70072)	=====	3
[12.70072,	13.27803)	=====	3
[13.27803,	13.85533)	=====	2
[13.85533,	14.43264)	=====	3
[14.43264,	15.00994)	=====	4
[15.00994,	15.58725)	=====	4
[15.58725,	16.16456)	=====	5
[16.16456,	16.74186)	=====	3
[16.74186,	17.31917)	=====	3

Outliers (>18 hours) 24

> 

Conan

- Wprowadzamy Conana do zarządzania paczkami
 - Pozostajemy przy Toolchain Builderze
 - Uogólnione mechanizmy uruchamiania testów jednostkowych
 - Profile do zarządzania opcjami kompilacji i ścieżkami do toolchaina
 - Profile - wspólne dla wszystkich (Git)
- Masowa akcja wyciągania paczek z I1 OBC
 - Każdy wyciąga po jednej paczce
 - Proste paczki dla nauki

Wersjonowanie paczek

- Problem z ``setuptools_scm``
 - Niekompatybilny z formatem wersji Conana
 - Trzeba pamiętać o dodawaniu tagów
- Tworzymy własny mechanizm wersjonowania

KP Version

- Opieramy się na zasadach Semantic Versioning 2.0.0
- Wspólny mechanizm formatowania wersji: SemVer (dla Conana) i PEP 440 (dla Pythona)
- Strategia wersji:
 - Każdy commit na masterze ma stabilną wersję
 - Wersja określana na podstawie Conventional Commits 1.0.0

Stan obecny

- Conan:
 - 94 paczki
 - 3997 wersji paczek na serwerze
 - 117GB paczek
 - Rekordzista budowany na 18 profili
- Nexus
 - 66 paczek
 - 3055 wersji
 - 506 MB

2023

Czas leci, wersje się zmieniają

- "Zabetonowane wersje"
 - Toolchain builder zcentralizował toolchain
- Jak zrobić aktualizację nie psując istniejących projektów?
- Nie tylko raz, ale **ciągle!**

Conan i ``tool_requires``

- Porzucamy toolchain buildera
- Conan obsłuży dostarczanie wymaganych narzędzi (CMake, GCC, QEMU, etc.)
- Każdy projekt może rozwijać się niezależnie
- Opcje kompilacji też będą wersjonowane
- Decentralizacja

Profil

```
[settings]
os=Embedded
arch=arm_cortex
arch.cpu=cortex-m3
compiler=gcc
compiler.libcxx=libstdc++11
compiler.version=12
compiler.cppstd=17

[tool_requires]
oryx-qemu-8/1.0.0@kplabs/stable
gcc-arm-none-eabi-12/1.0.0@kplabs/stable
ninja/1.10.1@kplabs/imported

[conf]
tools.cmake.cmaketoolchain:generator=Ninja
```

Recipe

```
class Package(ConanFile):
    tool_requires = (
        'cmake/3.27.7@kplabs/imported',
        'oryx-lua-cli/[^5.3.0]@kplabs/stable',
    )
```

Python problemy

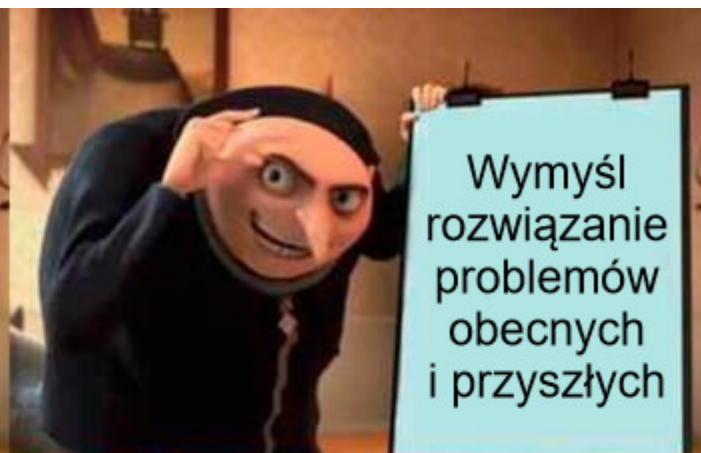
- Python osadzony w CMake'u nie jest fajny
- Mnóstwo ifologii na to czy odpalać Pythonowe rzeczy
- Jedna lista zależności to:
 - dłuższe czasy instalacji
 - potencjał na konflikty, które mają sensu (np. ``mypy`` vs ``sphinx``)
 - paczki czysto pythonowe potrzebują... CMake'a

Python podejście drugie

- `pyproject.toml` + Hatch
 - Hatch rządzi zależnościami
 - Podział na kilka virtualenvów
 - Przezroczyste wywołania
- CMake woła tylko `hatch run xxx`
 - Zarządzanie virtual envami przestało być istotne
- Dobra integracja z `kp_version`
 - Wymaga dość prostego plugina
- Paczki czystopythonowe używają tylko Hacha

pipx

- Wygodne na lokalnej maszynie
 - Nie ma potrzeby "śmiecenia" w systemowej instalacji Pythona
- Integracja z Jenkinsem w formie kroku w Jenkins Pipeline Library
 - `withPipxInstalled('3.11', ['hatch==1.7.0']) {}`` <- instaluje Hatcha tu i teraz
- Aktualizacje narzędzi stają się prostsze



Migracja

- Mamy ~90 paczek
- Jak zaktualizować wszystkie?

Paczka tygodnia

- Raz w tygodniu przydzielenie paczki do migracji każdej osobie
- ~8 paczek na tydzień
- ~2 miesiące na całą migrację
- Przez cały okres migracji wszystko musi być kompatybilne

Paczka tygodnia

- Równomierne rozłożenie pracy po całym zespole
- Propagacja wiedzy
- Weryfikacja dokumentacji
- Ma zastosowania do wszelkich żmudnych prac

Aktualizacje

- Ktoś robi zmianę w paczce
- Zewnętrzna paczka ma nową wersję
- My mamy ~50 użyć tej paczki
- Jak żyć?

Renovatebot

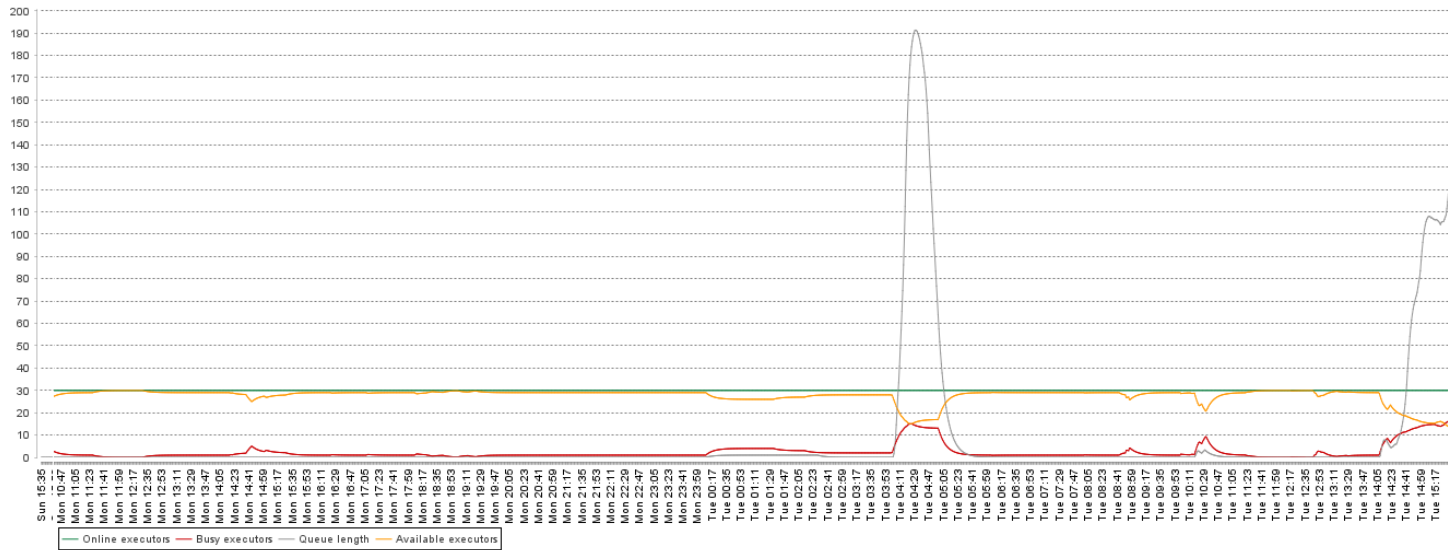
- Automat robiący aktualizację paczek raz na dobę
- Automatycznie wystawia merge requesty
- Bardzo (zbyt?) elastyczna konfiguracja
- Jak CI przechodzi to MR jest ok
- Szybkie aktualizacje => jednorazowo mniejsze zmiany => szybciej osiągamy stan aktualny
- Ponad 1000 merge requestów w ciągu roku

Renovate MRs create to merge	bucket	Total Counts
[0, 2.777778)		373
[2.777778, 5.555556)		132
[5.555556, 8.333333)		117
[8.333333, 11.11111)		21
[11.11111, 13.88889)		6
[13.88889, 16.66667)		134
[16.66667, 19.44444)		48
[19.44444, 22.22222)		29
[22.22222, 25)		3
[25, 27.77778)	.	1
[27.77778, 30.55556)		33
[30.55556, 33.33333)		17
[33.33333, 36.11111)		7
[36.11111, 38.88889)		3
[38.88889, 41.66667)		2
[41.66667, 44.44444)		0
[44.44444, 47.22222)		21
[47.22222, 50)		0
[50, 52.77778)		7
[52.77778, 55.55556)		0
[55.55556, 58.33333)		0
[58.33333, 61.11111)		6
[61.11111, 63.88889)	.	1
[63.88889, 66.66667)		24
[66.66667, 69.44444)		0
[69.44444, 72.22222)		0
[72.22222, 75)	.	1

Outliers (>80 hours) 12

Merged in less than 8 hours 62.373225%

> 



Szybkie aktualizacje


- Renovatebot zapewnia szybkie (24h!) aktualizacje zależności
- Dużo merge requestów z małymi zmianami wersji
- Nawet breaking change nie są trudne do naprawienia
- Konflikty wersji przestały występować

Wyszukiwanie kodu

- Jak znaleźć coś w 170 repozytoriach?
- Wyszukiwarka kodu Gitlaba jest kiepska
- Klonowanie 100+ repozytoriów jest trudne w utrzymaniu

Sourcegraph

- Indeksuje cały nasz kodzik
- Wyszukiwanie tekstowe
- Rozbudowane filtrowanie
- Szybkie



 Code Search ▾




Notebooks

Monitoring


Batch Changes

Insights


 

 context:global WriteDoubleWordLE lang:C++ repo:^git\.kplabs\.pl/oryx/packages/ Aa .* []  

91 results in 0.01s ▾ Actions ▾


oryx/packages/binary-logger > api/log_api/include/binary_logger/MessageWriter.hpp 

```
253     {
254         writer.WriteDoubleWordLE(static_cast<std::uint32_t>(value));
255     }
```




oryx/packages/antelope-bsp > antelope_bsp/libs/boot_state/src/BootStateSerialization.cpp

```
16     writer.WriteCharacter(value.UnconfirmedBootThreshold);
17     writer.WriteDoubleWordLE(value.BootCounter);
18     writer.WriteDoubleWordLE(value.LastPlannedRebootCounter);
19 }
```



oryx/packages/configuration > configuration/configuration/src/serialization.cpp

```
26     {
27         writer.WriteDoubleWordLE(value);
28         return true;
```

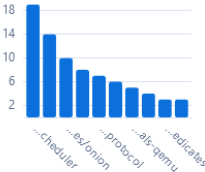


Filters

GROUP RESULTS BY ⓘ

Repository File Author

Capture group

 +7

Expand

SEARCH TYPES

Search repos by org or name

Find a symbol

Pełna lista

- Ansible
- Arek
- Badgen.net
- clang-format
- CMake
- CodeSaw
- Conan
- Doxygen
- Flake8
- GCC
- GitLab
- Hatch
- Hulk
- Jenkins
- Jira
- Kas
- Mypy
- Nexus
- Ninja
- OpenOCD
- Pipx
- Python
- QEMU
- Renovatebot
- Rocket.Chat
- Sourcegraph
- Sphinx
- Yocto

I co z tego wynika?

- Narzędzia i rozwiązania zbiera się przez lata
- Dopiero po czasie widać co się sprawdziło co nie
- Słuszna decyzja w T_1 może się okazać niekorzystna w T_2
 - wtedy trzeba ją zmienić
- KISS
- Automatyzacja, Automatyzacja, Automatyzacja
- "niech ktoś"

KONIEC

✉ mnowak@kplabs.pl

🐦 [@maciejt_nowak](https://twitter.com/maciejt_nowak)

📧 @novakov@mas.to

NADAL KONIEC

