Odp. Na Apel Twórcy j. C++ Bjarne Stroustrup

autor:	Jacek Marcin Jaworski
pseudonim:	Energo Koder Atlant
pomocnicy autora:	BRAK
miejsce:	Pruszcz Gd.
utworzono:	2025-03-24, pon.
wersja: 2123 z dnia:	2026-07-19
program składu:	Libre Office Writer
sys. op.:	Kubuntu
źródło:	energokod.gda.pl

Ten dok. w wer. PDF jest podpisany cyfrowo wolnym prog. GNU gpg dostępnym bezpłatnie ze s. www.gnupg.org/download. Instrukcja w j. pol. jak się posługiwać prog. GNU gpg w sys. Linuks z rodz. Debian/Ubuntu znajduje się w całkowicie bezpłatnym dok. PDF Konf. i Zabezp. Sys. Op. z Rodz. Ubuntu w roz. "Podpisywanie Dok. PDF".

Spis treści

Wstęp....1
1.1 Skróty....1
1.2 O Autorze....1
2 Rozw. Krok 1: Podstawa Wysokiej Jakości Prog. To Dok. Tech. i Testy Aut....2
3 Rozw. Krok 2: Zmiana Org. Pracy Nad Oprogramowaniem....3
4 Rozw. Krok 3: Zmiany w Kodowaniu: Konieczne Jest Spr. Pram. f. operator[](long long i)....3
5 Rozw. Krok 4: Fuzzery Sterowane SI Do Testów Aut....3
6 Rozw. Krok 5: Zmiany w Procesorach....3
7 Nonsensowna Argumentacja w Art. [tworca-c++-prosi-o-pomoc]....3
8 Podsumowanie....4
9 Bibliografia....4

Wstęp

Oto moja odp. na apel autora j. C++ Bjarne Stroustrup z Danii, w którym wzywa społeczność C++ do obrony tego j. prog. O jego apelu dowiedziałem się z art. [tworca-c++-prosi-o-pomoc].

1.1 Skróty

ADA	https://pl.wikipedia.org/wiki/Ada_(j%C4%99zyk_programowania)
art.	artykuł
b.	bardzo
d.	dzień
dok.	dokument
f.	funkcja
j.	język
kl.	klasa
kol.	kolejny
odp.	odpowiedź
org.	organizacja
ost.	ostatni
p.	punkt
pow.	powyższy
roz.	rozdział
s.	strona
spr.	sprawdzenie
sys. op.	system operacyjny
tab.	tablica
testy aut.	testy automatyczne
tyg.	tygodnia
wer.	wersja
wew.	wewnętrzna
wył.	wyłącznie
zad.	zadanie
zaw.	zawartość

Skrótów 4literowych i dłuższych nie tłumaczę, bo uważam je za oczywiste.

1.2 O Autorze

Od ur. w 1978 r. w Gd. żyję w Pruszczu Gd. na ul. Spacerowej. Od 2020-06-20 do 2021-06-19 mieszkałem w Gdyni na ul. Komandorskiej (dokładnie po roku wróciłem do Pruszcza Gd.).

Mój życiorys zawodowy jest dostępny na mojej s. WWW w dok. Najnowsze Podanie o Pracę.pdf. Natomiast pełen przebieg mojej kariery naukowej i zawodowej (łącznie ze sposobem wrobienia mnie w schizofrenię przez kol. z TŁ w Gd.) przedstawiam w 14. Raprot Totaliztyczny: Nauka i Praca Programisty C++ w III Rzeczy (pospolitej).pdf. W tym roz. podaję jedynie skrótowe info na mój temat:

Programuję komputery od lut. 1997 r. Od 1999 r. mam tytuł Technika Elektronika spec. Systemy Komputerowe.

Zaliczyłem 3 lata wieczorowych i nielegalnych studiów informy na PG (jednak dyplomu inż. nie zrobiłem). Legalną maturę zdałem w 2016 r. Jednak po 2005 r. wprowadzono w całej Polsce zmiany w prog. studiów zaocznych (wymuszono na wszystkich kier. 7 sem. z j. ang.) powoduje to, że nie jestem w stanie ich zaliczyć.

W latach 1999-2007 (jednak bez ciągłości zatrudnienia) byłem programistą aplikacji w C++ dla sys. Windows.

W latach 2017-2022 (jednak bez ciągłości zatrudnienia) byłem programistą aplikacji w C++ na sys. Linuks i Android.

Jeśli chodzi o mój kierunek rozwoju, to chcę być jak najlepszym inżynierem i architektem sys. i to nie tylko komputerowych.

Moje zainteresowania zawodowe to przede wszystkim:

✷Studia nad architekturą wzorcowych sys. komp.✷Studia nad nowymi algorytmami (jednak nie SI)✷Studia nad bezpieczeństwem współczesnych sys. komp.✷Studia nad prywatnością użytkowników współczesnych sys. komp.✷Podnoszenie jakości, efektywności i bezpieczeństwa w pracy przez realizację racjonalizatorskich projektów✷

✷Programowanie w językach: Asembler, C i C++.✷

Wystrzegam się jak mogę „produktów" wielkich korpo które z niejawnych (zło jest niejawne) powodów zwykle wynajdują patologiczne wynalazki[1].

Jestem poszukiwaczem i kolekcjonerem "dobrych zasad życiowych" i "dobrych zasad inżynierskich". Dzięki tym zwięzłym, hasłowym zasadom często widzę sens podejmowania większego wysiłku by uzyskać obiektywnie dobry efekt zamiast stosowania półśrodków.

Zdrowe zasady pozwalają zostawiać za sobą działające rozwiązania zamiast partactw.

Jestem praktykującym zwolennikiem filozofii dobra i moralności, czyli Totalizmu. Od końca 2001r. mam stały dostęp do Internetu. Z filozofią Totalizmu, w wyd. prof. Jana Pająka z NZ, po raz pierwszy zetknąłem się w Sieci Internet w I poł. 2002r., po wpisaniu w Google.pl hasła "ESP" - bez cudzysłoowu. Od razu b. się zafascynowałem filozofią Totlizmu i tak jest do d. dzisiejszego. Od 2020r. rozwijam jej popr. wer. w postaci Ideologii Geniuszy-Mocarzy. Mój Totalizm różni się od Totalizmu prof. Jana Pająka z NZ tym, że ja uważam że Totalizm składa się z dobra i z moralności, a nie z samej moralności. Wynika to z faktu, że sama moralność bez dobra prowadzi do filozofii moralności i zła czyli w istocie do filozofii nazistowskiej.

W latach 2004-2013 5x siedziałem w psychiatryku. Sądy i lekarze byli i są jednomyślni że jestem b. chory psychicznie, na przekór braku jakichkolwiek dowodów mojego rzekomego szaleństwa. Za "dowody" mojego szaleństwa lekarze uznają to, że otaczających mnie ludzi mam za "wrogich szatańskich pasożytów" i że po wzmocnionych dawkach leków psychotropowych (bo w szpitalach mogą dawać mocniejsze dawki) spałem całymi dniami - nazwali to "zespołem katatoniczno - paranoidalnym w przebiegu schizofrenii".

W d. 2025-06-17, wto. sąd rej. w Gd. skazał mnie na 6. odsiadkę w psychiatryku bez udowodnienia mi żadnych win oraz wcale nie odnosząc się do mojej obronnej argumentacji jasno wykazującej że to oskarżenie (personel domowy i medyczny) "ma coś z deklem" a nie ja. Sąd okręgowy w Gd., na posiedzeniu niejawnym, w d. 2025-10-27, pon. odrzucił moją apelację bez jej rozpatrzenia i podtrzymał wyrok sądu rej. Co dziwne tego wyroku na razie nikt nie wyegzekwował (piszę to w d. 2026-06-26, pią.). Przejawy mojej dyskryminacji oraz odrzuconą apelację do sądu okręgowego przytaczam w: 2025-10-06 Do Amnesty International Polska - Prośba o Pomoc Dla Prześladowanego Jacka Marcina Jaworskiego.pdf.

Od 2024-12-20, pią. prowadzę moją domową s. WWW: energokod.pl. Od 2025-11-07, pią. jest ona dostępna pod nowym adresem energokod.gda.pl. Na tej s. WWW pub. kilkadziesiąt dok. PDF o tematyce totaliztycznej, Mini Netykietę i moje dane kontaktowe. Nowości na mojej domowej s. WWW można śledzić dzięki plikowi/kanałowi energokod.gda.pl/rss.xml.

2 Rozw. Krok 1: Podstawa Wysokiej Jakości Prog. To Dok. Tech. i Testy Aut.

Moim zdaniem nie ma "cudownego sposobu" na eliminację błędów w kodach prog. pisanych przez ludzi. Można za to wskazać sposoby ich wykrywania i poprawiania.

Najważniejszymi narzędziami programisty pomocnymi w prog. wysokiej jakości kodu są: specyfikacja techniczna i testy automatyczne. Moim zdaniem wszelkie problemy z atakami typu "buffer-overflow" wynikają albo z nieprecyzyjnej specyfikacji technicznej, albo ze zbyt słabych te-

---

1 Jak wiecie język **C** i sys. **Uniks** stworzyło 2 ludzi: **Dennis Ritchie** i **Ken Thompson**. Dlatego jestem na 100% pewien, że gdyby ich projekt był prowadzony z rozmachem w stylu wielkich korpo, to: a) Język **C** nigdy by nie powstał, b) **Uniks** nigdy by nie powstał, c) powstały by potwory podobne do **Ada**, **Jawa**, **C#**, **Windows** i **Android**.
Jak zauważył prof. Jan Pająk z NZ: W organizacjach pasożytniczych wszystkie złe pomysły pojawiają się od górnie, a wszystkie postępowe koncepcje powstają oddolnie.

stów aut. I nie ma to związku z żadnym j. prog. Po prostu niedbałe prog. bez ich dopracowania i tak będą niedbałe i dziurawe. Dlatego może się okazać konieczne wydłużenie procesów projektowania i testowania oprogramowania, tak by nadać mu wysoką jakość. Na pewno należy dopracowywać wszelkie formaty i protokoły stosowane w sieci Internet.

3 Rozw. Krok 2: Zmiana Org. Pracy Nad Oprogramowaniem

Potrzebna jest metodyczne podejście do testowania. W tym celu trzeba dynamicznie przydzielać zadania mające na celu pełne przetestowanie kodu proj. Dlatego proponuję nowe rozw. org., cytat z mojej [arch-prog-nieup], roz. „Sposób wykrywania błędów":

„5. Podczas testów jednostkowych:

W ramach testów aut. testujemy: 1) Wart. typowe, 2) Skrajne i 3) Zabronione w dok. technicznej.

W czwartek po południu tester aut. podaje zad. testowe do realizacji w pią. Co pią. wszyscy programiści uzupełniają testy aut. i wzmacniają spr. niezmienników i testy kontraktowe. Tak samo w ost. tyg. (ost. 5 d. roboczych) każdego mieś."

4 Rozw. Krok 3: Zmiany w Kodowaniu: Konieczne Jest Spr. Pram. f. operator[] (long long i)

Obecna polityka polegająca na nie sprawdzaniu param. f. std::vector::operaotr[](long long i) to po prostu skrajna nie odpowiedzialność. Jeśli chodzi o spadek wydajności prog., to on i ma on miejsce również w innych sytuacjach: np. Position-independent code [pic], albo przy f. wirt., albo przy samym dostępie do kl. kontenerowych za pomocą f. iteratorów, a nawet same f. operator stanowią narzut w porównaniu do dostępu do prostych tab. (trzeba jednak zaznaczyć, że w przypadku iteratorów i f. operator mogą to być też szybkie f. wstawiane). Rzucanie wyjątku gdy indeks Jest większy od wielk. Wektora jest najbardziej naturalne. To powinno być standardowo włączone w kl. std::vector.

5 Rozw. Krok 4: Fuzzery Sterowane SI Do Testów Aut.

Fuzzery je rozumiem jako prog. Które dostają prawidłowy pakiet danych wej. i próbują znaleźć dziurę stopniowo psując ten pakiet wej. Mi się wydaje, że można je znacznie zoptymalizować sterując inteligentnie. Okazuje się, że istnieje już taki fuzzer: „American Fuzzy Lop" dostępny https://github.com/AFLplusplus/AFLplusplus. Ja go jeszcze nie używałem, ale wydaje się to b. dobrym p. wyjścia do inteligentnych testów aut.

6 Rozw. Krok 5: Zmiany w Procesorach

Dostęp do wszystkich tab. powinien się dokonywać za pomocą nowej instrukcji asemblerowej z 2 parametrami: 1) pocz. tab. i 2) indeks w tab. Wtedy wykorzystując to, że przed każdą tab. jest już umieszczona stała z jej wielk. można sprzętowo spr. czy nie przekroczono rozmiaru tab. W takiej sytuacji powinien następować wyjątek procesora i przekazanie kontroli do procedury obsługi tego wyjątku. Działać to ma podobnie do PIC (adresy względne w programach) i wyjątków koprocesora (przy jego braku do jego programowej emulacji – tak było w przypadku procesorów Intel 386 bez opcjonalnego koprocesora matematycznego i w przypadku tanich procesorów Intel 486SX z wył. uszkodzonym koprocesorem matematycznym). Te wyjątki wynikające z braku koprocesora wykorzystywały wczesne wer. rdzenia sys. op. Linuks (w ten sposób działała programowa emulacja koprocesora matematycznego).

7 Nonsensowna Argumentacja w Art. [tworca-c++-prosi-o-pomoc]

cytat z art. [tworca-c++-prosi-o-pomoc]:

"Presja ze strony regulatorów
Nie bez znaczenia są także zmiany regulacyjne. Amerykańska CISA opublikowała w październiku ubiegłego roku raport zalecający, by do 1 stycznia 2026r. producenci mieli plan eliminacji luk pamięciowych lub przeszli na języki bezpieczne dla pamięci. Stroustrup uznał to za "realne zagrożenie" dla C++.
Rząd USA proponuje następujące języki:
Rust
Go
C#
Java
Swift
JavaScript
Ruby"

Pow argumentacja jest nonsensowna z tego powodu, że na tej liście tylko Rust i Go są j. prog. a reszta to j. skryptowe. Obie grupy j. są nieporównywalne i mają inne zastosowania.

8 Podsumowanie

Moja końcowa myśl jest taka, że wygląda na to że trafne są moje wcześniejsze przewidywania, które opublikowałem w [f-35-zasady-kodowania] i że faktycznie rząd SZAP/USONA chce wycofać z rynku cywilnego j. C++ i że chcą zrobić z niego kol. "czarny proj.". Bo na przekór pow. propagandzie SZAP/USONA uruchamia kol. proj. militarne oparte o prog. kodowane w j. C++ (np. patrz art. w Nowej Technice Wojskowej nr 2024/11, [ntw-2024-11] cytat s. 45: *„I tak zmiany wprowadzone w pierwszym z wariantów pocisku JASSM-ER [prawdopodobnie chodzi o AGM158 B2 – przyp. JMJ] dotyczyć mają przede wszystkim wymiany komponentów teraz trudnodostępnych, wychodzących z produkcji czy też po prostu przestarzałych (według obecnych standardów). Systemy komputerowe pocisku otrzymały ponadto oprogramowanie, stworzone w języku C++, które zastąpiło wcześniej stosowane, napisane w języku ADA. Pocisk w wariancie AGM-158B-2 wyposażono również w zmodernizowany komputer pokładowy (Missile Control Unit, MCU), dysponujący zwiększoną mocą obliczeniową.”*).

Znamy to z przeszłości: wcześniej tak było np. z sys. op. Plan9, który był stworzony przez twórców sys. op. Unix. Sys. op. Plan9 był genialny i przełomowy, jednak został porzucony. Co było kol. proj. twórców sys. op. UNIX? Pozostaje tajemnicą bez odp. Jednak ja się domyślam: następca sys. op. Plan9 należał do "czarnych proj." czyli tajnych i to zazwyczaj na zawsze.

Innym przykładem wycofywania rozw. z cywilnego rynku są dżojstiki używane przez domowych graczy komputerowych w latach 80 i 90 XXw. Teraz dżojstiki są używane wyłącznie w wojsku. Natomiast obecnie w domach do gier komputerowych używa się nieergonomicznych padów.

9 Bibliografia

Bibliografia

tworca-c++-prosi-o-pomoc: Paweł Czajkowski, Twórca C++ prosi o pomoc. Przyszłość języka zagrożona przez rząd USA i Rust, 2025, https://ithardware.pl/aktualnosci/tworca_c_jezyk_zagrozony_usa_rust-39456.html

arch-prog-nieup: Jacek Marcin Jaworski, Arch. Prog. Nieuprzywilejowanych, 2026, https://energokod.gda.pl/monografie/Arch.%20Prog.%20Nieuprzywilejowanych.pdf

pic: , Position-independent code, 2025, https://en.wikipedia.org/wiki/Position-independent_code

f-35-zasady-kodowania: Jacek Marcin Jaworski, F-35 - Zasady Kodowania - komentarz, 2026, https://energokod.gda.pl/komentarze-do-ksiazek/2025-01-19%20F-35%20-%20Zasady%20Kodownia%20-%20komentarz.pdf

ntw-2024-11: Michał Gajzler, JASSM niejedno ma imię, czyli AGM-158B-2, B-3, D oraz LRASM i JASSM-XR, 2024