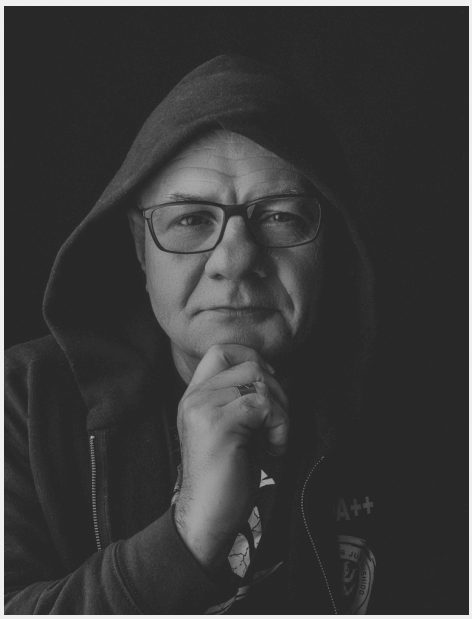
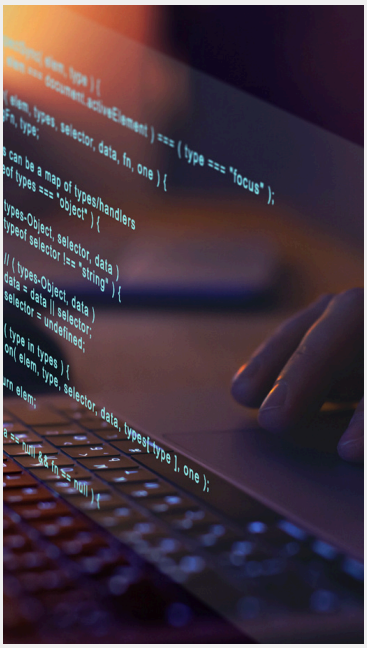
Requiem for the VFP ? I don't think so.

13 lat temu w 2012 roku oprogramowanie mojego autorstwa pracowało w dużych przedsiębiorstwach segmentu piekarskiego. Jest to oprogramowanie którego geneza sięga roku 1998r. Oprogramowanie to było napisane w języku c++ oraz Clipper 5.2E.



W tamtych czasach jedyną rozsądna dla mnie (zachowałem bardzo dużo kodu źródłowego z Clipper) alternatywą dla przenosin do świata 32bit był oczywiście Visual FoxPro firmy Microsoft. Język w zasadzie taki sam jak Clipper z bardzo niewielkimi zmianami. Bardzo szybko przekonałem się co to narzędzie potrafi jeśli chodzi o wydajność. Zarazem pojąłem dlaczego firma M$ kupiła od Ashton-Tate FoxPro w 1992. Kupiła zaawansowaną matematykę tkwiącą w jego motorze bazodanowym, a którą teraz możemy podziwiać w kolejnych odsłonach MSSQL.



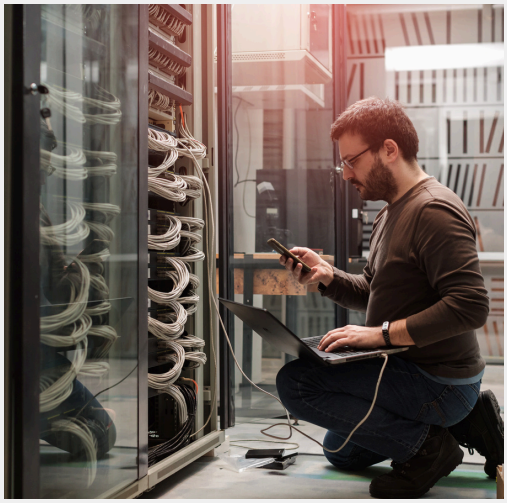
Jako że w 1998r nie było jeszcze MSQL natomiast SQL 7 firmy Microsfot (pamiętacie to jeszcze ?) wzbudzał we mnie tylko politowanie gdy testowałem jego wydajność, zdecydowałem się na VFP oraz pliki DBF/CDX. Już wtedy bariera 2GB na rozmiar pliku była dla mnie sporym problemem jednak miałem nadzieję, że w niedługim czasie firma M$ złamie tę i inne bariery mocno doskwierające już wtedy VFP (brak możliwości budowania natywnych usług, brak wbudowanej wielowątkowości itd.). Miałem nadzieję, że w niedługim czasie firma VFP usunie te bariery. Jak wiemy nie usunęła a produkt ubiła w 2007 roku, tak jak wiele innych które kupiła po to by przejąc technologię a sam produkt zakończyć by udawać, że to ich osiągniecie inżynierskie. Powyższe niedogodności zostały oczywiście rozwiązane przy użyciu bibliotek napisanych w c++ czyli vfp2c32. źródło: Christian Ehlscheid **https://github.com/ChristianEhlscheid/vfp2c32**

W roku 2012 doszedłem do ściany moich możliwości technologicznych jeśli chodzi o pliki DBF/CDX. Mój największy klient miał w tym okresie około 70 użytkowników pracujących w centrali na aplikacjach typu desktop.

Oraz ponad 300 różnych stanowisk w sieci WAN, głownie POS, które również kilkadziesiąt razy na godzinę komunikowało się z centralą.

Przeglądając obecne wtedy na rynku języki oprogramowania natknąłem się na kompilator niemieckiej firmy Alaska Software | Xbase++.

https://www.alaska-software.com/.



Język ten bardzo podobny (ta sama rodzina języków XBASE) do VFP ma wiele zalet, że wymienię tylko niektóre z nich.

- ✓ jest bardzo szybki i stabilny (Alaska Software to bezpośrednia nakładka na c++).
- ✓ rewelacyjna wielowątkowość.
- ✓ bardzo mocne natywne wsparcie dla TCP/IP zestaw funkcji umożliwiających budowanie dowolnych usług TCP/IP oraz UDP.
- ✓ rewelacyjna współpraca z serwerami COM/DCOM VFP.
- ✓ bardzo dobre wsparcie dla Postgresql.

Za pomocą tego narzędzia zbudowałem usługę TCP/IP która wykonuje każde polecenie VFP po stronie serwera. Czyli klient-serwer dla plików DBF/CDX ale nie tylko. Gdyż w chwili obecnej możliwa jest obsługa następujących motorów bazodanowych:

✓DBF/CDX ✓MSSQL ✓Postgresql ✓Oracle ✓MySql ✓SQLite

Zapytania piszemy w SQL który znamy z VFP. W przypadku gdy w danym silniku bazodanym występują różnice w składni języka SQL, Foxx-i Server sam dokonuje translacji na daną odmianę języka SQL. Warunkiem jest jednakże 100% zgodność składni zapytań z VFP.
Dzięki temu zbudowałem aplikację która bez zmiany ani jednej linijki kodu może pracować na wszystkich wymienionych wyżej silnikach bazodanowych.

Usługa napisana w Alaska Software (deamon TCP/IP) ma za zadanie odbierać przychodzące połączenia z sieci lokalnej oraz WAN. Wątkami roboczymi są serwery COM Visiual FoxPro i może ich być dowolna ilość.
Foxx-i Server sam sobie dobiera optymalną ilość wątków roboczych w zależności od obciążenia.
Po stronie serwera można wykonać dowolną instrukcję SELECT, UPDATE, DELETE ect. wywołać dowolną procedurę składowaną, bądź za pomocą natywnej funkcji VFP EXECSCRIPT() wykonać dowolny kod VFP z ekstremalną szybkością znaną z VFP po stronie serwera.

W 2019 roku w oparciu o Foxx-i Server zbudowałem swój własny model AI. Ada++.
Jako podstawową bazę danych dla moich sieci neuronowych wybrałem VFP z tego względu, że ani MSSQL ani Postgresql nie zapewniały mi odpowiedniej wydajności w zapytaniach które stosuję, a VFP tak. Dzięki temu jestem w stanie błyskawicznie generować prognozy sprzedaży na 7 dni do przodu uwzględniając 23 różnych kryteriów. U największego mojego klienta który ma kilkaset sklepów własnych estymacja która zawiera prognozę zamówień na 7 dni do przodu wykonuje się w niecałe 30 min i nie potrzeba do tego koparek NVIDIA. Daje mi to niesamowitą elastyczność np. w przypadku dynamicznie zmieniającej się prognozy pogody, świąt itd.
Algorytm ten w chwili obecnej samodzielnie zamawia towar dla sieci sklepów własnych o wartości **1 mln EUR dziennie.**

https://infopiek.pl/en/offer/ada/

oraz Ada++.AntiFraud. Algorytm który pilnuje by rozliczenia finansowe w sklepie zgadzały się.

https://infopiek.pl/en/offer/ada-antifraud/

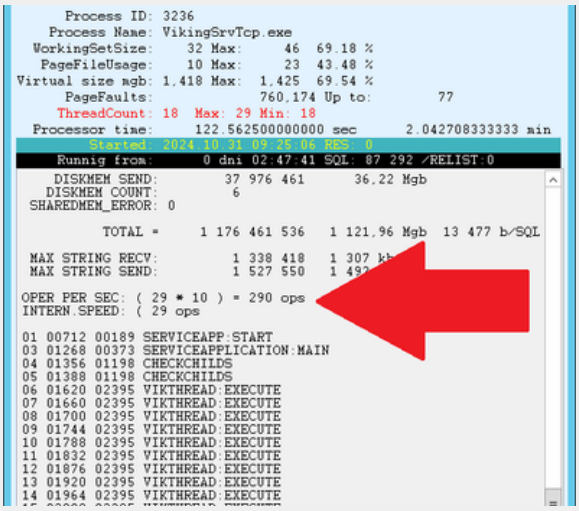
Foxx-i Server w chwili obecnej zwraca dane z zapytań w następujących formatach:

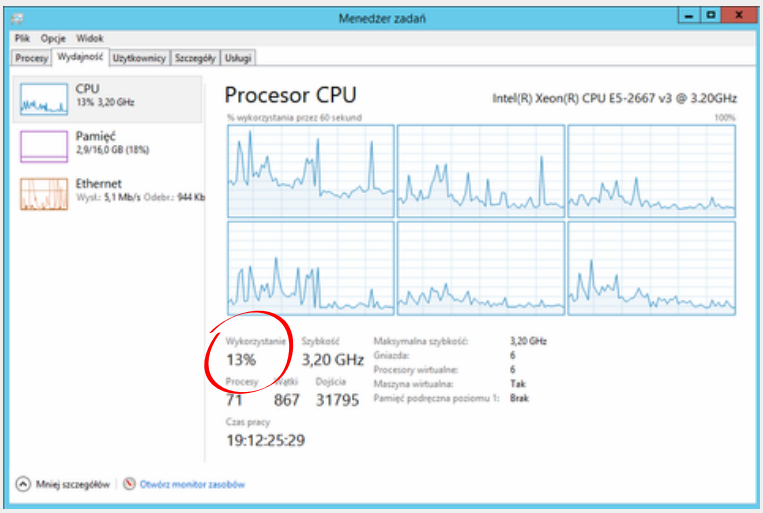
- ✓ Foxx-i Server format
- ✓ JSON
- ✓ XML
- ✓ CSV
- ✓ DBF
- ✓ TXT
- ✓ XLS
- ✓ HTML

Poniżej zrzuty ekranu z produkcyjnej instalacji Foxx-i Server jednego z naszych klientów..

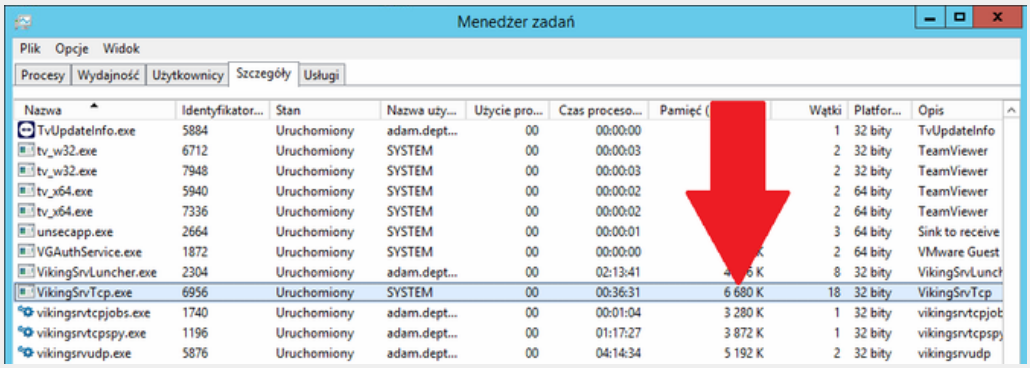
Średnia wydajność tego serwera wynosi 290 operacji zapisu/odczytu na sekundę.

Procesor Xeon(R) CPU E5-2667 v3 3.2 GHz, w chwili gdy piszę te słowa jest wart 10 USD a obciążenie procesora bardzo rzadko przekracza 23 %.

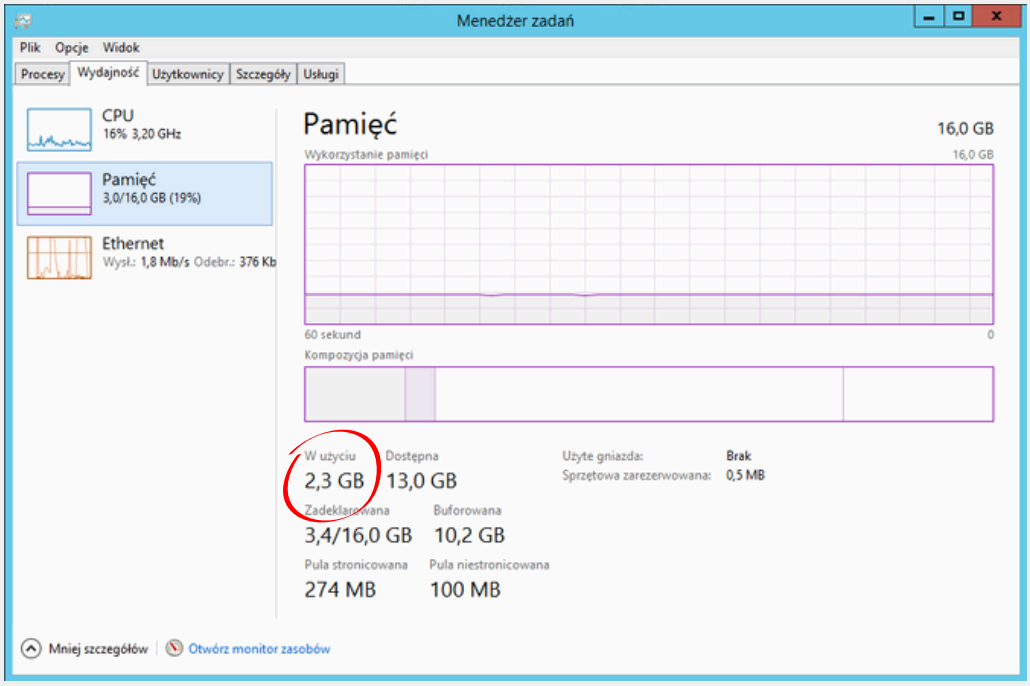




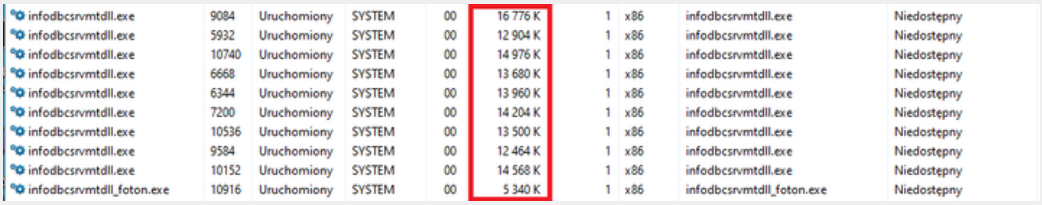
Sama usługa Foxx-i Server zajmuje w pamięci serwera zaledwie 10 mgb RAM.



Cały ekosystem rozwiązania na serwerze zużywa zaledwie 2.3gb RAM



10 wątków roboczych VFP w technologii COM zajmuje w pamięci zaledwie około 112 mgb RAM.



infodbcsrvmtdll.exe	9084	Uruchomiony	SYSTEM	00	16 776 K	1	x86	infodbcsrvmtdll.exe	Niedostępny
infodbcsrvmtdll.exe	5932	Uruchomiony	SYSTEM	00	12 904 K	1	x86	infodbcsrvmtdll.exe	Niedostępny
infodbcsrvmtdll.exe	10740	Uruchomiony	SYSTEM	00	14 976 K	1	x86	infodbcsrvmtdll.exe	Niedostępny
infodbcsrvmtdll.exe	6668	Uruchomiony	SYSTEM	00	13 680 K	1	x86	infodbcsrvmtdll.exe	Niedostępny
infodbcsrvmtdll.exe	6344	Uruchomiony	SYSTEM	00	13 960 K	1	x86	infodbcsrvmtdll.exe	Niedostępny
infodbcsrvmtdll.exe	7200	Uruchomiony	SYSTEM	00	14 204 K	1	x86	infodbcsrvmtdll.exe	Niedostępny
infodbcsrvmtdll.exe	10536	Uruchomiony	SYSTEM	00	13 500 K	1	x86	infodbcsrvmtdll.exe	Niedostępny
infodbcsrvmtdll.exe	9584	Uruchomiony	SYSTEM	00	12 464 K	1	x86	infodbcsrvmtdll.exe	Niedostępny
infodbcsrvmtdll.exe	10152	Uruchomiony	SYSTEM	00	14 568 K	1	x86	infodbcsrvmtdll.exe	Niedostępny
infodbcsrvmtdll_foton.exe	10916	Uruchomiony	SYSTEM	00	5 340 K	1	x86	infodbcsrvmtdll_foton.exe	Niedostępny

Każdego dnia usługa Foxx-i Server odpowiada za zapisywanie, odczyt oraz edycję około 5 500 różnych dokumentów oraz kilkudziesięciu tysięcy paragonów fiskalnych z POS. Natomiast wszystkie instalacje Foxx-i Server w Polsce, w chwili obecnej zapisują dziennie ponad pół miliona paragonów. Foxx-i Server wykonuje średnio 350 tyś operacji bazodanowych, z wydajnością 290 operacji na sekundę. Przy praktycznie zerowym USD nakładzie dla klienta za oprogramowanie bazodanowe.

Adam Deptuła