

A decorative graphic on the left side of the slide, consisting of a network of thin, light blue lines and small circles, resembling a circuit board or a neural network, extending from the top to the bottom of the frame.

DON'T PANIC

HITCHHIKER'S GUIDE TO THE JVM MONITORING

O CZYM BĘDZIEMY MÓWIĆ

- Gdzie możemy napotkać problemy
- Jak diagnozować przyczyny problemów
- Jakich narzędzi użyć
- Jak ich użyć

The background is a dark blue gradient. In the corners, there are white, stylized lines resembling circuit traces or neural network connections. These lines end in small circles, some of which are connected to each other, forming a sparse network pattern.

KTO BĘDZIE MÓWIĆ?

The background is a dark blue gradient. In the corners, there are decorative white lines resembling circuit traces or a stylized network. These lines connect small white circles, some of which are larger than others. The lines are more dense in the top-left and bottom-left corners and more sparse in the top-right and bottom-right corners.

GDZIE MIESZKAJĄ PROBLEMY?

GDZIE MIESZKAJĄ PROBLEMY?

Niewydajne algorytmy

Niewłaściwe
kolekcje

Niewydajny
model danych

Kod

Wycieki pamięci

GDZIE MIESZKAJĄ PROBLEMY?

Pule wątków

Pule połączeń

Kontener

Kod

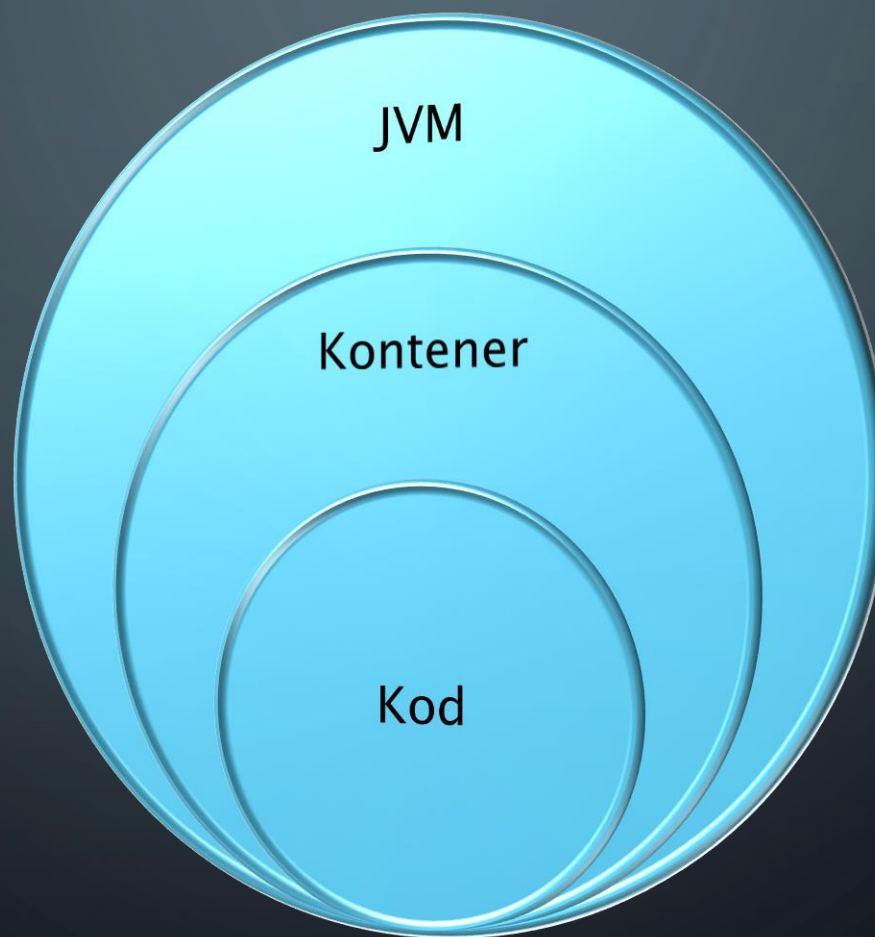
Cache



GDZIE MIESZKAJĄ PROBLEMY?

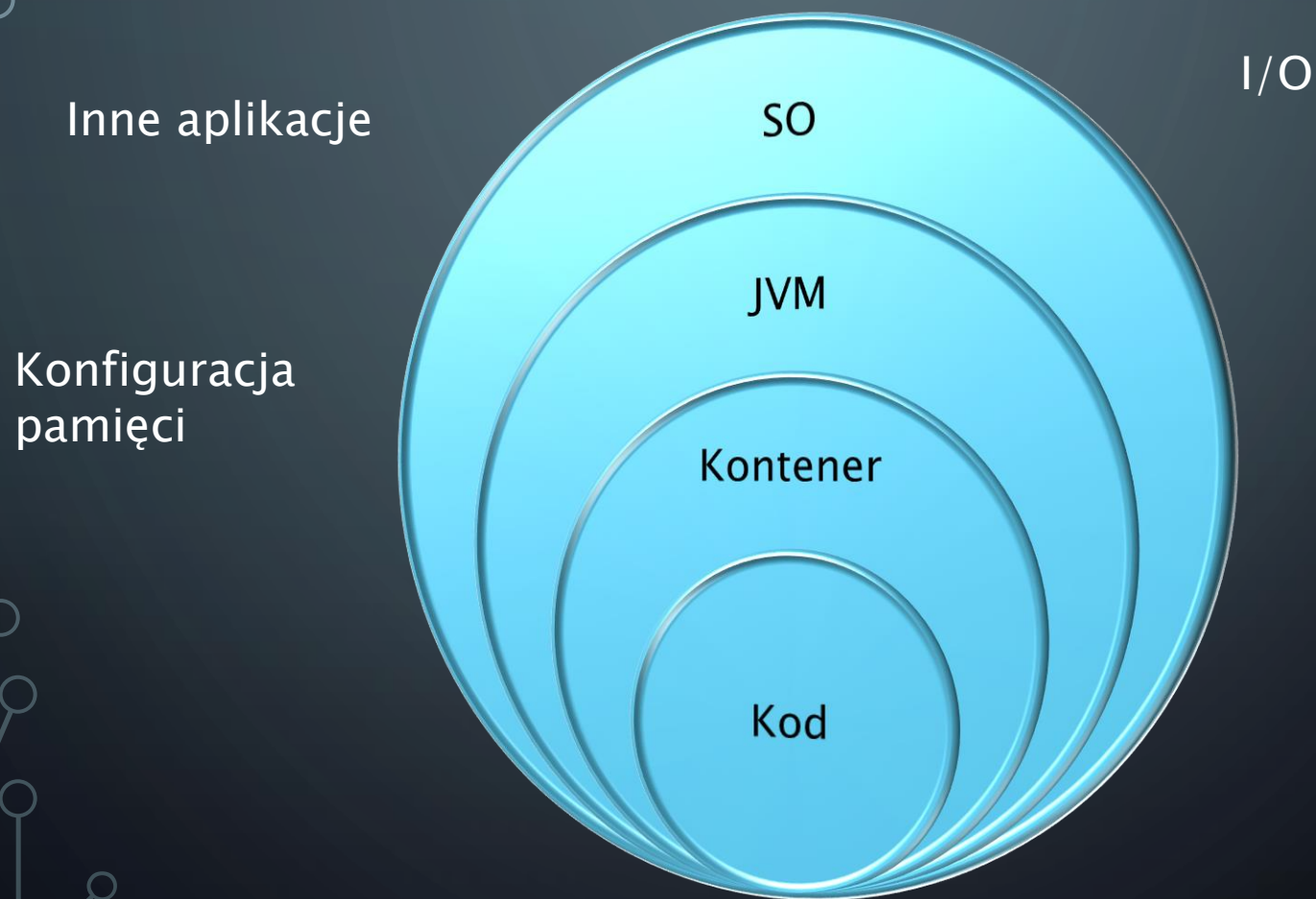
GC i
zarządzanie
pamięcią

Lock'i i
synchronizacja

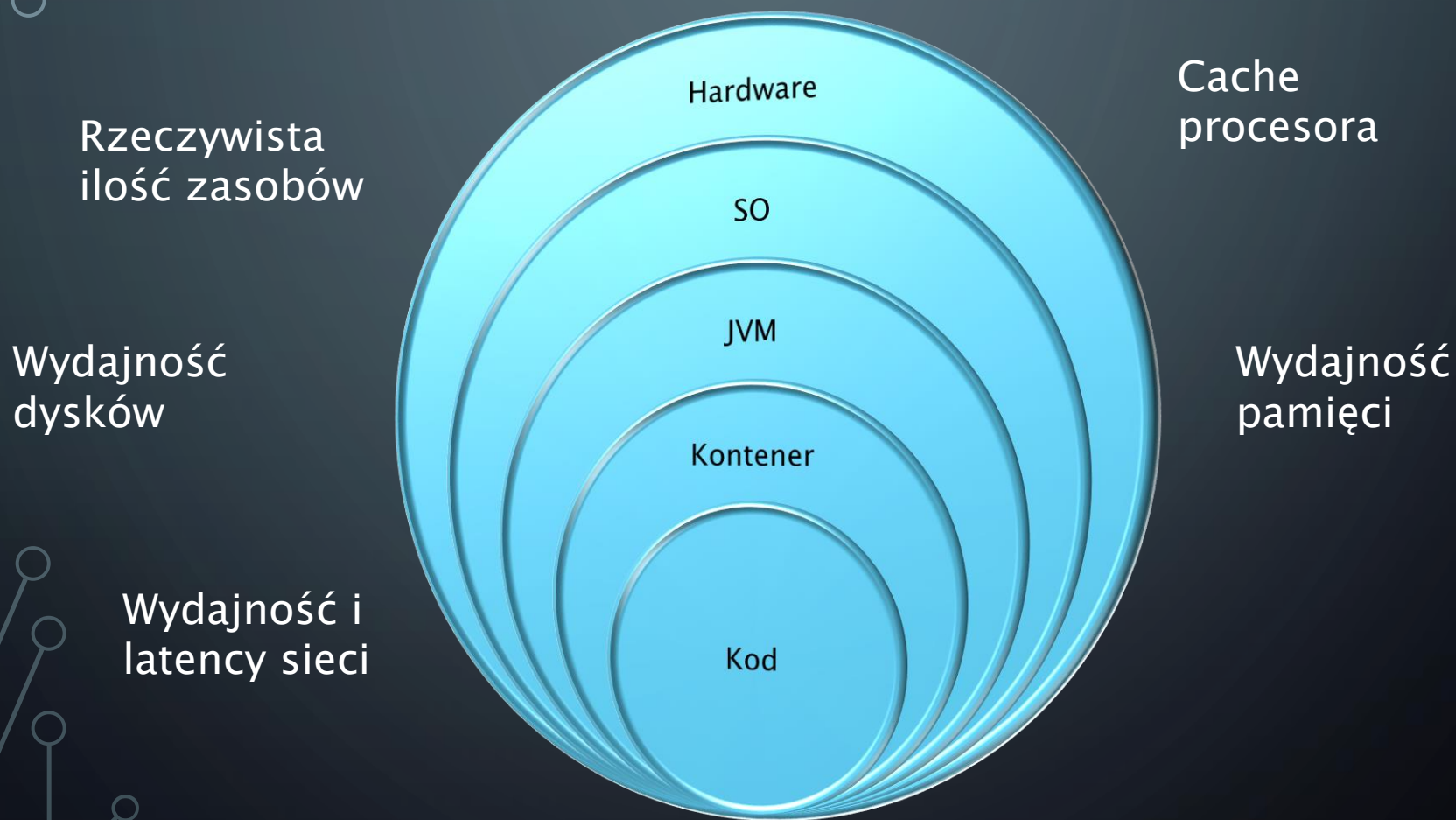


JIT

GDZIE MIESZKAJĄ PROBLEMY?



GDZIE MIESZKAJĄ PROBLEMY?



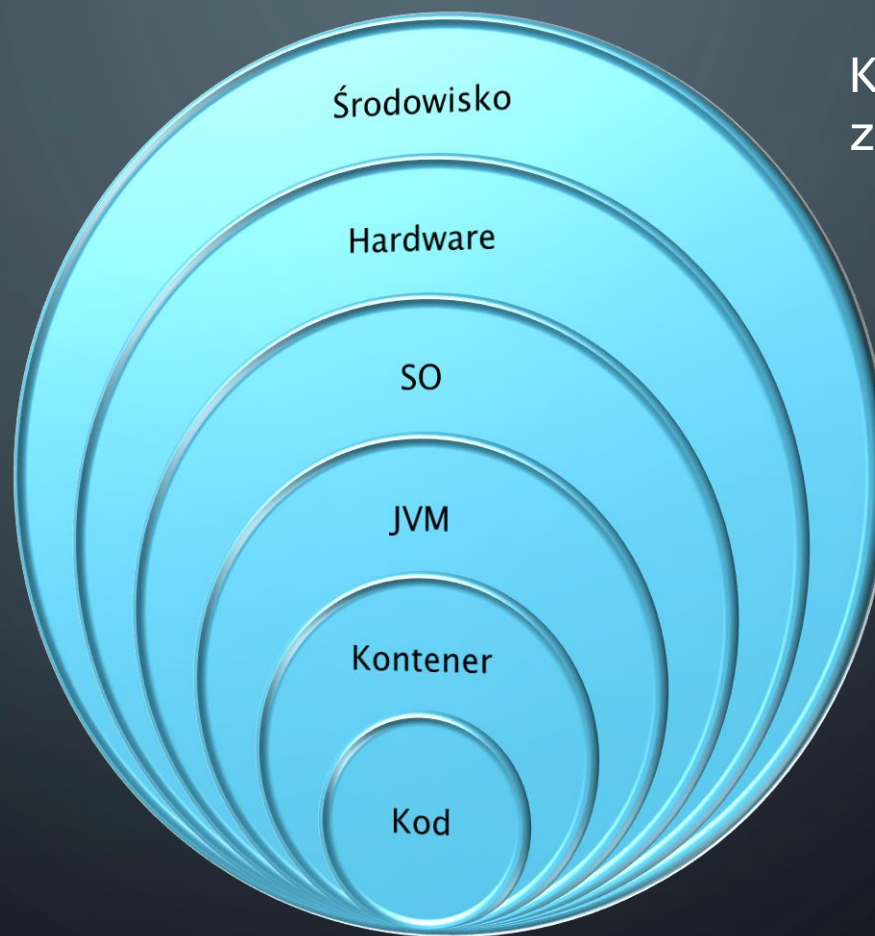
GDZIE MIESZKAJĄ PROBLEMY?

Komunikacja
z systemami
zewnętrznymi

Komunikacja
z IDM

Komunikacja
z bazą danych

Komunikacja z
przeglądarką



WNIOSKI

- Szukanie problemów w kodzie jest jak szukanie igły w stogu siana.
- Warto sięgnąć po narzędzia które poprowadzą nas dokładnie do źródła problemu.
- Czasami problemy nie wynikają z kodu a z konfiguracji całego otoczenia aplikacji.

NARZĘDZIA

- Monitorowanie systemu operacyjnego
 - top
 - vmstat
 - iostat
 - iotop
 - strace
 - pidstat

NARZĘDZIA

- Monitorowanie JVM:

- Pamięć:

- jmap
 - jhat
 - Eclipse MAT

- GC:

- jstat
 - gcviewer

- Kod:

- jstack
 - TDA
 - Profilery

- Ogólne:

- jcmd
 - jinfo
 - jconsole
 - jvisualvm
 - JMission Control

- APM:

- Dynatrace
 - AppDynamics
 - Plumb

CZAS NA ...

DEMO

PODSUMOWANIE

- Problemy nie wynikają jedynie z kodu.
- Szybka identyfikacja przyczyn problemów wymaga sięgnięcia po właściwe narzędzia.
- W razie problemów sprawdź:
 - Czy to problem SO (wysoki system time, wysoki iowait)? – <vmstat, iostat, top>
 - Czy to problem z GC i pamięcią (częste FGC, wysokie zużycie procesora, freeze aplikacji, OOM)? – <jstat, top, vmstat, gcviewer, jvisualvm, jmap, mat>
 - Czy to problem z Lockami (duża ilość zablokowanych wątków w zrzucie stosu wątków, niskie zużycie procesora)? – <jstack, thread dump analyzer, jvisualvm, top, vmstat>
 - Gdzie system spędza najwięcej czasu? (profiler, zrzut stosu wątków) <jstack, , jtop, jvisualvm, jprofiler, yourkit>

DODATKOWE INFORMACJE

- Kirk Pepperdine – „Live Java Performance Troubleshooting”

<https://vimeo.com/35387464>

- Leonid Igolnik – "The dark art of performance tuning or how to become a performance hero ..."

<https://www.youtube.com/watch?v=dqg0R3gYGac>

- Scott Oaks – „Java Performance: The Definitive Guide – Getting the Most Out of Your Code”

42?



SO LONG
AND
THANKS
FOR ALL
THE FISH