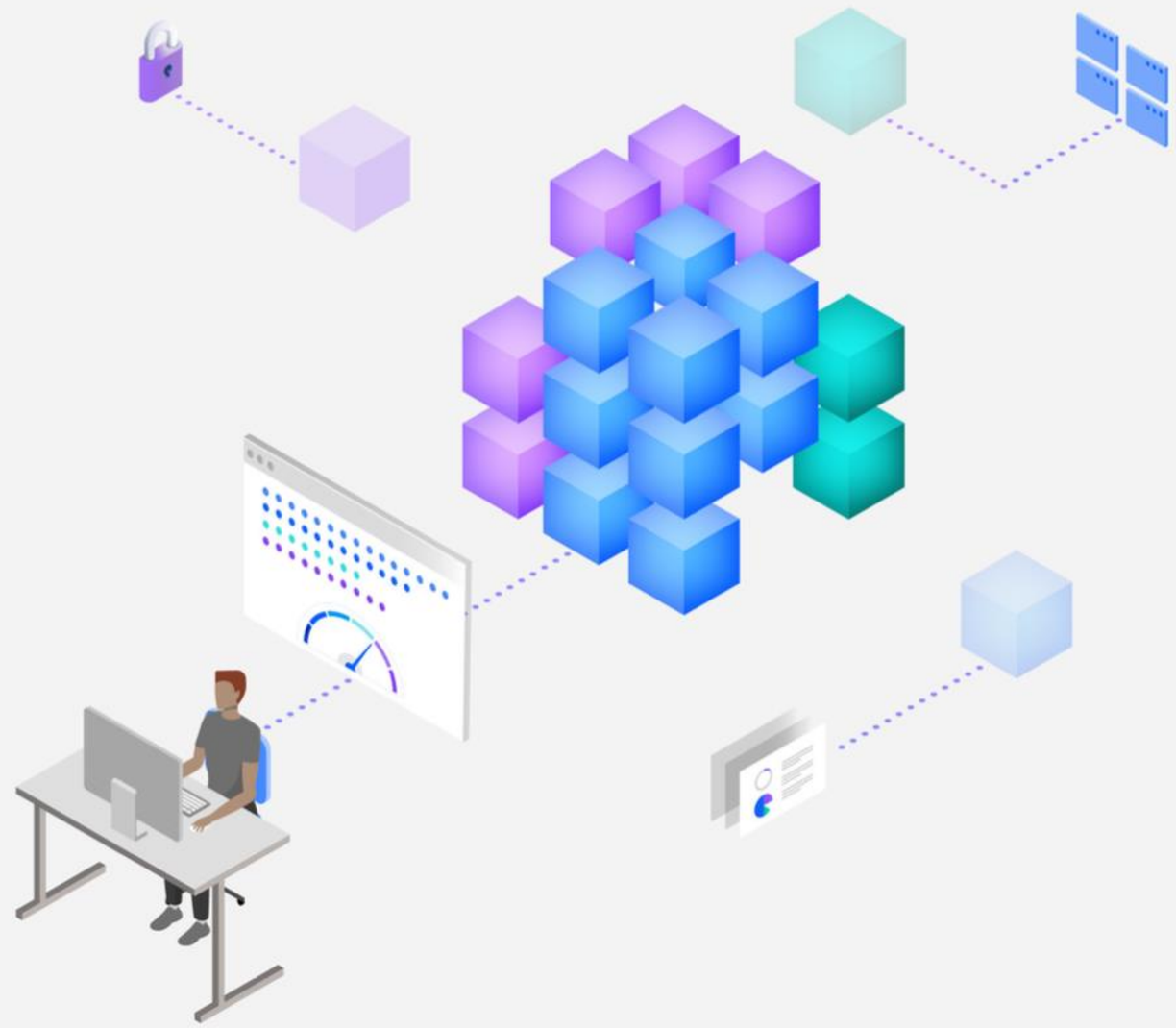
Modernizacja aplikacji z Mono2Micro

Mikołaj Jaworski
Partner Technical Specialist @ IBM



Jak narzędzia modernizacyjne mogą pomóc?

Modernizacja architektury

IBM Mono2Micro

Planowanie
(rady i rekomendacje)

Jak może wyglądać aplikacja:

- Dekompozycja aplikacji w ramach granic funkcjonalności i/lub potrzeb biznesowych
- Przekształcenie w zbiór mikroserwisów.

Działanie
(wsparcie w modernizacji)

- Generowanie kodu umożliwiającego komunikację pomiędzy mikroserwisami Liberty.

IBM Mono2Micro

Zapewnia możliwość maksymalnego wykorzystania istniejącego kodu Java przy jednoczesnej transformacji aplikacji na mikrousługi.

Monolith

Silnik analizujący istniejące aplikacje.

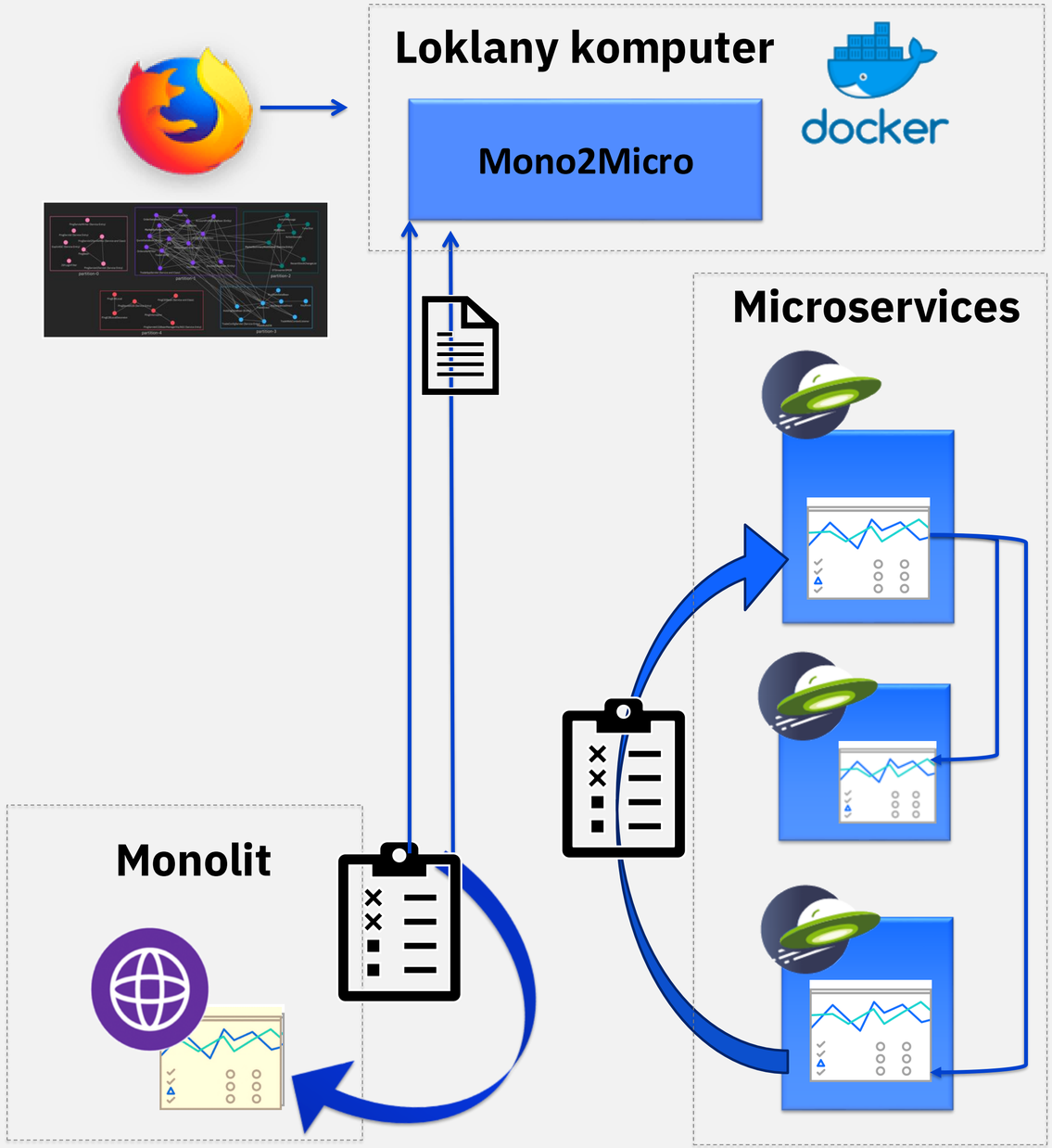
Sztuczna inteligencja identyfikuje komponenty o wysokiej spójności i łatwym połączeniu.

Microservices

Generowanie startowego kodu mikrousług do wdrożenia w Liberty.

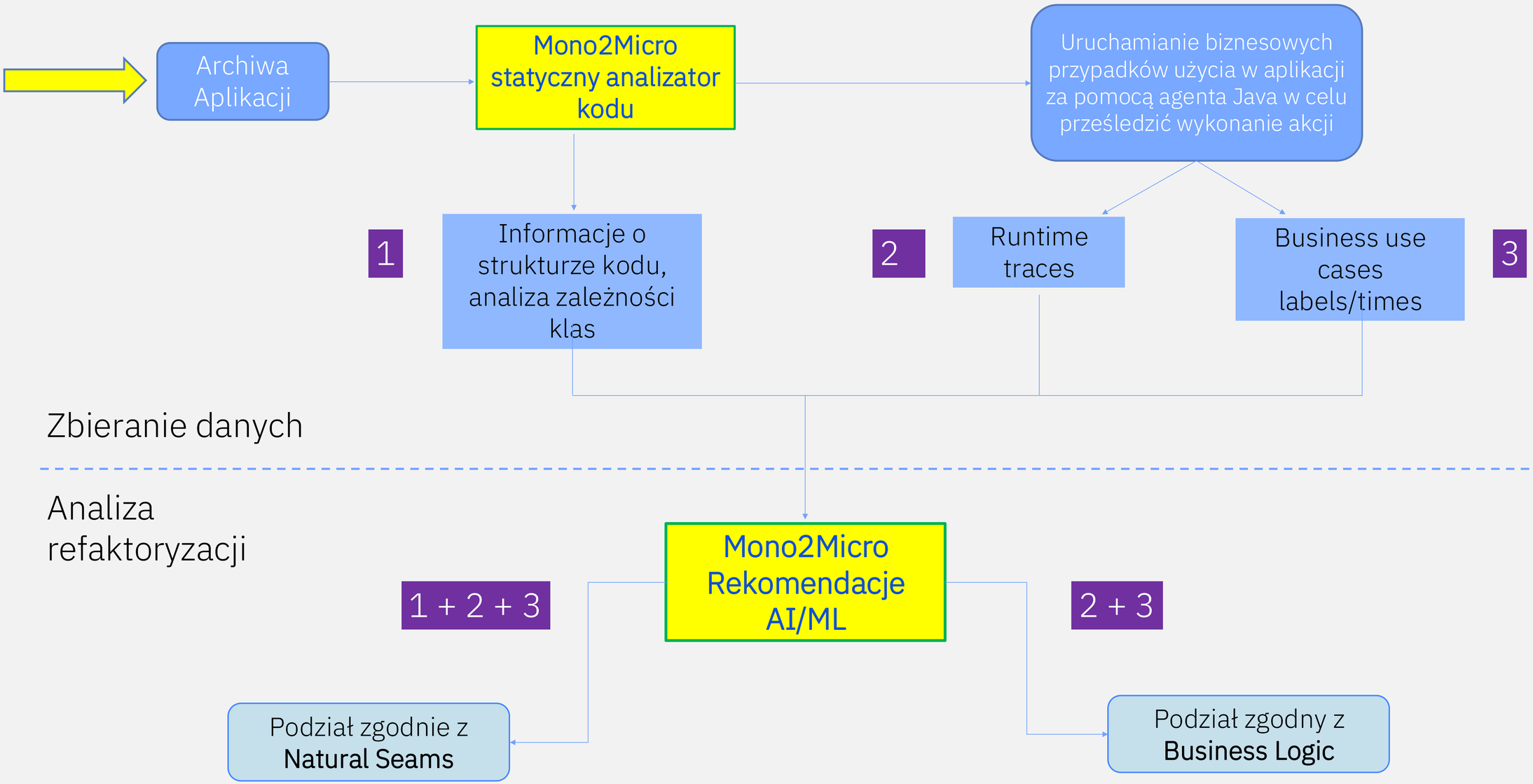
Transformacja wspomagana sztuczną inteligencją przy zmniejszonym ryzyku i niższych kosztach.

Jak IBM Mono2Micro działa?

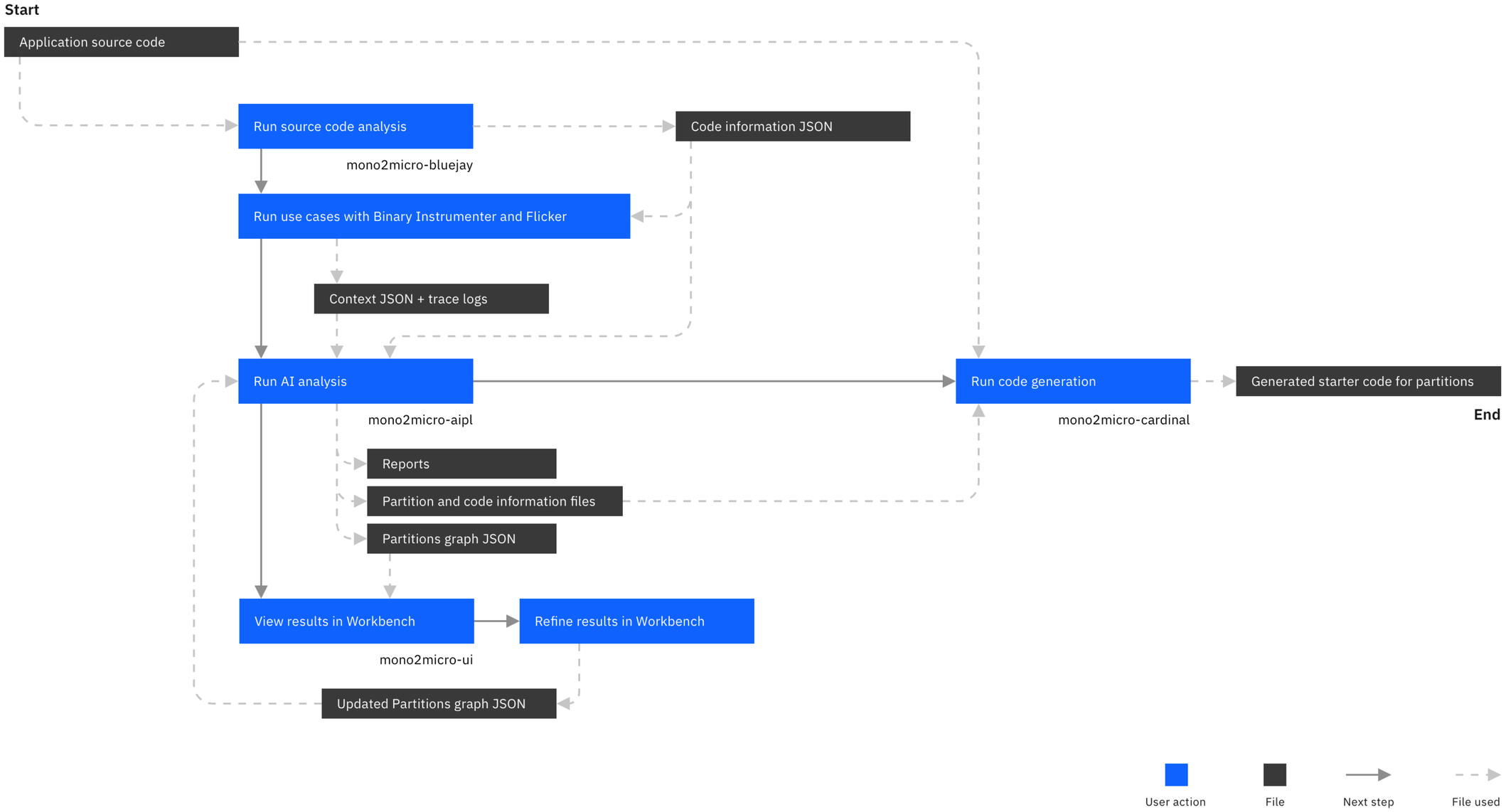


1. Zainstaluj Mono2Micro na swoim komputerze.
2. Mono2Micro analizuje i opracowuje kod źródłowy monolitu.
3. Wykonaj przypadki testów biznesowych w odniesieniu do aplikacji monolitycznej.
4. Sztuczna inteligencja Mono2Micro analizuje powstałe pliki logu.
5. Uzyskaj dostęp do interfejsu użytkownika z przeglądarki, przeglądaj i dostosowuj zalecane partycje.
6. Mono2Micro tworzy nowe partycje mikrousług i generuje kod umożliwiający komunikację między partycjami.
7. Wykonaj te same przypadki testów biznesowych na nowych partycjach mikrousług w Liberty.

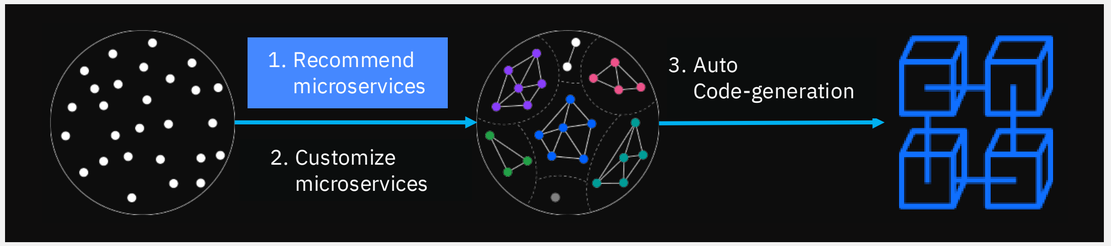
Mono2Micro – Refaktoryzacja aplikacji poprzez analizę statyczną i środowiska uruchomieniowego



Przepływ użycia Mono2Micro



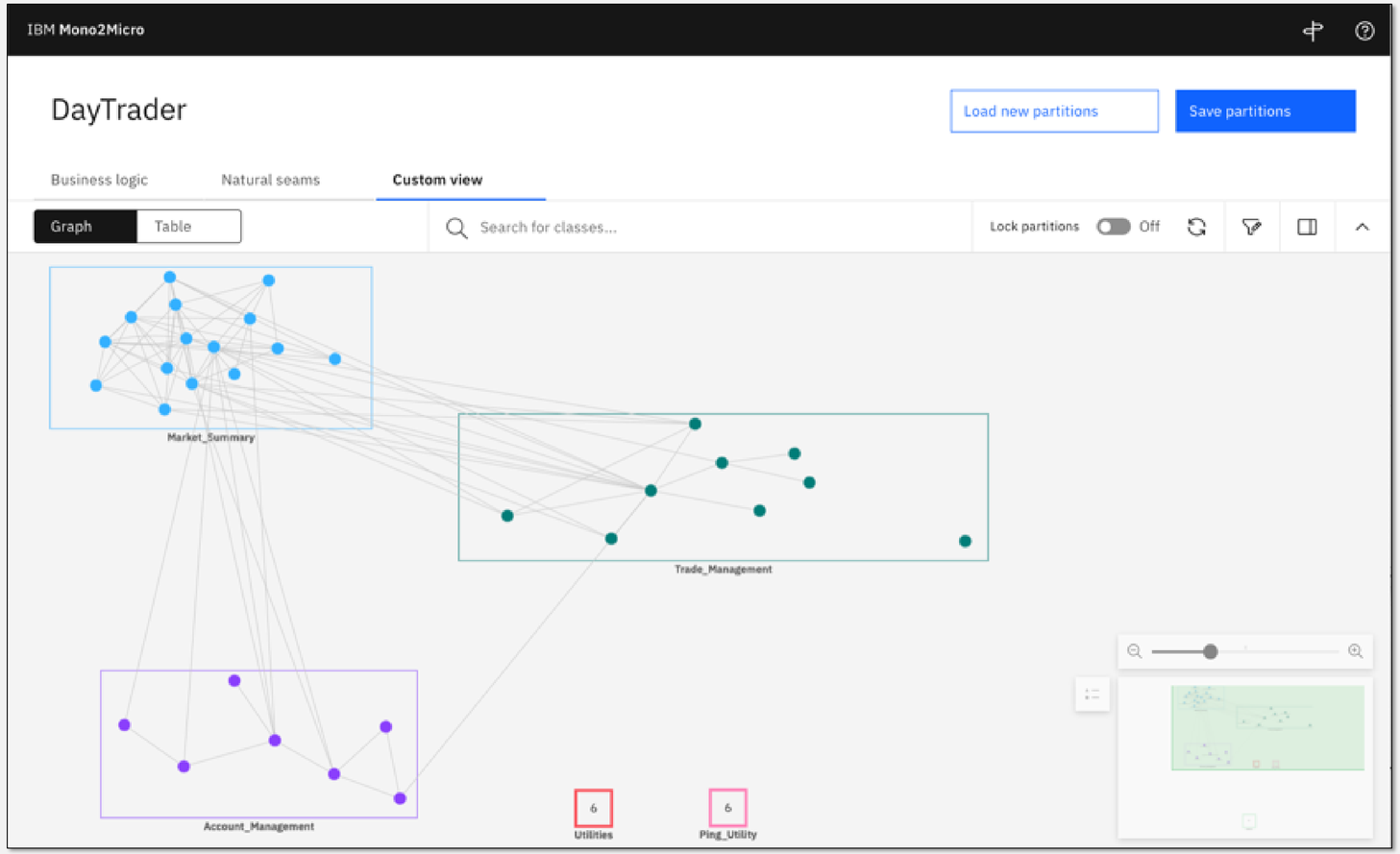
IBM Mono2Micro – Rekomendacje mikrousług



Wejście:
Runtime Traces,
Use Cases, Code
Metadata

Machine Learning &
Clustering

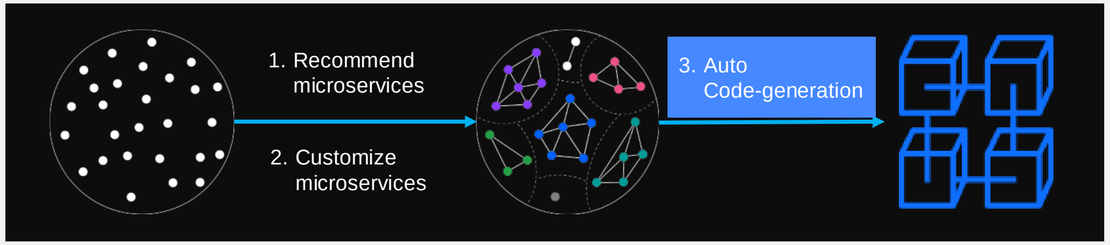
Wyjście:
Business Logic Seams



Nodes: Partitions / Groups of Classes - **Edges:** Call Volumes

`mono2micro-aipl – oparty o AI generator rekomendacji dotyczących partycji, które ostatecznie mogą stać się mikrousługami.`

Automatyczne generowanie kodu



Wejście

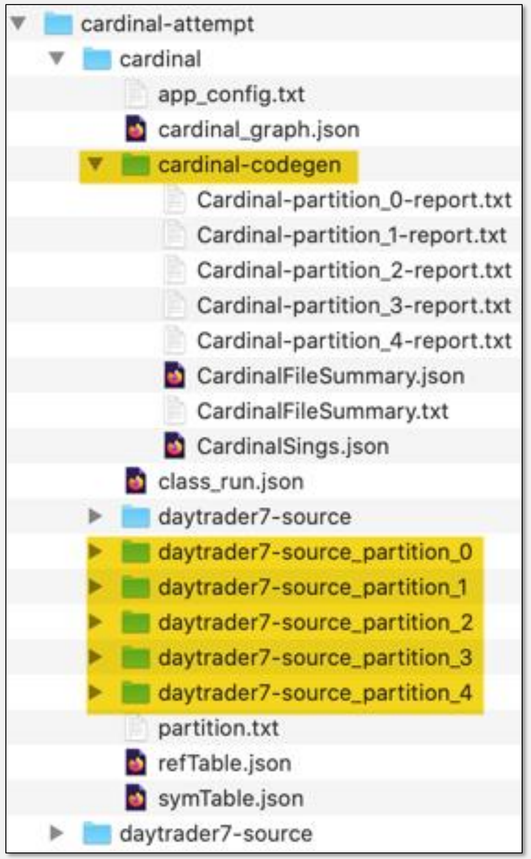
Recommended Microservice (Business Logic)

+

Original Source Code

mono2micro-cardinal – oparty o AI generator kodu, który posiada głęboką wiedzę na temat semantyki języka programowania Java.

Wyjście



Codebase for each microservice

Przykładowy automatycznie wygenerowany kod dla mikrousługi

```
. . .
public Collection<QuoteDataBean> getTopGainers() {


        Form form = new Form();
        form.param("referenceId",
String.valueOf(this.referenceId));

         JsonObject response_json =
client.target(DAYTRADER_BIZLOGIC_REST_URL)
            .path("getTopGainers")
            .request(MediaType.TEXT_PLAIN)
            .post(Entity.entity(form,
MediaType.APPLICATION_FORM_URLENCODED_TYPE), String.class);
        String response =
(String)response_json.get("return_value");


        Collection<String> response_decoded =
(Collection<String>)SerializationUtil.decode(response);
        String ret_type = (String)response.get("return_type");
. . .
```

Transformacja monolitów w mikrousługi

Analiza Manualna

- Trudny, kosztowny i czasochłonny proces
- Wymaga wysokiego poziomu umiejętności
- Wymaga dogłębnej znajomości istniejącego kodu
  - nie zawsze dostępna
  - twórcy aplikacji mogli odejść
- Znaczna szansa na popełnienie błędu
  - może być konieczna rearchitektura i ponowne wdrożenie
  - sprawdzany na przypadkach rzeczywistych

Mono2Micro - Analiza oparta na AI

- Automatyczna analiza statyczna i środowiska uruchomieniowego
- Brak konieczności posiadania specjalnych umiejętności
- Dogłębna znajomość kodu nie jest wymagana
- ML jest kluczem do uzyskania właściwych rekomendacji dotyczących mikrousług
  - ML emuluje wiedzę i inteligencję MŚP
  - Refaktoryzacja, ale nie rearchitektura i ponowna implementacja

- Mono2Micro pomaga architektom w prawidłowym partycjonowaniu klas, gdy tylko jest to możliwe, ze świadomością funkcji biznesowych.

Rozważania dotyczące przejścia na mikrousługi z Mono2Micro

- Transformacja aplikacji to podróż i wieloetapowy proces
- Refaktoryzacja w celu modernizacji architektury jest czynnością związaną z projektowaniem i kodowaniem.
- Podczas refaktoryzacji monolitu
    - Wykorzystaj Mono2Micro, aby rozpocząć kodowanie, które wygeneruje początkową partycję aplikacji w celu zwiększenia szybkości rozwoju.

Korzyści z analizy „monolitów” są następujące:

- Analiza zależności (hermetyzacja, specyficzne wywołania, dziedziczenie)
- Partycje skoncentrowane na logice biznesowej oparte na klastrowaniu przestrzenno-czasowym
- Partycje z czystym kontekstem biznesowym (podobne przypadki użycia wskazują na spójność)
- Wolumen połączeń (wolumen połączeń między partycjami wskazuje na sprzężenie)
- Strukturalnie modularne partycje (spójność i sprzężenie)
- Wykrywanie potencjalnie martwego kodu
- Wykrywanie potencjalnego kodu użytkowego

DEMO - Mono2Micro