*Architektura IT w NCBR*

**Zamawiający:**

**Dyrektor NCBR**

**Pryncypia Architektoniczne**

Wersja 0.5

**NCBR, *2021-07-30***

**Dokument zastrzeżony, do użytku wewnętrznego**

**Historia modyfikacji**

|  |  |  |  |
| --- | --- | --- | --- |
| Wersja | Data | Autor zmiany | Opis zmiany |
| 0.5 | 23.08.2021 | Zdzisław Krysztofiak | Utworzenie dokumentu |
| 0,52 | 23.08.2021 | Zdzisław Krysztofiak | Połączenie z treścią dokumentu opisów pryncypiów architektury korporacyjnej podmiotów publicznych. |
|  |  |  |  |
|  |  |  |  |
| 1.0 |  | Dyr. DSI | Wersja przyjęta |

F

Spis treści

[**1** **Cele, korzyści i zakres formułowania pryncypiów architektury korporacyjnej podmiotów publicznych** 5](#_Toc80886046)

[1.3 Klasyfikacja pryncypiów architektury korporacyjnej 5](#_Toc80886047)

[1.5 Cechy dobrych pryncypiów 6](#_Toc80886048)

[1.6 Strategia informatyzacji a pryncypia 6](#_Toc80886049)

[1.7 Lista pryncypiów architektonicznych poziomu całej organizacji NCBR 7](#_Toc80886050)

[1.8 Szablon opisu pryncypiów 7](#_Toc80886051)

[**2** **Pryncypia architektury biznesowej** 8](#_Toc80886052)

[2.1 Rys ogólny 8](#_Toc80886053)

[2.2 Abstrakcja w architekturze biznesowej 8](#_Toc80886054)

[2.3 Poziomy abstrakcji 9](#_Toc80886055)

[**3** **Pryncypia architektury danych** 10](#_Toc80886056)

[3.1 Cele 10](#_Toc80886057)

[3.2 Zakres pojęciowy architektury danych 10](#_Toc80886058)

[3.3 Zakres pojęciowy architektury danych – co nie wchodzi w zakres 10](#_Toc80886059)

[3.4 Zakres architektury danych wg TOGAF’a 11](#_Toc80886060)

[**7.** **Pryncypia architektury aplikacji** 11](#_Toc80886061)

[7.1 Cele 11](#_Toc80886062)

[7.2 Dokumentacja 11](#_Toc80886063)

[7.2.1 APL1 Log decyzji architektonicznych ADR (Architecture Decision Record) 11](#_Toc80886064)

[7.2.2 APL2 Wizualizacja architektury 12](#_Toc80886065)

[7.3 Modelowanie aplikacji 12](#_Toc80886066)

[7.3.1 APL3 Dekompozycja Domeny 12](#_Toc80886067)

[7.3.2 APL4 Event Storming 12](#_Toc80886068)

[7.3.3 APL5 Integracja przy użyciu Szyny Danych ESB 13](#_Toc80886069)

[7.3.4 APL6 Wymagania niefunkcjonalne 13](#_Toc80886070)

[**8.** **Pryncypia obejmująca architekturę infrastruktury** 14](#_Toc80886071)

[8.1 Definicja infrastruktury 14](#_Toc80886072)

[8.2 Chmura lub on-premises 14](#_Toc80886073)

[8.2.1 TECH1 Rozwiązania *serverless* 14](#_Toc80886074)

[8.3 Infrastruktura jako kod (IaC) 14](#_Toc80886075)

[8.3.1 TECH2 Przechowywanie kodów/skryptów infrastruktury IaC 14](#_Toc80886076)

[8.4 Mechanizmy kolejkowe 15](#_Toc80886077)

[8.5 Roadmapy technologiczne 15](#_Toc80886078)

[**9.** **Pryncypia generalne** 15](#_Toc80886079)

[9.1.1 GEN1 Bezpieczeństwo i prywatność 15](#_Toc80886080)

[9.1.2 GEN2 Neutralność technologiczna 15](#_Toc80886081)

[9.2 Usługi publiczne wystawiane przez NCBR 16](#_Toc80886082)

[9.2.1 GEN3 Dostępność usługi 16](#_Toc80886083)

[9.2.2 GEN4 Mierzalność usługi 16](#_Toc80886084)

[9.2.3 GEN5 Interoperacyjność usługi 16](#_Toc80886085)

[9.3 Strategia dotycząca systemów informatycznych 20](#_Toc80886086)

[**10.** **Kontrola Budowy i Rozwoju oprogramowania** 22](#_Toc80886087)

[10.1 Zapewnianie jakości i testy 22](#_Toc80886088)

[10.2 (ZK) Architektura i Integracja 22](#_Toc80886089)

[10.3 Zarządzanie konfiguracją 22](#_Toc80886090)

[**Mechanizmy kontroli aplikacji** 23](#_Toc80886091)

# **Cele, korzyści i zakres formułowania pryncypiów architektury korporacyjnej podmiotów publicznych**

* 1. **Architektura korporacyjna i cele strategiczne informatyzacji**

Cele, korzyści i zakres formułowania pryncypiów architektury korporacyjnej podmiotów publicznych są pochodnymi roli i funkcji architektury korporacyjnej państwa.

Architektura korporacyjna jest „zarówno sposobem zarządzania, jak i metodą dokumentacji, które razem wprowadzają możliwy do wykonania, skoordynowany widok celów strategicznych, procesów biznesowych, przepływu informacji i wykorzystania zasobów (w tym w szczególności zasobów informatycznych).

Podstawową funkcją architektury korporacyjnej jest wspieranie realizacji strategii organizacji - w szczególności poprzez przyczynianie się do osiągania celów strategicznych ustanowionych przez organizację. W przypadku administracji publicznej cele strategiczne, których realizacja powinna być wspierana przez architekturę korporacyjną, zawarte są w dokumentach strategicznych informatyzacji państwa.

* 1. **Definicja pryncypiów architektury korporacyjnej podmiotów publicznych**

Kluczowymi elementami architektury korporacyjnej są pryncypia. Przez pryncypia architektury korporacyjnej rozumieć należy „zbiór podstawowych, trwałych zasad bazujących na strategii organizacji i stanowiących reprezentację całościowych potrzeb organizacji w zakresie tworzenia jej rozwiązań IT”9.

Pryncypia architektury korporacyjnej odnoszą się do następujących domen architektonicznych: architektury biznesowej, architektury danych, architektury aplikacji, architektury infrastruktury technicznej.

## Klasyfikacja pryncypiów architektury korporacyjnej

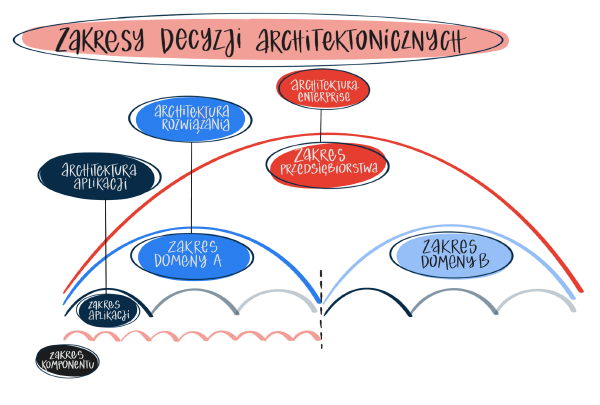
|  |  |
| --- | --- |
| Pryncypia o charakterze generalnym | |
| Dotyczące biznesu | Dotyczące danych |
| Dotyczące aplikacji | Dotyczące infrastruktury |
| Pryncypia architektury korporacyjnej | |

* 1. **Zakres decyzji korporacyjnej, systemowej, aplikacyjnej**

**Architekt Korporacyjny** odpowiada za to aby strategia biznesowa firmy wykorzystuje odpowiednią architekturę systemów technologicznych do osiągnięcia swoich celów. Muszą nadążać za najnowszymi trendami w technologii i decydować, czy będą odpowiednie dla firmy.

**Architekt Rozwiązań** odpowiada za zakres domeny i odgrywaj kluczową rolę w każdej organizacji, która chce dostosować swoje cele biznesowe i potrzeby do usług, produktów, oprogramowania i infrastruktury IT. Wypełnia lukę między biznesem a IT. Zarządza roadmapą technologiczną.

**Architekt Aplikacyjny** jest liderem technicznym, odpowiada za budowę strategii dla aplikacji, dobiera optymalne technologicznie: języki programowania+IDE, framework’i, bazy danych, biblioteki. Nadzoruje pracę deweloperów w oparciu o „dobre praktyki”, dba o wykorzystywanie dostępnych „building blocks” – gotowych komponentów, tak aby skracać czas wykonania, redukować potencjalne błędy i przyspieszać realizację, zgodnie z zasadą „*now code/low code*”



## Cechy dobrych pryncypiów

**„Każde pryncypium jest zasadą, nie każda zasada jest pryncypium.”**

Pryncypia powinny być:

* **Jednoznaczne** – dla każdego pryncypium powinna istnieć tylko jedna jego interpretacja;
* **Kompletne** – należy dążyć do stworzenia takiej listy pryncypiów, które z jednej strony wyczerpywałaby wszystkie kluczowe zagadnienia, a z drugiej strony nie byłaby nadmiarowa;
* **Zrozumiałe** – pryncypia są swoistym kontraktem, który obowiązuje w całej organizacji; aby ten kontrakt był skuteczny, interesariusz muszą rozumieć pryncypia;
* **Stabilne** – pryncypia powinny być w miarę możliwości neutralne (agnostyczne) technologicznie;
* **Spójne** – pryncypia nie mogą być ze sobą w konflikcie;

## Strategia informatyzacji a pryncypia

Kolejność definiowania strategii w poniższej kolejności powinny dostarczać „wsadu wejściowego” czyi wytycznych skutkujących opracowaniem i aktualizacja Pryncypiów architektury korporacyjnej IT

## Lista pryncypiów architektonicznych poziomu całej organizacji NCBR

* NCBR zapewnia bezpieczeństwo, poufność i prywatność przetwarzanych informacji i gromadzonych przez siebie danych
* Systemy oprogramowania używane w NCBR bazują na powszechnie uznanych i stosowanych standardach technicznych lub/i dobrych praktykach
* Systemy oprogramowania NCBR zapewniają interoperacyjność na płaszczyźnie technicznej
* Systemy oprogramowania stosowane w NCBR zapewniają możliwość ciągłego świadczenia usług publicznych
* Minimalizowanie stopnia zróżnicowanie technologii informatycznych będących podstawą systemów oprogramowania NCBR

Systemy oprogramowania NCBR są ergonomiczne i przyjazne w obsłudze

Pryncypiów architektonicznych obejmujących architekturę biznesową, danych, aplikacji i infrastruktury

## Szablon opisu pryncypiów

W celu jednolitego ujęcia pryncypiów zdefiniowano szablon ich opisu. W ramach tworzenia opisu pryncypium wymagane jest wypełnienie wszystkich pól.

|  |  |
| --- | --- |
| Nazwa pryncypium | Niepowtarzalna nazwa pryncypium (nie powinna przekraczać 7-8 wyrazów). |
| Identyfikator pryncypium | Jednoznaczny identyfikator pryncypium – składa się on z prefiksu oraz numeru.  Dopuszczalne prefiksy:   * BIZ Pryncypia architektury biznesowej * DANE Pryncypia architektury danych * APL Pryncypia architektury aplikacji * TECH Pryncypia architektury technicznej * GEN Pryncypia generalne   Prefiks identyfikatora pryncypium musi być zgodny z kategorią pryncypium. |
| Wersja pryncypium | Numer wersji pryncypium.  Numer dla wersji formalnie niezatwierdzonej pryncypium pochodzi z przedziału 0.1-0.99.  Pierwsza wersja pryncypium formalnie zatwierdzona ma numer 1.0.  Wersje pryncypium powyżej 1.0 oznaczają, że pryncypium zostało zmienione i zmiana ta została przeprowadzona przez proces zarządzania zmianą. |
| Syntetyczny opis | Opis myśli przewodniej pryncypium. |
| Uzasadnienie pryncypium (opcjonalne) | Korzyści dla Usługobiorców, Właściciela usługi i Usługodawców ze stosowania pryncypium w ramach tworzenia i świadczenia |

# **Pryncypia architektury biznesowej**

## Rys ogólny

Pryncypia tej kategorii w największym stopniu związane są z domeną architektury biznesowej (tj. dotyczą one celów, procesów i usług biznesowych, a także aspektów organizacyjnych i ludzkich).

Jedną z kluczowych cech architektury jest patrzenie na „duży obraz”, ale głównym wyzwaniem jest to, że nie możemy przedstawić pełnego obrazu na jednej wielkiej kartce papieru – musi zmieścić się na jednym arkuszu lub modelu . Aby to zrobić, musimy wymyślić nowe koncepcje, które podsumowują ogólny obraz w niewielkiej liczbie elementów i relacji. Możemy to zrobić za pomocą różnych technik, takich jak dziel i zwyciężaj, kategoryzacja, uogólnianie i tak dalej

## Abstrakcja w architekturze biznesowej

Abstrakcja jest procesem lub wynikiem uogólniania, usuwania właściwości lub dystansowania idei od obiektów. W IT jest to proces ukrywania szczegółów implementacji w programach i danych. Należy zauważyć, że same modele są abstrakcją. Modele zawierają zestaw pojęć i relacji w kontekście. Dobrze uformowane modele mają spójny i konkretny zestaw pojęć, z których każde samo w sobie jest abstrakcją. Razem zapewniają reprezentację pożądanego ( strategii lub przyszłego), rzeczywistego ( takiego , jaki jest) lub zamierzonego (projektowego) stanu rzeczy rzeczywistych w kontekście modelu.

## Poziomy abstrakcji

* Koncepcyjne — modele koncepcyjne skupiają się na kluczowych koncepcjach i relacjach całego rozwiązania, a nie na sposobie działania systemu . Jako takie, są to generalnie modele statyczne, w których łączniki, jeśli są obecne, pokazują relacje, a nie przepływy.
* Logiczne — modele logiczne opisują sposób działania rozwiązania pod względem funkcji i logicznych relacji między zasobami, czynnościami, wynikami i wynikami. Mogą pokazywać widok statyczny lub widok dynamiczny.
* Fizyczne — modele fizyczne odnoszą się do określonych produktów, protokołów, reprezentacji danych, możliwości sieciowych, specyfikacji serwera, wymagań sprzętowych i innych informacji związanych z wdrażaniem proponowanego systemu.

Modele koncepcyjne są bardziej abstrakcyjne niż modele logiczne, które są bardziej abstrakcyjne niż modele fizyczne. Proces przekształcania jednego modelu w drugi opisujemy jako udoskonalenie, gdy obniżamy poziom abstrakcji. Zauważ, że transformacja modeli między poziomami to coś więcej niż tylko dodawanie szczegółów. Podczas transformacji pojęcia abstrakcyjne są przekształcane w pojęcia bardziej konkretne. Na przykład pojęcie „klienta” może zostać przekształcone w logiczną jednostkę informacji o kliencie, a następnie przekształcone w zestaw tabel i złączeń na fizycznym poziomie danych. Możemy również przekształcać modele w innym kierunku, przechodząc od fizycznych (bardziej dopracowanych) do logicznych, do konceptualnych (mniej dopracowanych). Proces ten nazywamy abstrakcją.

# **Pryncypia architektury danych**

## Cele

Pryncypia tej kategorii w największym stopniu związane są z domeną architektury danych (tj. dotyczą one zasobów informacyjnych organizacji, danych gromadzonych w tych zasobach i mechanizmów zarządzania nimi).

Architektura danych – jest to formalny opis pryncypiów, modeli, polityk, standardów, które są (powinny być) stosowane podczas zarządzania:

* Gromadzeniem
* Przechowywaniem
* Integracją
* Używaniem

danych w organizacji

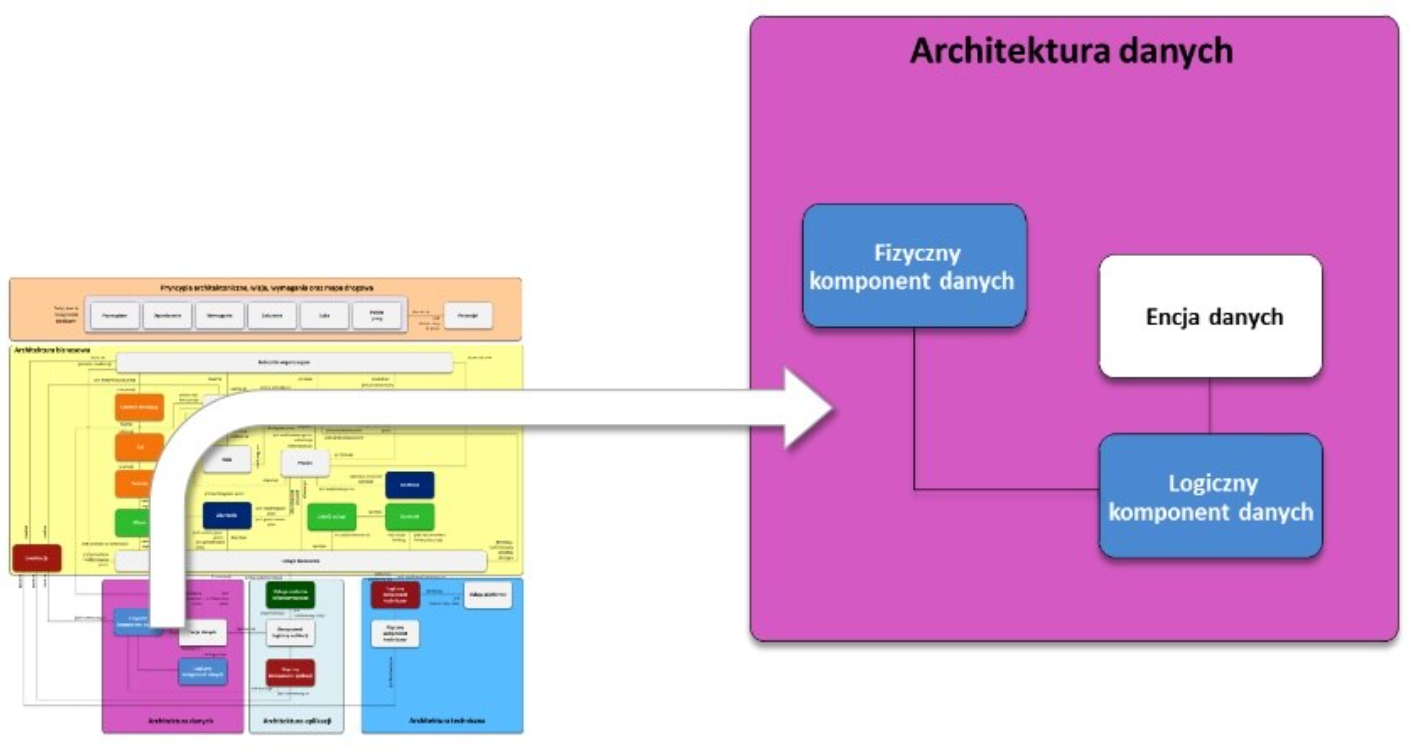
## Zakres pojęciowy architektury danych

1. Lista bytów istotna dla biznesu
2. Model semantyczny
3. Logiczny model danych
4. Fizyczny model danych
5. Definicja danych
6. Dane

## Zakres pojęciowy architektury danych – co nie wchodzi w zakres

* Serwery bazo-danowe
* Elementy architektury technicznej

## Zakres architektury danych wg TOGAF’a



# **Pryncypia architektury aplikacji**

## Cele

Pryncypia tej kategorii w największym stopniu związane są z domeną architektury aplikacji (tj. dotyczą one aplikacji – ich funkcji, interfejsów, usług aplikacyjnych).

## Dokumentacja

### APL1 Log decyzji architektonicznych ADR (Architecture Decision Record)

ADR jest mini-dokumentem, który przechwytuje istotne decyzje, opisuje kontekst, opcje brane pod uwagę i konsekwencje przyjęcia decyzji.

**Zalecenie 1**

Należy dokumentować ważne decyzje w formacie jak poniżej, wypełniając sekcje w szablonie:

**Tytuł**

**Status**

Jaki jest status, np. zaproponowany, zaakceptowany, odrzucony, przestarzały, zastąpiony itp.?

**Kontekst**

Jaki problem, który widzimy, motywuje tę decyzję lub zmianę?

**Decyzja**

Jaką zmianę proponujemy i/lub wprowadzamy?

**Konsekwencje**

Co staje się łatwiejsze lub trudniejsze z powodu tej zmiany?

### APL2 Wizualizacja architektury

Wybrany standard (notacja, język) wizualizacji statycznej komponentów architektury powinien być zrozumiały dla możliwie dużego grona pracowników NCBR, w szczególności powinien ułatwiać komunikację z biznesem.

Unified Modeling Language (UML) obecnie nie jest powszechnie rozumianą notacją, powoli zapominany, skutkuje to trudnością użycia

Archimate – wszystkie warstwy na jednym obrazku – nieczytelny

**Zalecenie 1**

Należy używać przejrzystego, warstwowego modelu C4 - od ogółu do szczegółu System (Context)->Container->Component->Code.

**Zalecenie 2**

Stosując podejście *Schematy jako Kod*, do modelowania w standardzie C4, należy używać narzędzia [Structurizr](https://structurizr.com/)

## Modelowanie aplikacji

### APL3 Dekompozycja Domeny

Modelowanie aplikacji należy poprzedzić analizą umożliwiającą dekompozycję domeny biznesowej tak aby uzyskać mniejsze modułu z perspektywą ich wydzielenia o osobne systemy/mikrousługi. W dłuższej perspektywie pozwala zmniejszyć błędy szacowania.

**Zalecenie 1**

Należy wyekstrahować domeny wg kategorii:

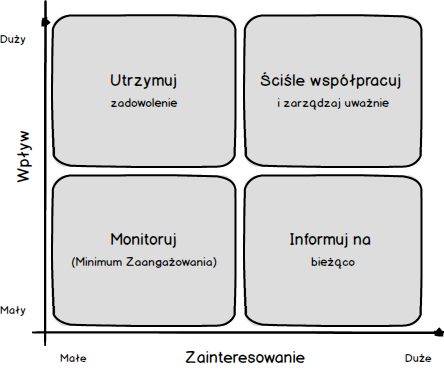
* Domena główna (*core domain*) – jest wyróżnikiem biznesowym związanym ze strategią, odpowiada na pytania:
  + dlaczego powstaje ten system
  + dlaczego go nie kupiliśmy
* Domena pomocnicza – zapewnia funkcje wspierające *core domain*, nie istniej gotowy produkt je dostarczający
* Domeny generyczne – dostarczane przez nie funkcjonalności nie są unikalne dla NCBR, istnieją dla tych funkcji gotowe rozwiązania

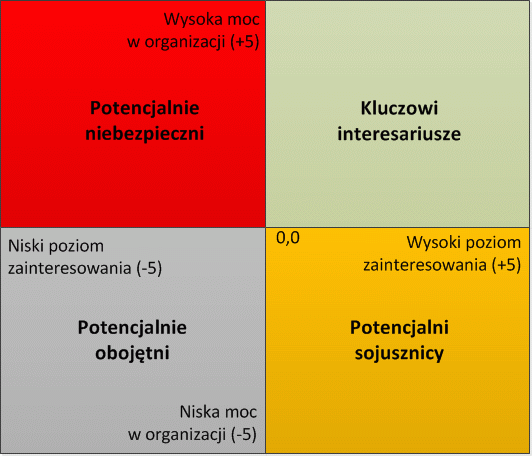
### APL4 Event Storming

Modelowanie Domen należy przeprowadzić posługując się technikami warsztatowymi Event Storming, które pozwalają (współpracując z Ekspertami Dziedzinowymi) zakreślenie granic domen/kontekstów modelu (*Bounded Context*)

**Zalecenie 1**

Przygotowujemy listę uczestników w oparciu o Mapowanie Interesariuszy (Stakholders Mapping). Mapowanie służy do wizualnego zaprezentowania ich wzajemnych powiązań i zależności. Na poniższych rysunkach wybieramy Interesariuszy, którzy są bardzo zainteresowani/zaangażowani oraz wpływowi (prawy górny róg – Kluczowi)





**Zalecenie 2**

Rolę tzw. Facylitatora (osoba, której zadaniem jest czynny udział w udrażnianiu komunikacji oraz zapobiegania ewentualnym konfliktom) powierzamy osobie zawodowo się tym zajmującej.

### APL5 Integracja przy użyciu Szyny Danych ESB

TBD

### APL6 Wymagania niefunkcjonalne

Wymagania niefunkcjonalne – opisują ograniczenia, przy zachowaniu których system powinien realizować swe funkcje.   
NCBR posiada listę wymagań funkcjonalnych, które powinny być uwzględnione przy zamawianiu aplikacji

Wymagania, jako szablon powinny obejmować cechy, zaś miary (liczbowe) powinny być specyfikowane dla konkretnej aplikacji.  
Przykład:

Cecha: Miary:

– Wydajność: Liczba trans. obsłużonych/s.

– Dostępność: np. 99.99% w dni robocze, w godz. 7:00-17:00

– Liczba jednoczesnych np. 300

użytkowników

– Rozmiar: Wymagana pamięć RAM; Wymagana pamięć dyskowa

– Łatwość użytkowania: Czas dla przeszkolenia Liczba stron dokumentacji

– Odporność Czas restartu systemu po awarii

– Wspierane przeglądarki Chrome ver.>84 lub nowsza, Firefox ver. 92.1 lub nowsza

# **Pryncypia obejmująca architekturę infrastruktury**

## Definicja infrastruktury

Infrastruktura to wszystko, czego potrzebujemy do uruchomienia aplikacji, poza samą aplikacją a mianowicie:

* hardware
* hypervisor / cloud
* maszyny wirtualne
* konfiguracja systemu operacyjnego
* skrypty budujące i pipeline’y wdrożeniowe
* parametryzacja aplikacji i skrypty wdrożeniowe (deploymentowe).

## Chmura lub on-premises

### TECH1 Rozwiązania *serverless*

Rozwiązania *serverless* sprawdzają się w przypadku pisania prototypów, *MVP, Proof of Concept* – przy małych obciążeniach nie generują praktycznie żadnych kosztów, a do tego są w każdym momencie przygotowane do „przyjęcia” obciążeń produkcyjnych.

Usługi, w których obserwujemy częste, nieprzewidywalne skoki obciążeń będą także dobrymi kandydatami do implementacji przy użyciu rozwiązań *serverless*.

Tak samo jak akcje wykonywane w regularnych odstępach czasowych (np. *cron joby*).

W przypadku aplikacji działających z równomiernym, dużym obciążeniem zastosowanie usług serverless nie będzie optymalne kosztowo. Trzeba jednak zawsze wziąć pod uwagę cenę nie tylko samej infrastruktury, ale także pracy osób, które nią zarządzają – w każdej organizacji/zespole kalkulacja ta będzie inna.

**Zalecenie 1**

Przygotowujemy listę serwisów, których prototypowanie może być prowadzone „w chmurze”

## Infrastruktura jako kod (IaC)

**Infrastruktura jako kod** ( **IaC** ) to proces zarządzania komputerowymi [centrami danych](https://en.wikipedia.org/wiki/Data_center) i udostępniania ich za pomocą plików definicji do odczytu maszynowego, a nie fizycznej konfiguracji sprzętu lub interaktywnych narzędzi konfiguracyjnych

Zalety podejścia IacC:

* większa prostota i jakość utrzymania,
* hermetyzacja / enkapsulacja,
* uproszczenie procedur tworzenia kopii zapasowych,
* większa przenoszalność,
* większy stopień czytelności / przejrzystości.

### TECH2 Przechowywanie kodów/skryptów infrastruktury IaC

Ze względu na potrzebę przejrzystości zmian i audytowalność należy składować konfigurację infrastruktury analogicznie jak kod aplikacji.

**Zalecenie 1**

Przygotowujemy kod używając mechanizmów kontroli wersji GIT+Github

### TECH3 Roadmapy Technologiczne

Mapa drogowa technologii to elastyczna technika planowania wspierająca planowanie strategiczne i długoterminowe, poprzez dopasowywanie celów krótko- i długoterminowych do konkretnych rozwiązań technologicznych.Jest to plan, który ma zastosowanie do nowego produktu lub procesu i może obejmować wykorzystanie prognozowania technologii lub poszukiwania technologii w celu zidentyfikowania odpowiednich nowych technologii

**Zalecenie 1**

Należy publikować Roadmapy Technologiczne na Intranecie oraz traktować je jako podstawę i wsparcia przy tworzeniu listy wymagań niefunkcjonalnych

**Zalecenie 2**

Należy wyznaczyć Właściciela Roadmapy dla poszczególnych obszarów

### TECH4 Mechanizmy kolejkowe MQ

Kolejka jako element infrastrukturalny z założenia ma dużą większą dostępność niż sama aplikacja, która z niej korzysta.

# **Pryncypia generalne**

### GEN1 Bezpieczeństwo i prywatność

Dobór technicznych i organizacyjnych środków bezpieczeństwa musi wynikać z obowiązujących przepisów prawa i wymagań biznesowych, w szczególności oczekiwanego poziomu niezawodności, integralności i poufności danych i systemów.

**Zalecenie 1**

Przy formułowaniu założeń systemu teleinformatycznego należy zidentyfikować wszystkie przepisy prawa i standardy, które są wymagane dla systemu i muszą być spełnione na etapie projektowania, budowy i eksploatacji.

**Zalecenie 2**

Projektując rozwiązanie współdzielone należy brać pod uwagę potrzeby potencjalnych interesariuszy w zakresie bezpieczeństwa.

**Zalecenie 3**    
Projektując rozwiązania teleinformatyczne należy przywidzieć audytowalność zasad bezpieczeństwa i ich stosowania oraz automatyczne raportowanie incydentów bezpieczeństwa do właściwych instytucji.

### GEN2 Neutralność technologiczna

Systemy teleinformatyczne należy projektować w sposób minimalizujący ryzyko zachowań monopolistycznych ze strony dostawców (vendor lock-in) oraz zapewniający zachowanie zasad pełnej konkurencyjności.

**Zalecenie 1**  
Należy przeprowadzić analizę ryzyka związanego z dostawcami (vendor lock-in), technologiami i ewentualnymi zachowaniami monopolistycznymi w celu zapewnienia zarządzalnej neutralności technologicznej. Wynik analizy ryzyka i działania mitygacyjne muszą zostać zaakceptowane przez właściciela biznesowego.

**Zalecenie 2**    
Stosowanie otwartych standardów w obszarze przechowywania i przetwarzania danych: Zastosowane w projekcie modele i struktury danych i metadanych muszą uwzględniać obowiązujące standardy, a przy ich braku - uznane dobre praktyki.   
   
**Zalecenie 3**    
Nie należy wprowadzać rozwiązań opartych na specyficznych, niszowych  technologiach o ile nie wynika to z rzeczywistych potrzeb biznesowych.    
   
**Zalecenie 4**    
Należy zapewnić możliwość przenoszenia danych pomiędzy systemami i aplikacjami urzędów i instytucji uprawnionych zarówno na poziomie krajowym, jak i europejskim, poprzez stworzenie otwartych interfejsów lub otwartych formatów eksportu i importu danych.

## Usługi publiczne wystawiane przez NCBR

### GEN3 Dostępność usługi

Usługa jest dostępna co najmniej na poziomie oczekiwanym przez usługobiorców, przy jednoczesnym uwzględnieniu możliwości realizacyjnych usługodawcy, a także na poziomie dostępności (availability) nie mniejszym niż określony w przepisach prawa

**Zalecenie 1**

Należy zapewnić poziom dostępności (availability) usługi oczekiwany przez Usługobiorców określany w szczególności na podstawie empirycznych badań potrzeb usługobiorców.

**Zalecenie 2**

Należy określić i udostępnić publiczny poziom dostępności

### GEN4 Mierzalność usługi

Definiowane i stosowane są mierniki będące podstawą weryfikacji wartości publicznej dostarczanej przez usługę

**Zalecenie 1**

Należy ustalić mierniki i wartości mierników w celu mierzalnego określania dostarczania wartości publicznej przez usługę.

**Zalecenie 2**

W fazie świadczenie usługi należy prowadzić cykliczną weryfikację wartości mierników.

### GEN5 Interoperacyjność usługi

Usługa ma zdolność do współdziałania z innymi usługami (dostarczanymi przez tego samego lub różnych usługodawców), w celu efektywnego wsparcia realizacji procesów biznesowych.

**Zalecenie 1**

System teleinformatyczny wspierający realizację usługi jest oparty na modelu usługowym, o którym mowa w obowiązujących przepisach.

**Zalecenie 2**

Należy udostępnić interfejsy usług sieciowych (web services) dla kluczowych funkcji systemu teleinformatycznego wspierającego świadczenie usługi

**Zalecenie 3**

Właściciel usługi po podjęciu decyzji o wycofaniu usługi powiadamia o tym fakcie - z odpowiednim wyprzedzeniem - usługobiorców oraz innych właścicieli usług, które na podstawie modelu procesów biznesowych wymieniają dane z usługą

**Problem z terminowym zatwierdzaniem wymogów biznesowych co negatywnie wpływa na planowanie informatyczne.**

W przypadku projektów powstających na zlecenie IZ, wymogi biznesowe w dużej części są narzucane z zewnątrz. W takich sytuacjach NCBR jest Wykonawcą usługi. Przekłada się to na często niespójne wymagania biznesowe względem stosowanych narzędzi informatycznych Centrum, wobec czego wymogi informatyczne polegają na próbie dostosowania obecnie funkcjonujących narzędzi do wymogów biznesowych. Dodatkowo, wymagania te są przekazywane zbyt  późno, pozostawiając niewiele czasu na ich szczegółową analizę, weryfikację jakości i terminowe wdrożenie.

W przypadku projektów własnych, w których klientem oraz wykonawcą jest NCBR, nie wprowadzono zasady **planowania i** **kreowania wymagań wspólnie z biznesem (np. w przypadku projektu LSI 2.0 wpływ biznes został znacząco ograniczony)**.

Sformułowano błędne założenie, iż Wykonawca (IT) lepiej rozumie potrzeby Zamawiającego. Rozpoczęcie projektu następuje po ogólnym zarysie przedstawiającym potrzeby klienta wewnętrznego. W wyniku takiego podejścia, w momencie jego pierwszego zetknięcie się  z rozwiązaniem informatycznym, powstaje wiele uwag i często konieczność przerabiania projektu, niwecząc dotychczas wykonaną pracę.

**Rozwiązanie:** Jako środek zaradczy, należy próbować stosować silniejszą negocjację z IZ, polegającą na przedstawianiu możliwości technicznych Centrum. Dodatkowo, rozmowy należy podejmować **wcześniej lub uzależnić termin wdrożenia projektu od czasu zakończenia rozmów**. Jednocześnie, sama realizacja projektu informatycznego może być realizowana w sposób zwinny - prace można rozpocząć od momentu pozyskania pierwszych materiałów z IZ; powstaje ryzyko powstawania elementów projektu, które będzie trzeba kompletnie przerobić albo wręcz z nich zrezygnować. Należy kategorycznie zakazać oczekiwanie na skompletowanie wszystkich wymagań.

1. **Brak przejrzystych zasad i informacji na temat mapowania potrzeb biznesowych/celów strategicznych na projekty IT, harmonogram realizacji, planowanie zasobów; brak procedury określania kryteriów akceptacji i podstawowych metryk do monitorowania postępów w projektach;**

Nie wprowadzono **zasad ani narzędzi** umożliwiających przejrzyste mapowanie potrzeb biznesowych i ich **jasne komunikowanie** dla zespołów projektowych IT i managerów średniego szczebla (kierowników zespołów). Pomimo zakupu narzędzi ułatwiających harmonogramowanie i planowanie zasobów (JIRA, Confluence), nie wprowadzono **standaryzowanych i udokumentowanych** zasad i procedur definiujących: zasady zgłaszania projektów w IT, metodykę harmonogramowania projektów i inicjatyw rozwojowych, umożliwiających zarządzanie współdzielonymi zasobami ludzkimi. Harmonogramy realizacji poszczególnych projektów i wdrożeń były narzucane jednoosobowo bez szczegółowych analiz dostępności zasobów w poszczególnych zespołach i wspólnych uzgodnień IT z biznesem, a także w ujęciu z harmonogramami innych projektów (te same zasoby osobowe). W rezultacie proces produkcji oprogramowania był oparty o ciągłe, nieskoordynowane przesunięcia terminów.

Uruchamiane projekty nie zawierały jasno zdefiniowanych kryteriów **określających miary sukcesu** dla wdrażanych produktów ani podstawowych metryk (**co najmnie**j w układzie plan/wykonanie) pozwalających na monitoring i analizę stanu faktycznego projektu i wdrożenie właściwych działań korygujących.

*Rozwiązanie: Wdrożyć w JIRA narzędzie do harmonogramowania projektów, cele biznesowe realizowane przez projekty IT oraz ich harmonogram, (BigPicture) wyznaczyć osobę do utrzymania BigPicture. Problemem dla PM’ów jest podwójna praca (JIRA, Hadron). Utworzyć w JIRA dedykowany projekt służący do zgłaszania projektów i inicjatyw rozwojowych dla systemów informatycznych, z uzasadnieniem, wskazaniem celów, korzyści i ścieżką decyzyjną.*

1. **brak monitoringu wpływu projektów informatycznych na cele biznesowe**

Brakuje podstawowych mierników efektywności pozwalajacych wskazać, czy projekt informatyczny realizuje cele biznesowe (patrz LSI 2.0); brak raportów pokazujących stopień zaawansowania realizacji celów biznesowych (np. struktura przypadków użycia, procesów biznesowych, wymagań - stopień ich realizacji, ilość błędów, ilość zmian i modyfikacji).

Brak metodyki ustalającej zasady tworzenia projektów, metod dekompozycji celów biznesowych na poszczególne zadania i grupy zadań, które będą łatwo raportowane dla kierownictwa, łatwo monitorowane i będą wskazywać prawdziwy stopień realizacji celów biznesowych

*Rozwiązanie: wdrożenie zasady, że wszystkie projekty IT (i z udziałem IT) są prowadzone w trybie operacyjnym w JIRA; ustalić metodykę zarządzania projektami IT (mogą być 2 podejścia), skonfigurować te podejścia w JIRA, skonfigurować standardowe dashboardy dla kierownictwa; projekty muszą być zestandaryzowane (typy zadań, statusy itp), inaczej nie będą w łatwy sposób raportowalne; raporty skonfigurować w układzie od ogółu (cel biznesowy) do szczegółu (pakiet zadań lub pojedyncze zadanie) - pomocne BIgPicture.*

1. **Nieskoordynowany proces inwestycji informatycznych (zakupy i plany zamówień)**

Brak analiz zapotrzebowania na systemy i aplikacje (na poziomie planu zamówień oraz braku strategii IT) powoduje że organizacja korzysta z niezintegrowanych narzędzi, z podobnymi funkcjami co w efekcie prowadzi do dodatkowej pracy i braku jednego źródła danych zarządczych. Brakuje szczegółowych uzasadnień dla zakupów, analiz wykonalności, ryzyk, analizy korzyści i analizy architektury i potrzeb integracji z systemami na poziomie NCBR. Brakuje szczegółowego harmonogramu realizacji zakupów i inwestycji w systemy informatyczne wraz ze wskazaniem osób odpowiedzialnych za realizację poszczególnych etapów (np OPZ, negocjacje, szacunki itp).

*Rozwiązanie: Należy wdrożyć proces zgłaszania zamówień, uzasadniania, decyzji z elementami uzgodnień architektonicznych (Główny Architekt) analizy wykonalności i ryzyk. Proces powinien być przygotowany w JIRA, zakomunikowany dyrektorom działów; wyeliminować przełym winformacji i uzasadnień mailem (nieraportowalne). Wdrożenie procesu w JIRA musi być umocowane na poziomie kierownictwa NCBR (lub wypracować ścieżkę wdrażania)*

## Strategia dotycząca systemów informatycznych

1. **Brak udokumentowanej strategii informatycznej lub brak komunikacji w zakresie:**

* zdefiniowanych miar sukcesu,
* metod pomiarów,
* metryk,
* stanów wyjściowych w projektach,
* harmonogramów realizacji projektów i zmapowania z potrzebami biznesowymi.

Dotychczas nie powstały żadne **udokumentowane oraz standaryzowane** miary sukcesu, metody pomiarów, metryki, stany wyjściowe. Harmonogram realizacji projektów jest dynamiczny i wielokrotnie zmieniany - począwszy od przedstawiania nierealnych, niczym nie uzasadnionych terminów realizacji, tylko po to, aby potem je wielokrotnie przekładać, powodując frustrację u klienta końcowego[[1]](#footnote-1), błędnie **tłumacząc to podejściem “zwinnym”**.

Od marca/kwietnia 2020 roku wprowadzono autorską metodykę tzw. “dailyscrum”, która doprowadziła do tego, że harmonogram powstaje bez uwzględniania dostępnych zasobów ludzkich, bez konsultacji z aktualnymi wykonawcami (analitycy, programiści, testerzy), bez uwzględnienia innych trwających aktualnie projektów. Pracownicy w ciągu jednego dnia byli odrywani od obecnych zadań do innych, w celu “gaszenia pożaru” w innych obszarach. Podejście oparte o “dailyscrum” prowadziło do nieefektywności wykorzystania zasobów (szczególnie programistów) oraz chaotycznego sposobu wykonywania zadań. Nie wdrożono w tym zakresie żadnych **udokumentowanych** procedur zarządzania krytycznymi zgłoszeniami i zadaniami, ich priorytetyzacji oraz poprzez brak komunikacji wypracowana metodyka prowadziła do nieporozumień wewnątrz zespołów.

*Rozwiązanie: Należy wprowadzić metodę harmonogramowania zadań z udziałem programistów,* ***przed podjęciem decyzji o uruchomieniu projektu****. Należy stworzyć dedykowane zepoły do poszczególnych obszarów, co umożliwi automatyzację zadań w JIRA (przypisania osób, zmiany statusów, SLA). Do opracowania pozostaje metoda nadawania priorytetów i wpływu na biznes zgłoszeniom ze środowiska produkcyjnego w celu kolejkowania zadań dla programistów; nalezy zdefiniować prawa i obowiązki analityków na poziomie analiz zgłoszeń i uwarunkować przejscia do realizajcji po spełnieniu okrełśonych warunków.*

1. **brak wdrożonych procedur/metodyki zarządza ryzykiem projektowym i produktowym w IT**

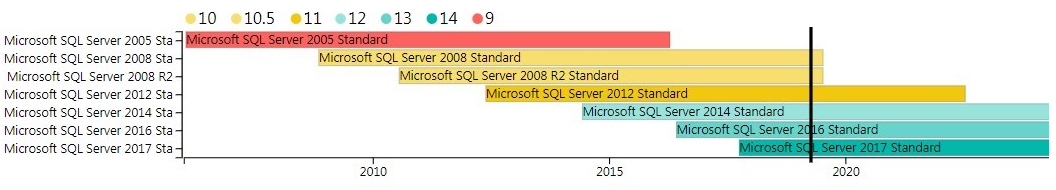
Nie wdrożono żadnego podejścia do zarządzania ryzykami; Ryzyka projektowe nie były określane na etapie planowania czy analizy projektu. Nie wdrożono i nie utrzymywano żadnego rejestru ryzyk projektowych. Pierwsze nieudokumentowane i niemonitorowane wzmianki związane z ryzykiem powstawały ad-hoc  dopiero na etapie realizacji projektu. W takich sytuacjach, działania zaradcze są często ad-hoc i zawężane do określonego obszaru. Nie ma jakichkolwiek informacji w jaki sposób reagować w przypadku materializacji ryzyka kto jest odpowiedzialny za zarządzanie ryzykiem i jakie są zaplanowane działania zaradcze.

Działania zaradcze w przypadku materializacji ryzyka były zazwyczaj prowadzone bez szczegółowej analizy, podejmowane jednoosobowo bez uzgodnień z zespołami wytwórczymi lub osobami (zazwyczaj PM) odpowiedzialnymi za prowadzenie projektów. Brak rejestrów ryzyk projektowych i produktowych **uniemożliwia obecnie wyciąganie wniosków przy planowaniu bieżących projektów i potencjalnych ryzyk.**

*Rozwiązanie: potrzebna krótka procedura identyfikowania i ewidencjonowania ryzyk; funkcjonalność przypisywania ryzyk do poszczególnych projektów, zadań, grup zadań posiada BigPicture; skrupulatna identyfikacja ryzyk jest niezbędna do śledzenia projektu i zarządzania pojedyncznymi zadaniami przy ograniczonych zasobach.*

1. **brak zarządzania Roadmap’ami produktowymi i technologii, ustalenia opiekunów - kto i jak często dokonuje przeglądu i aktualizacji?**

*Rozwiązanie:*

1. *Inwentaryzacja technologii użytkowanych w firmie, przypisanie im statusu (wersja dopuszczona, planowana, przestarzała/zabroniona). Wykorzystanie do tego celu gotowego/dedykowanego narzędzia.*
2. *Wyznaczenie osób odpowiedzialnych za przegląd i aktualizację poszczególnych obszarów/technologii takich jak: OSy, bazy danych, języki programowania itp,*
3. *Ustalenie okresowych przeglądów/aktualizacji*
4. *Publikacja Roadmapy na Intranecie, traktowanie ich jako podstawy i wsparcia do listy wymagań niefunkcjonalnych*   
   Przykład cyklu życia bazy MS SQL, który jest podstawą do wskazania wersji i statusu w firmie (przy wewnętrznych roadmapach nie musimy uwzględniać wszystkich wersji)  
   

# **Kontrola Budowy i Rozwoju oprogramowania**

## Zapewnianie jakości i testy

1. **Niedokreślony proces zapewniania jakości w tym nieokreślone szczegółowe role i obowiązki**
2. **Brak wdrożonej metodyki testów w zakresie planowania i przeprowadzania testów systemów informatycznych na wszystkich poziomach (od jednostkowych po akceptacyjne, wydajnościowe)**

Nie wdrożono polityki testowania oprogramowania określającej role, podejście, odpowiedzialność poszczególnych obszarów. NIe ma sprecyzowanych i udokumentowanych zasad projektowania testów (scenariuszy, przypadków, danych testowych) co uniemożliwia monitorowanie przebiegu testów, pokrycia wymagań testami, ryzyka. Metodyka była przygotowana i prezentowana kierownictwu IT w II kwartale 2020 - zabrakło decyzji o jej wdrożeniu na poziomie całej organizacji.

*Rozwiązanie: wdrożyć politykę testowania oprogramowania uwzględniającą również nabywanie usług i produktów na zewnętrze.*

## (ZK) Architektura i Integracja

**Zdzisław Krysztofiak**

## Zarządzanie konfiguracją

1. Jaką politykę organizacja stosuje do zarządzania konfiguracją?
2. Jaka grupa lub jakie osoby odpowiadają za automatyzację i końcową instalację w środowisku produkcyjnym?
3. (ZK) Czy są stosowane narzędzia (np. Ansible, Jenkins, Chef, Puppet) do automatyzacji pipeline’ów wdrożeniowych ?
4. (ZK) Czy wdrożono (gdzie ?) koncepcje zarządzania infrastrukturą/konfiguracja poprzez kod (Infrastruktura jako kod)?

# **Mechanizmy kontroli aplikacji**

1. Cz**y zarządzanie danymi źródłowymi, dokumentami, zbiorem danych jest wystarczające?**
2. **Czy organizacja posiada dostateczne procedury radzenia sobie z błędami?**

Pomimo istnienia mechanizmu informującego o wystąpieniu błędu w środowisku produkcyjnym, nie jest on podejmowany do momentu zgłoszenia przez podmiot zewnętrzny lub pracownika Centrum.

1. **Jak obsługiwane są błędy w danych produkcyjnych?**

Dotychczas błędy produkcyjne były usuwane za zgodą dyrektora departamentu merytorycznego odpowiedzialnego za dany konkurs, dyrektora odpowiedzialnego za obszar teleinformatyczny. Błędy rzadko podlegały analizie, dzięki której można było ich uniknąć w przyszłości. Znacząca część zadań zgłaszanych jako błędy produkcyjne wynikają z pomyłek wnioskodawców, ekspertów oraz pracowników NCBR. Duża ich część dotyczy również aktualizacji danych, np. zmiana nazwy firmy, beneficjenta.

1. **W jaki sposób zarządza się autoryzacją wprowadzania danych do aplikacji?**

W części przypadków, w celu poprawienia danych w środowisku produkcyjnym wymagana jest zgoda dyrektora departamentu merytorycznego odpowiedzialnego za dany konkurs, dyrektora odpowiedzialnego za obszar teleinformatyczny. Niektóre przypadki są realizowane bez tych zgód - np. zmiany nazwy beneficjentów. Nie wiadomo czy dokonywana jest weryfikacja z beneficjentem o zasadności dokonywania zmian - istnieje ryzyko ataku socjotechnicznego pozwalającego przejąć dane podmiotu atakowanego.

1. **Czy reguły i wymogi procesów biznesowych są odpowiednio mapowane do aplikacji?**

Nie.

1. **Czy mechanizmy w aplikacji gwarantują integralność i kompletność transakcji?**

Nie. Pomimo zastosowania relacyjnych baza danych, dane znajdujące się w nich nie są powiązane w żaden sposób kluczami. Powoduje to zawieszanie się procesu zasilania hurtowni danych. W połowie kwietnia 2021 roku postanowiono sukcesywnie uzupełniać bazy danych o więzy integralności.

1. **Czy mechanizmy umożliwiające śledzenie w aplikacji są wystarczające do tego celu?**

Pomimo dość skutecznej metody prowadzenia audit-logu, analitycy, testerzy i w znacznej części programiści nie posiadają dostępu do niego co najmniej w trybie do odczytu.

Dane znajdujące się w logach aplikacyjnych są nieczytelne i nie przedstawiają stanu faktycznego (np. błędy krytyczne są opisane jako błędy informacyjne i odwrotnie).

1. Przykładowo: projekt InfoExpert, projekt Wniosku o Zmianę [↑](#footnote-ref-1)