|  |  |
| --- | --- |
| Politechnika Warszawska  Wydział Elektroniki i Technik Informacyjnych  Instytut Informatyki | Rok akademicki 2010/2011 |

Praca Dyplomowa Inżynierska

Adrian Wiśniewski

System Service Desk zgodny z zaleceniami ITIL:

Zarządzanie komponentami usług i konfiguracją oraz zarządzanie zmianą

Opiekun pracy:

dr inż. Michał Rudowski

|  |  |
| --- | --- |
| Ocena |  |
| Podpis przewodniczącego  Komisji Egzaminu Dyplomowego |  |

|  |  |
| --- | --- |
|  | Kierunek: Informatyka  Specjalność: Inżynieria Systemów Informatycznych  Data urodzenia: 24 grudnia 1987 r.  Data rozpoczęcia studiów: październik 2006 r. |

Życiorys

Urodziłem się 24 grudnia 1987 r. w Nowym Dworze Mazowieckim. Mieszkam w Wołominie, gdzie ukończyłem Sportową Szkołę Podstawową nr 5 im. C. K. Norwida i Sportowe Gimnazjum nr 3 im. Polskich Olimpijczyków. Następnie uczęszczałem do klasy o profilu matematyczno-fizycznym w IV Liceum Ogólnokształcącym im. Adama Mickiewicza w Warszawie. W październiku 2006 r. rozpocząłem studia na Politechnice Warszawskiej na Wydziale Elektroniki i Technik Informacyjnych na kierunku Informatyka.

Rok później podjąłem pracę w firmie Psiloc Sp. z o. o., gdzie pracując jako programista aplikacji internetowych tworzyłem interfejs aplikacji Locatik, służącej do śledzenia aktualnego miejsca przebywania znajomych za pomocą telefonów komórkowych. Po ukończeniu tego projektu w 2008 r. przez kilka lat wykonywałem zlecenia dotyczące aplikacji mobilnych i internetowych. W okresie tym tworzyłem oprogramowanie dla marek takich jak McDonald’s, Allegro, czy gazeta.pl. Obecnie jestem zatrudniony w firmie tworzącej gry komputerowe 11 bit studios S.A. i zajmuję się tematyką sieci komputerowych oraz aplikacji rozproszonych.

Od 2006 roku należę do społeczności twórców gier komputerowych www.gamedev.pl, a w latach 2009 – 2010 pełniłem rolę administratora i członka zarządu Koła Naukowego Twórców Gier Komputerowych „Polygon”.

|  |  |
| --- | --- |
|  | Podpis studenta |

Egzamin dyplomowy

Złożył egzamin dyplomowy dn. 2011 r.

Z wynikiem

Ogólny wynik studiów

Dodatkowe wnioski i uwagi Komisji

Streszczenie

ITIL to obecnie najpopularniejsza metodyka zarządzania infrastrukturą informatyczną przedsiębiorstwa, zorientowana na zarządzanie usługami i należąca do rodziny ITSM. Praca zawiera opis jej struktury i procesów, ze szczególnym naciskiem położonym na scharakteryzowanie zarządzania komponentami usług i konfiguracją oraz zarządzania zmianą. Stworzony system jest aplikacją internetową napisaną w Javie 6 EE przy użyciu biblioteki Spring i posiada moduły do obsługi wyżej wymienionych procesów. Jego praktyczne zastosowania przedstawiono na przykładzie opisanej firmy. Praca omawia wszystkie etapy powstawania systemu od analizy wymagań, przez projekt i implementację do testowania. Ponadto jest uzupełnieniem i kontynuacją pracy inżynierskiej Piotra Kalańskiego, skoncentrowanej na obsłudze funkcji Service Desk — zarządzania incydentami i zarządzania problemami.

*Słowa kluczowe:*

ITIL, ITSM, Service Desk, Zarządzanie komponentami usług i konfiguracją, SACM, Zarządzanie zmianą

Abstract

ITIL is presently the most popular information technology management framework. It is service management oriented approach that belongs to ITSM family. This thesis includes description of ITIL’s structure and all processes, with special focus on Service Asset and Configuration Management (SACM) and Change Management. The new-created system is a web application written using Java 6 EE and Spring framework. It has modules, which support these processes. It’s practical applications are shown on sample business instance. Thesis describes all stages of development: requirement analysis, design, implementation and tests. Moreover it’s complementary to engineer thesis written by Piotr Kalański, which focuses on Service Desk function, especially incident management and problem management.

*Keywords:*

ITIL, ITSM, Service Desk, Service Asset and Configuration Management, SACM, Change Management

Spis treści

[Rozdział 1. Wstęp 7](#_Toc301705013)

[1.1. Cel pracy 8](#_Toc301705014)

[1.2. Zakres 9](#_Toc301705015)

[Rozdział 2. ITIL 10](#_Toc301705016)

[2.1. Pojęcia podstawowe 11](#_Toc301705017)

[2.1.1. Usługa 11](#_Toc301705018)

[2.1.2. Zarządzanie usługami IT (ITSM) 12](#_Toc301705019)

[2.1.3. Proces 12](#_Toc301705020)

[2.1.4. Funkcja 13](#_Toc301705021)

[2.1.5. Rola 13](#_Toc301705022)

[2.2. Historia 14](#_Toc301705023)

[2.2.1. Wersja pierwsza 14](#_Toc301705024)

[2.2.2. Wersja druga 15](#_Toc301705025)

[2.2.3. Wersja trzecia 17](#_Toc301705026)

[2.3. Struktura 18](#_Toc301705027)

[2.3.1. Strategia usług (Service Strategy) 20](#_Toc301705028)

[2.3.2. Projektowanie usług (Service Design) 21](#_Toc301705029)

[2.3.3. Przekazanie usług (Service Transition) 22](#_Toc301705030)

[2.3.4. Eksploatacja usług (Service Operation) 23](#_Toc301705031)

[2.3.5. Ustawiczne doskonalenie usług (Continual Service Improvement) 25](#_Toc301705032)

[2.4. Wyszczególnione procesy 26](#_Toc301705033)

[2.4.1. Zarządzanie komponentami usług i konfiguracją 26](#_Toc301705034)

[2.4.2. Zarządzanie zmianą 30](#_Toc301705035)

[2.5. Inne metodyki 35](#_Toc301705036)

[Rozdział 3. Firma RedHost 36](#_Toc301705037)

[3.1. Charakterystyka 36](#_Toc301705038)

[3.2. Wprowadzenie zarządzania konfiguracją 37](#_Toc301705039)

[3.2.1. Plan zarządzania konfiguracją 37](#_Toc301705040)

[3.2.2. Model konfiguracji 39](#_Toc301705041)

[3.3. Wprowadzenie zarządzania zmianą 41](#_Toc301705042)

[3.3.1. Plan zarządzania zmianą 41](#_Toc301705043)

[3.3.2. Model zmian zwykłych 43](#_Toc301705044)

[3.3.3. Model zmian pilnych 44](#_Toc301705045)

[3.3.4. Model zmian standardowych 44](#_Toc301705046)

[3.3.5. Kategorie zmian 44](#_Toc301705047)

[Rozdział 4. System Service Desk 45](#_Toc301705048)

[4.1. Wymagania 45](#_Toc301705049)

[4.1.1. Technologia (TECH) 45](#_Toc301705050)

[4.1.2. Architektura (ARCH) 46](#_Toc301705051)

[4.1.3. Rdzeń systemu (CORE) 47](#_Toc301705052)

[4.2. Wykorzystane narzędzia 49](#_Toc301705053)

[4.3. Ogólny projekt architektury 51](#_Toc301705054)

[4.3.1. Warstwa logiki aplikacji 52](#_Toc301705055)

[4.3.2. Warstwa prezentacji: witryna internetowa 53](#_Toc301705056)

[4.4. Projekt rdzenia systemu 55](#_Toc301705057)

[4.4.1. Schemat bazy danych 55](#_Toc301705058)

[4.5. Implementacja rdzenia systemu 58](#_Toc301705059)

[4.5.1. Pakiety 58](#_Toc301705060)

[4.5.2. Aspekty 59](#_Toc301705061)

[Rozdział 5. Moduł zarządzania konfiguracją 60](#_Toc301705062)

[5.1. Wymagania 60](#_Toc301705063)

[5.2. Projekt 62](#_Toc301705064)

[5.2.1. Schemat bazy danych 62](#_Toc301705065)

[5.3. Implementacja 67](#_Toc301705066)

[5.3.1. Pakiety 67](#_Toc301705067)

[5.4. Testy 68](#_Toc301705068)

[5.5. Rozwój 68](#_Toc301705069)

[Rozdział 6. Moduł zarządzania zmianą 69](#_Toc301705070)

[6.1. Wymagania 69](#_Toc301705071)

[6.2. Projekt 71](#_Toc301705072)

[6.3. Implementacja 71](#_Toc301705073)

[6.4. Testy 71](#_Toc301705074)

[6.5. Rozwój 71](#_Toc301705075)

[Rozdział 7. Użytkowanie systemu 72](#_Toc301705076)

[Rozdział 8. Testy i ocena wydajności 73](#_Toc301705077)

[Rozdział 9. Podsumowanie 74](#_Toc301705078)

[Bibliografia 75](#_Toc301705079)

[Dodatek A. Zawartość płyty CD 77](#_Toc301705080)

[Wykaz skrótów 78](#_Toc301705081)

[Spis Tabel 79](#_Toc301705082)

[Spis ilustracji 80](#_Toc301705083)

# Wstęp

W dobie społeczeństwa informacyjnego komputery stały się nieodłącznym elementem naszego życia. Są dziś atrybutem nowoczesności i motorem napędowym postępu, a ich ogromny wpływ na zmiany w gospodarce jest niekwestionowany. Przedsiębiorcy od największych korporacji do małych firm korzystają z informatyki, która jest już nierozerwalnym elementem biznesu. Systemów tworzonych na potrzeby przedsiębiorczości przybywa każdego dnia i są one coraz bardziej skomplikowane biorąc pod uwagę zarówno złożoność rozwiązywanych problemów, ilość wykorzystywanych zasobów, jak i integrację z innymi systemami.

Dynamiczny rozwój infrastruktury informatycznej przedsiębiorstw stawia nowe wyzwania w dziedzinie zarządzania. Dotychczas zadania działu IT skupiały się na technologii i dostarczaniu pojedynczych elementów takich jak aplikacja, serwer, sieć, czy baza danych. Postępowanie takie skutkuje rozproszeniem infrastruktury w przedsiębiorstwie, co z kolei znacznie utrudnia zarządzanie. Brak jednolitych procedur uniemożliwia efektywne przeprowadzanie audytów oraz skutkuje obniżeniem jakości świadczonych usług, co kończy się awariami. Ponadto zasoby systemów są od siebie odizolowane i niemożliwy jest ich transfer, a w konsekwencji dopasowanie do zmiennego w czasie zapotrzebowania użytkowników. Systemy przez większość czasu nie korzystają z pełni dostępnej mocy, co jest marnotrawstwem i może być postrzegane jako strata dla przedsiębiorstwa.

Model ITSM (ang. IT service management) podchodzi do problemu w zupełnie odwrotny sposób. Traktuje on infrastrukturę IT jako całość i skupia się na postrzeganiu jej z perspektywy użytkownika. Nie jest ważne jak i z czego zbudowane są poszczególne systemy, natomiast kluczowe znaczenie ma fakt, że świadczą one usługi, istotne dla osiągnięcia celów biznesowych przedsiębiorstwa. Usługą IT nazywamy zbiór sprzętu komputerowego, oprogramowania, łącz oraz innych elementów infrastruktury; które razem z procesami i realizującym je personelem, zapewniają spójną funkcjonalność pozwalającą uzyskać zamierzony efekt. Warto zwrócić uwagę, że definicja ta ma charakter uniwersalny. Pozwala to na grupowanie typowych czynności związanych z zarządzaniem usługami IT w procesy, które dzięki swojej ogólności mogą być stosowane niezależnie od specyfiki przedsiębiorstwa. Ponadto funkcjonowanie działu IT w roli dostawcy usług pozwala skupić się bezpośrednio na celach biznesowych, ponoszonych kosztach oraz wymaganej jakości; podczas gdy kwestie techniczne są rozstrzygane wewnątrz departamentu informatycznego przez odpowiednio wykwalifikowany personel.

ITSM jedynie definiuje sposób w jaki należy postrzegać infrastrukturę informatyczną. Nie jest gotową receptą na problemy zarządzania IT, lecz jako nowatorskie podejście stanowi punkt wyjścia dla konkretnych sposobów organizacji usług i poświęconych temu zagadnieniu metodyk. Bez wątpienia obecnie najbardziej popularną i najszerzej stosowaną z nich jest ITIL (ang. Information Technology Infrastructure Library). Jest to kodeks postępowania opracowany przez OGC[[1]](#footnote-1), znane z wprowadzenia innych modeli zarządzania takich jak PRINCE2[[2]](#footnote-2), MSP[[3]](#footnote-3), czy M\_o\_R[[4]](#footnote-4). ITIL składa się ze zbioru najlepszych praktyk, wypróbowanych w wielu przedsiębiorstwach, co zarazem stanowi jego główną zaletę. Nie narzuca sztywnych wytycznych, przez co może być przystosowany do różnych potrzeb. Ponadto — jako metodyka wytworzona przez organizacje rządowe — nie jest własnością żadnej firmy, dzięki czemu jest niezależna od platformy technologicznej i specyfiki branży.

## Cel pracy

Stworzenie prototypu systemu opartego o ITIL w wersji 3 do zarządzania infrastrukturą informatyczną przedsiębiorstwa oraz zobrazowanie zastosowań tego systemu na podstawie przypadków użycia w przykładowej firmie.

Na rynku istnieją gotowe, skomplikowane i duże produkty do zarządzania usługami IT oparte o ITIL, a ogromne koszty licencji sprawiają, że są one dostępne jedynie dla największych firm. Dodatkowo często występującym problemem jest brak możliwości adaptacji systemu do zmieniającej się charakterystyki przedsiębiorstwa. Jest to spowodowane przechowywaniem informacji na temat elementów konfiguracji w relacyjnych bazach danych w tradycyjny sposób, gdzie informacje o elementach jednego typu znajdują się w jednej tabeli. Schemat bazy danych tworzony jest na sztywno, a w konsekwencji dodanie nowych atrybutów lub typów elementów wymaga zarówno zmian w bazie danych jak i modyfikacji systemu. Moja praca dyplomowa jest próbą stworzenia niedrogiego, intuicyjnego i adaptowalnego narzędzia, które może być stosowane także przez średnie i małe przedsiębiorstwa.

## Zakres

ITIL jest standardem złożonym z bardzo dużej liczby procesów. Wykonanie systemu wspierającego wszystkie byłoby zadaniem przerastającym siły niejednej firmy, nie wspominając o możliwościach dyplomanta. W związku z tym z całego standardu zostały wybrane najważniejsze procesy związane z przekazaniem usług i ich eksploatacją.

Następnie procesy z obu tych grup zostały równolegle zaimplementowane w ramach dwóch prac dyplomowych inżynierskich. Procesy z grupy przekazania usług — zarządzanie komponentami usług i konfiguracją oraz zarządzanie zmianą — są opisane przez autora w niniejszej pracy. Pozostałe procesy z drugiej grupy — zarządzanie incydentami i problemami — zostały opisane przez Piotra Kalańskiego [1].

Należy zaznaczyć, że w ramach współpracy autorów obu prac powstał jeden system, w którym wszystkie zaimplementowane procesy współdziałają ze sobą. Prace związane z projektowaniem i implementacją współużytkowanych części systemu oraz opisaniem przykładowej firmy zostały wykonane wspólnie.

Praca została podzielona na dziewięć rozdziałów, z których pierwsze trzy stanowią wprowadzenie teoretyczne, a pozostałe sześć opisuje zaimplementowany system. Następujący po wstępie rozdział drugi przedstawia kodeks postępowania ITIL ze szczególnym uwzględnieniem zarządzani konfiguracją i zmianą. W rozdziale trzecim przedstawiono profil przykładowej firmy, która zamierza wprowadzić ITIL wraz z systemem informatycznym automatyzującym jego procesy. Rozdziały czwarty, piąty i szósty omawiają kolejne etapy wykonania tego systemu, odpowiednio: zebranie wymagań funkcjonalnych i niefunkcjonalnych, projekt oraz implementację. Rozdział siódmy prezentuje zastosowania systemu po jego wdrożeniu w opisanej wcześniej firmie. Rozdział ósmy skupia się na omówieniu testów poprawności i wydajności implementacji. Kończący pracę rozdział dziewiąty stanowi domknięcie i podsumowanie całości. Wszystkie nazwy związane z ITIL występujące w tej pracy zostały przetłumaczone zgodnie z oficjalnym słownikiem wydanym przez OGC [2].

# ITIL

Information Technology Infrastructure Library to kodeks postępowania pomagający w zarządzaniu infrastrukturą informatyczną w firmie. Może być stosowany zarówno w samodzielnych przedsiębiorstwach IT, jak i wewnętrznych działach informatycznych niezależnie od branży. Postrzega infrastrukturę informatyczną z perspektywy użytkownika i określa wszystkie świadczenia IT mianem usług. Tym samym wpisuje się w dziedzinę metodyk zarządzania usługami IT (ITSM), w ramach której bez wątpienia jest najpopularniejszym opracowaniem. Używają go tysiące firm na całym świecie zarówno z sektora prywatnego, jak i rządowego. Swoją popularność zawdzięcza temu, że stanowi zbiór sprawdzonych i najlepszych praktyk, napotkanych do tej pory w biznesie. ITIL nie narzuca sztywnych wytycznych. Należy go odczytywać jako zbiór porad i wskazówek, które w razie potrzeby mogą być dostosowane do specyficznych potrzeb i możliwości.

Najmniejszą częścią ITIL są pojedyncze czynności wykonywane przez pracowników IT. Czynności są łączone w powiązane grupy, a te z kolei w procesy odpowiadające za wybrane aspekty zarządzania usługami. Procesy tworzą pięć obszarów odpowiadających etapom cyklu życia usługi. Obszary te to:

* Strategia usług
* Projektowanie usług
* Przekazanie usług
* Eksploatacja usług
* Ustawiczne doskonalenie usług

Zanim nastąpi ich omówienie, należy zapoznać się z definicjami wielu podstawowych pojęć oraz prześledzić historię ewolucji metodyki. Pozwoli to na dogłębne zrozumienie wszystkich przedstawianych celów, struktur i elementów. Procesy, których implementacja jest przedmiotem tej pracy, zostały wyróżnione i omówione szczegółowo.

W dalszej części rozdziału przyjęto, że organizacją świadczącą usługi informatyczne jest samodzielne przedsiębiorstwo. Nic nie stoi jednak na przeszkodzie, aby wszystko to odnosić do wewnętrznego departamentu IT, ponieważ ITIL może być stosowany tak samo przez oba rodzaje organizacji. Jest to jedynie zabieg redakcyjny mający na celu skrócenie i ujednolicenie tekstu — ciągłe powtarzanie tej alternatywy byłoby co najmniej nurzące.

## Pojęcia podstawowe

### Usługa

Usługa jest głównym pojęciem, na którym oparty jest cały ITIL. Wszyscy posiadają wyobrażenie o istocie tego terminu, jednak warto przytoczyć formalną definicję:

„A means of delivering value to customers by facilitating outcomes customers want to achieve without the ownership of specific costs and risks.”

Sposób dostarczania wartości klientom poprzez ułatwienie otrzymania pożądanych wyników, bez posiadania specyficznych kosztów i ryzyka.

Definicja usługi, źródło: [3]

Aby w pełni zrozumieć tą formułę, trzeba rozwinąć pojęcia wartości, ułatwienia otrzymania wyników oraz posiadania kosztów i ryzyka.

Klienci korzystają z usług, ponieważ chcą osiągnąć pewne cele biznesowe bądź prywatne. Dla przykładu weźmy jednostkę chcącą zamieścić w internecie swoją stronę www. Posiadanie takiej witryny wiąże się z potrzebą zakupu odpowiedniego sprzętu, oprogramowania, łączy oraz utrzymaniem infrastruktury i personelu. Klient natomiast nie chce być za to wszystko płacić i być odpowiedzialny. Rezultatem tej potrzeby jest powstanie na rynku przedsiębiorstw, które zajmują się wykonywaniem prac związanych z hostingiem i przejmują na siebie związane z nimi ryzyko. Ponadto dzięki specjalizacji są w stanie zmniejszyć koszty poprzez ich rozłożenie na wielu odbiorców usługi.

Jednak żeby konsument chciał skorzystać z takiej oferty musi być spełniony jeszcze jeden warunek. Świadczona usługa musi, w odczuciu klienta, być potrzebna i pozytywnie wpływać na spełnienie jego celów. W ten sposób usługodawca wytwarza wartość, za którą odbiorca usługi płaci. Ten specyficzny towar składa się z dwóch podstawowych elementów. Pierwszym z nich jest użyteczność, określająca w jakim stopniu usługa spełnia oczekiwania klienta i generuje pożądane wyniki. Polega zwykle na przyspieszeniu wykonywania zadań, podniesieniu jakości wytwarzanych produktów lub usunięciu różnego rodzaju ograniczeń. Drugim składnikiem jest gwarancja, która zapewnia dostępność usługi, odpowiednią pojemność, ciągłość świadczenia i bezpieczeństwo. Nawet najbardziej użyteczna usługa może nie posiadać wartości dla klienta, gdy jest niedostępna, wadliwa lub niebezpieczna w użyciu.

### Zarządzanie usługami IT (ITSM)

ITSM (ang. IT Service Management) to dziedzina metod zarządzania infrastrukturą IT, postrzegająca ogół czynności świadczonych przez personel informatyczny w kategorii usług. Odcina się od tradycyjnego podejścia skupiającego się technologii. Dla niej najważniejsze są potrzeby użytkownika, czemu wyraz daje koncentracja na ogóle komponentów potrzebnych do osiągnięcia zamierzonego celu, a nie pojedynczych elementach infrastruktury. Dzięki ustanowieniu ról usługodawcy i usługobiorcy można jasno określić zbiór obowiązków i schematy komunikacji między oboma grupami. Podstawą interakcji są różnego rodzaju umowy, na podstawie i w ramach których następuje świadczenie usług. Ponadto podział ten umożliwia ukrycie wszystkich zbędnych detali technicznych wewnątrz organizacji usługodawcy. Usługobiorca jest dzięki temu zwalniany z obowiązku podejmowania wielu szczegółowych decyzji. W efekcie pozwala to na specjalizację dostawcy oraz przeniesienie i rozproszenie kosztów oraz ryzyka. Takie podejście zaskakująco dobrze współgra z zastosowaniem chmur obliczeniowych[[5]](#footnote-5), obecnie bardzo popularnych.

Czytając materiały można często natknąć się na zjawisko utożsamiania ITIL i ITSM. Jest to błąd mający swoją przyczynę w tym, że ITIL był pierwszą i swojego czasu jedyną znaną metodyką tego typu. Wraz z jego rozwojem i powstaniem konkurencyjnych praktyk, zaczęto pojęcia te doprecyzowywać i odróżniać od siebie. Należy pamiętać, że ITSM sam w sobie nie stanowi gotowego rozwiązania, ale dostarcza niezbędną podstawę pojęciową, na której wyrosły różne metodyki.

### Proces

„A structured set of Activities designed to accomplish a specific Objective.   
A Process takes one or more defined inputs and turns them into defined outputs.   
[…]”

Uporządkowany zestaw czynności mających przeznaczony do osiągnięcia konkretnego celu. Proces posiada jedno lub więcej określonych wejść, które zamienia w określone wyjścia. […]

Definicja procesu [4]

Podstawowym elementem struktury ITIL jest proces. Wygodnym sposobem rozumienia tego pojęcia jest model czarnej skrzynki[[6]](#footnote-6), która przyjmuje dane, a następnie je przetwarza i zwraca wyniki. Wejściami procesu mogą być na przykład wnioski, dokumenty, pewne okoliczności lub zjawiska; które są przekształcane w nowe lub zaktualizowane obiekty i inne zdarzenia. Operacje wykonywane wewnątrz procesu zależą ściśle od jego specyfiki i nie są w żaden sposób ograniczone. Najmniejszym elementem przetwarzania jest pojedyncza czynność. Czynności są wykonywane przez określone role i mogą być łączone w grupy oraz wykonywane w ściśle określonym porządku. W procesie uczestniczy zazwyczaj wiele różnych osób i funkcji. Ponadto każdy proces posiada właściciela[[7]](#footnote-7), odpowiedzialnego za jego działanie zgodnie z wyznaczonym celem, oraz zestaw udokumentowanych norm, zasad i metryk.

### Funkcja

„A team or group of people and the tools they use to carry out one or more Processes or Activities […]”

Zespół lub grupa ludzi oraz narzędzia przez nich używane do realizacji jednego lub więcej procesów lub czynności […]

Definicja funkcji [5]

Niektóre przedsiębiorstwa do określenia funkcji używają nazw takich jak jednostka biznesowa, departament lub dział. Typowymi funkcjami są marketing, sprzedaż, zasoby ludzkie itp. Klasycznym przykładem funkcji wprowadzanej przez ITIL jest Service Desk. Jego zadaniem jest obsługa klientów i ich żądań. Zatrudnia wielu konsultantów oraz wyspecjalizowanych pracowników. Posiada system informatyczny rejestrujący rozmowy oraz umożliwiający wprowadzenie różnego rodzaju wniosków. A także uczestniczy w procesach zarządzania incydentami i realizacji wniosków.

### Rola

Jest to obowiązek wykonywania określonego zbioru czynności, wraz z niezbędnymi uprawnieniami, nadawany pojedynczej osobie lub grupie. Role są określane przez procesy. Jedna osoba może pełnić więcej niż jedną rolę.

## Historia

Prace nad ITIL zaczęły się około trzydzieści lat temu — w drugiej połowie lat osiemdziesiątych. Rząd Wielkiej Brytanii szukał sposobów na ograniczenie wydatków związanych z utrzymaniem systemów informatycznych, które sięgały kwoty około 8 miliardów funtów w skali roku [6]. Źródła wysokich kosztów dopatrywano się przede wszystkim w fakcie, że wszystkie departamenty i instytucje rządowe zarządzały infrastrukturą IT we własnym zakresie. Nie istniały jednolite procedury i standardy, a co za tym idzie możliwości przeprowadzania audytów i kontroli wydatków były znacznie ograniczone. W celu rozwiązania problemu zlecono komórce rządowej o nazwie Central Computer and Telecommunications Agency (CCTA) opracowanie metody zarządzania pozwalającej na efektywne wykorzystanie zasobów IT.

Tak w roku 1988 powstał GITIMM (ang. Government Information Technology Infrastructure Management Method) będący poprzednikiem ITIL [7]. Myślą przewodnią nowej metodyki było ujednolicenie sposobu zarządzania IT. Dzięki wprowadzeniu wspólnego języka i stosowaniu podobnych praktyk możliwa stała się profesjonalizacja pracowników zarządzających infrastrukturą IT, co w konsekwencji pozwoliło podnieść efektywność i obniżyć koszty. Ponadto metodyka zawierała bardzo wiele cennych i sprawdzonych porad opartych na doświadczeniach wielu różnych firm, z którymi prowadzono konsultacje. W szczególności skorzystano z opracowanych przez IBM i wydanych w 1980 r. publikacji z serii „A Management System for the Information Business” opisujących koncepcję ITSM. GITIMM spełnił oczekiwania i został szybko przyjęty przez sektor prywatny oraz instytucje rządowe nie tylko Wielkiej Brytanii.

### Wersja pierwsza

Rok później, w sierpniu 1989, podczas krótkiej rozmowy dwóch twórców zmieniono nazwę na ITIL [8]. Zmiana była kluczowa, ponieważ z nazwy zniknęło słowo „government” oznaczające rząd, które odstraszało wielu przedsiębiorców. Ponadto zrezygnowano ze słowa „metoda”, ponieważ metodyka ta jest w rzeczywistości zbiorem wskazówek, a nie sztywnych wytycznych. W ciągu kilku lat wydano łącznie 42 tomy składające się na pierwszą wersję ITIL [9]. Do popularyzacji i dalszego rozwoju metodyki przyczyniła się utworzona w 1991 r. w internecie grupa użytkowników ITIMF (ang. IT Infrastructure Management Forum), obecnie funkcjonująca pod nazwą itSMF (ang. IT Service Management Forum) jako organizacja non-profit szerząca wiedzę o ITSM [10].

### Wersja druga

W 2000 roku podjęto próbę uporządkowania i ujednolicenia metodyki. Efektem był podział ITIL na dwa kluczowe obszary zawierające łącznie jedenaście dyscyplin. Pierwszy obszar — wspieranie usług (ang. Service Support) — obejmuje sześć dyscyplin związanych z technologią, niezbędnych do zapewnienia wysokiej jakości świadczonych usług:

|  |  |
| --- | --- |
| Nazwa dyscypliny | Opis |
| Service Desk | Zapewnia pojedynczy punktu kontaktu dla użytkowników końcowych potrzebujących wsparcia. Skupia się na komunikacji z klientem i kontroli incydentów. Jedyna dyscyplina będąca funkcją a nie procesem. |
| Zarządzanie incydentami  (Incident Management) | Głównym celem tej dyscypliny jest przywrócenie działania usługi po wystąpieniu incydentu w jak najkrótszym czasie i przy uwzględnieniu obowiązujących SLA[[8]](#footnote-8). Incydent to zdarzenie, które wpływa lub może wpływać na obniżenie jakości świadczonej usługi. |
| Zarządzanie problemami  (Problem Management) | Skupia się na znalezieniu problemów będących źródłem incydentów i podjęciu działań korygujących. W trakcie naprawy udostępnia informacje o znanych błędach i metodach ich obejścia. |
| Zarządzanie zmianami  (Change Management) | Zapewnia standardowe procedury zarządzania wszelkimi zmianami dotyczącymi usług. Minimalizuje ich negatywny wpływ poprzez odpowiednią ocenę, autoryzację i planowanie każdej zmiany. |
| Zarządzanie wersjami  (Release Management) | Grupuje zmiany w wersje i odpowiada za skoordynowanie ich implementacji oraz wdrożenia. |
| Zarządzanie konfiguracją  (Configuration Management) | Zajmuje się przechowywaniem i ciągłą aktualizacją informacji o wszystkich komponentach potrzebnych do świadczenia usług. Stanowi bazę wiedzy, z której korzystają inne procesy. |

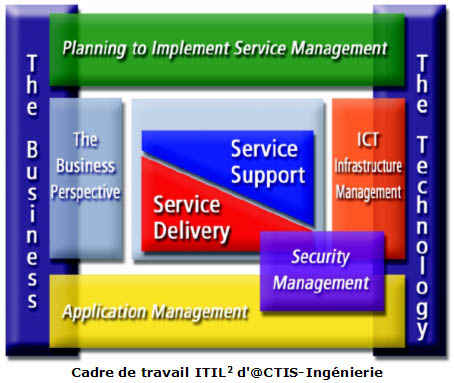
Tabela .. Dyscypliny wspierania usług ITIL v2

Drugi obszar — świadczenie usług (ang. Service Delivery) — jest bliższy perspektywie biznesowej i dotyczy przestrzegania gwarancji danych klientowi przez świadczącego usługi. Składa się z pięciu dyscyplin:

|  |  |
| --- | --- |
| Nazwa dyscypliny | Opis |
| Zarządzanie poziomem usług  (Service Level Management) | Pozwala na dopasowanie usług IT do wymagań biznesu. Skupia się na określeniu przejrzystych oczekiwań i wymagań klienta w stosunku do pobieranych przez niego usług. Dyscyplina ta zajmuje się tworzeniem katalogu usług, zbieraniem wymagań i negocjacją SLA. |
| Zarządzanie finansami  (Financial Management) | Zajmuje się obliczaniem kosztów świadczonych usług, ich księgowaniem oraz ustalaniem budżetu. Pozwala na funkcjonowanie wewnętrznych departamentów IT jako jednostek biznesowych. |
| Zarządzanie dostępnością  (Availability Management) | Celem tej dyscypliny jest zapewnienie dostępności usług zgodnie z SLA w efektywny kosztowo sposób. Wymaga to analizy, planowania, mierzenia i ewentualnej poprawy wszystkich czynników mających wpływ na dostępność. |
| Zarządzanie pojemnością  (Capacity Management) | Dyscyplina podobna do zarządzania dostępnością, zorientowana na monitorowanie i przestrzeganie gwarancji dotyczących pojemności świadczonych usług. |
| Zarządzanie ciągłością  usług informatycznych  (IT Service Continuity Management) | Znana także jako zarządzanie sytuacją kryzysową. Głównym celem jest przywrócenie działania systemów informatycznych firmy w uzgodnionym czasie. Składa się z zapobiegania wystąpieniu sytuacji kryzysowych i opracowaniu planów przeciwdziałania im. |

Tabela .. Dyscypliny świadczenia usług ITIL v2

Po wydaniu pierwszych dwóch tomów dotyczących wsparcia i świadczenia usług powstało jeszcze pięć następnych opisujących inne obszary i dwa dotyczące odpowiednio ITSM oraz stosowania ITIL w małych przedsiębiorstwach. W tym czasie doszło do restrukturyzacji instytucji rządowych Wielkiej Brytanii, na skutek której w 2001 r. OGC (ang. Office of Government Commerce) wchłonęło CCTA i tym samym stało się posiadaczem praw do ITIL. Druga wersja wprowadziła jasno wydzielone procesy i znacząco uporządkowała czynności opisane w wersji pierwszej. Skonsolidowano wiedzę zawartą w czterdziestu dwóch tomach do zaledwie dziewięciu.



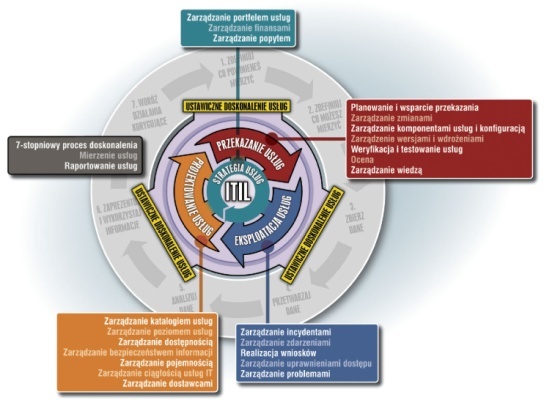
Ilustracja .. Obszary ITIL v2 [11]

### Wersja trzecia

W listopadzie 2004 r. OGC uruchomiło projekt ITIL Refresh [12] mający na celu stworzenie kolejnej wersji ITIL. Projekt polegał głównie na przeprowadzaniu konsultacji z tysiącami organizacji, które używały ITIL, i skorzystaniu z ich opinii oraz doświadczeń. W maju 2007 r. projekt został zakończony, a jego rezultatem było opublikowanie pięciu tomów ITIL w wersji trzeciej. Zmian było bardzo dużo, zarówno w strukturze metodyki, poszczególnych procesach, jak i samej filozofii zarządzania infrastrukturą IT. Zakres poszerzono blisko dwukrotnie, zwiększając liczbę procesów i funkcji prawie trzykrotnie. W lipcu 2011 r. zaprzestano publikacji materiałów ITIL v2 i oficjalnie wycofano wsparcie dla tej wersji. Obecnie, w ramach cyklicznego planu modernizacji, przygotowywane jest pierwsze uaktualnienie ITIL v3 [13].

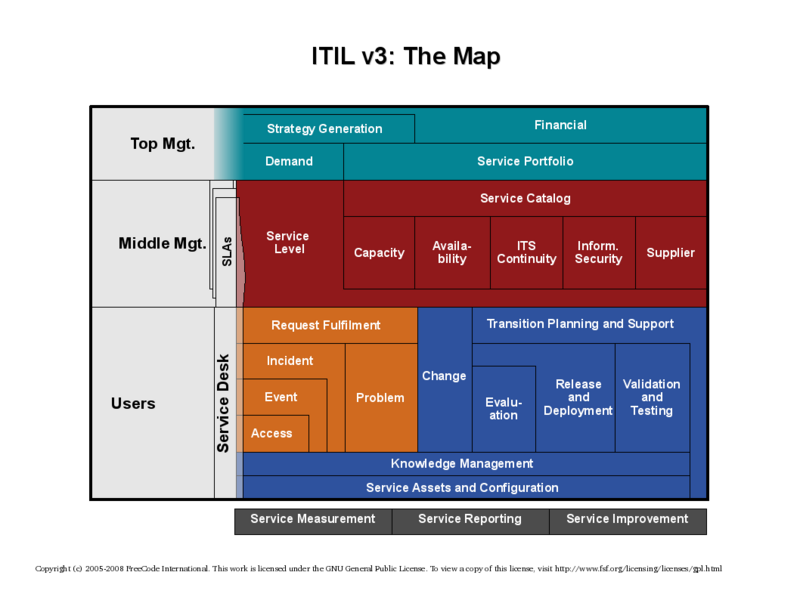
## Struktura

Kluczową zmianą w trzeciej wersji ITIL jest odejście od modelu w którym biznes i IT stanowią oddzielne ośrodki, które ze sobą współpracują. Podział ten zrodził się z przekonania, iż oba pełnią funkcje zasadniczo różniące się od siebie. Z tego powodu nacisk w zarządzaniu był kładziony na dopasowanie IT i biznesu (ang. Business IT Alignment). Niestety wraz z rozwojem technologii i wzrostem złożoności świadczonych usług współpraca między nimi stawała się coraz bardziej zażyła i skomplikowana. Obecnie oba ośrodki są wręcz nierozerwalne i sytuacja ta rodzi szereg problemów. Patrząc na usługę z perspektywy biznesu pomija się techniczne zawiłości, a w konsekwencji naraża na obniżenie jakości lub przerwy w działaniu usługi w związku z niedostateczną dostępnością, pojemnością lub błędami. Z drugiej strony spojrzenie IT nie jest w stanie uchwycić faktycznych potrzeb klienta i podejmować decyzji o charakterze strategicznym. Dlatego ITIL v3, zamiast starać się dopasować biznes i IT do siebie, mówi o ich integracji. Produktem tego innowacyjnego podejścia jest stworzenie modelu cyklu życia usługi IT (ang. IT service lifecycle), w którym obie domeny dopełniają się.



Ilustracja .. Cykl życia usługi w ITIL [14]

Cykl życia usługi w ITIL v3 składa się z pięciu obszarów, którym odpowiada pięć dotychczas wydanych publikacji. Obszary te tworzą trzy koncentryczne, pierścienie, które obracając się wprawiają w ruch pozostałe (patrz: Ilustracja 2.2). Każda faza cyklu życia składa się z zestawu procesów i posiada jasno zdefiniowane wejścia oraz wyjścia, dzięki którym komunikuje się z innymi. Pierwszym i zarazem centralnym obszarem jest strategia usług, odpowiedzialna za określenie portfela usług[[9]](#footnote-9) z uwzględnieniem dalekosiężnych celów przedsiębiorstwa. Wizja usługi stanowi początek, na podstawie którego projektowanie usług opracowuje specyfikację i architekturę rozwiązania, biorąc pod uwagę wymagania jakościowe, wydajnościowe, ryzyko i inne. W ten sposób powstaje pakiet projektu usługi (ang. Service Design Package), który jest przesyłany do obszaru przekazania usług odpowiedzialnego za jej implementację oraz wdrożenie. Działająca usługa znajduje się w obszarze eksploatacji usług. Zadaniem tego etapu jest zagwarantowanie dostarczania usługi zgodnie z uzgodnionymi warunkami. Gdy zajdzie konieczność poważnej modyfikacji, usługa jest ponownie przekazywana do etapu projektowania usług lub wycofywana. Ostatnim obszarem jest ustawiczne doskonalenie usług, które odpowiada za ciągłe podnoszenie jakości i wydajności usług.



Ilustracja .. Procesy ITIL pogrupowane wg hierarchii zarządzania [15]

### Strategia usług (Service Strategy)

„People do not want quarter-inch drills. They want quarter-inch holes.”

Ludzie nie chcą ćwierć-calowych wierteł. Chcą ćwierć-calowych otworów.

***Profesor Emeritus Theodore Levitt, Harvard Business School*** [16 str. 31]

Źródłem sukcesu usługodawcy jest dogłębne rozumienie potrzeb rynku. Klienci chcą kupować zaspokojenie swoich potrzeb, a nie sprzedawane im produkty. Dlatego przed określeniem swojej oferty, usługowca musi zidentyfikować potencjalne rynki zbytu i dokładnie je zbadać. Dopiero na podstawie zebranych informacji jest w stanie określić jakie usługi są potrzebne i będą generować zyski.

Dokładnie tym zajmuje się etap strategii usług, obsługiwany przez najwyższe kierownictwo firmy. Ważnym procesem tego obszaru jest zarządzanie portfelem usług, który składa się z katalogu usług[[10]](#footnote-10) widocznych dla klienta oraz usług niewidocznych: będących w przygotowaniu lub wcześniej wycofanych. To właśnie w jego obrębie zapadają decyzje o tworzeniu, zmianach, zakończeniu lub wznowieniu świadczenia usług.

Aby móc efektywnie zarządzać portfelem usług, trzeba najpierw stworzyć strategię przedsiębiorstwa wyznaczającą cele i kierunek rozwoju w długiej perspektywie czasu. Jest to kluczowy proces pozwalający na specyfikację ram funkcjonowania firmy, określenie docelowych rynków oraz na stworzenie szeregu metryk, według których działanie przedsiębiorstwa będzie mogło być oceniane i w razie potrzeby korygowane. Definiując strategię otrzymuje się jasny obraz firmy, jej bieżącej i docelowej pozycji oraz wytyczne jak drogę te pokonać. Wiedza ta stanowi niezbędną bazę na podstawie której można oceniać usługi zarówno świadczone jak i dopiero tworzone.

Jednak informacje i plany same w sobie nie wystarczą, trzeba jeszcze odnaleźć złoty środek między konsumpcją świadczonych usług oraz kosztami przez nie generowanymi i cały czas go monitorować. Z pomocą przychodzą ostatnie dwa procesy tego obszaru: zarządzanie popytem i zarządzanie finansami. Pierwszy z nich skupia się na monitorowaniu zapotrzebowania oraz pozwala na dostosowywanie pojemności usług zgodnie z wahaniami rynku i długofalowym planem. Drugi kontroluje zyski i wydatki firmy, co pozwala na ustalanie odpowiedniego stosunku ceny do jakości świadczonych usług oraz funkcjonowanie przedsiębiorstwa w uwarunkowany ekonomicznie sposób.

### Projektowanie usług (Service Design)

Celem tej fazy cyklu życia usługi jest, zgodnie z nazwą, projektowanie usług w ramach wizji określonej przez strategię usług. Głównym artefaktem, powstającym w ramach współdziałania wszystkich siedmiu procesów tego etapu, jest katalog usług zawierający informacje o wszystkich świadczonych oraz obecnie implementowanych usługach. Stanowi on także podzbiór portfela usług, który dodatkowo przechowuje dane o usługach planowanych na przyszłość. Procesem zapewniającym przechowywanie informacji o katalogu i ich spójność jest zarządzanie katalogiem usług[[11]](#footnote-11).

W momencie gdy zajdzie potrzeba stworzenia nowej usługi lub zmiany istniejącej, procesem zaczynającym projektowanie usługi jest zarządzanie poziomem usług[[12]](#footnote-12). Jego zadaniem jest negocjacja i zebranie biznesowych wymagań dotyczących usługi oraz zapisanie ich w postaci formalnej jako wymagań dla poziomu usługi (SLR)[[13]](#footnote-13) oraz umów SLA. Następnie na ich podstawie tworzona jest specyfikacja usługi składająca się z: odpowiedniej architektury, procesów, zasad oraz dokumentacji; spełniających teraźniejsze i przyszłe potrzeby przedsiębiorstwa [17]. Pozostałe pięć procesów obszaru projektowania usług pomaga w tym kroku. Zarządzanie pojemnością i zarządzanie dostępnością odpowiadają za określenie infrastruktury potrzebnej do spełnienia odpowiadającym im wymagań zawartych w SLA. Polityka bezpieczeństwa i poufności jest egzekwowana przez zarządzanie bezpieczeństwem informacji[[14]](#footnote-14), natomiast zarządzanie ciągłością usług informatycznych[[15]](#footnote-15) odpowiada za zabezpieczenie usługi przed sytuacjami kryzysowymi i opracowanie planów przeciwdziałania. Ostatnim procesem jest zarządzanie dostawcami[[16]](#footnote-16), które definiuje wymagania względem kooperantów dostarczających komponenty usługi. Ponadto ta piątka procesów pełni rolę proaktywną i na bieżąco monitoruje, czy działające usługi spełniają nałożone na nie wymagania.

W ten sposób powstaje pakiet projektu usługi[[17]](#footnote-17), który zawiera projekt oraz opis oczekiwanych rezultatów, potencjalnego ryzyka i wpływu na inne usługi. Jest to kompletna specyfikacja usługi, według której w kolejnej fazie zostanie zaimplementowana. Aby mieć pewność, że zbudowana usługa faktycznie spełnia wymagania funkcjonalne i jakościowe, równolegle powstaje drugi dokument zawierający kryteria odbioru usługi[[18]](#footnote-18).

### Przekazanie usług (Service Transition)

Po zaprojektowaniu usługi przychodzi czas na jej implementację i wdrożenie. Obie te czynności odbywają się w etapie przekazania usługi, który buduje systemy niezbędne do świadczenia usługi i przygotowuje przedsiębiorstwo do ich przyjęcia.

Centralnym procesem tej fazy cyklu życia usługi jest planowanie i wsparcie przekazania[[19]](#footnote-19). Jego celem jest określenie wszystkich etapów potrzebnych do przekazania usługi i koordynacja ich wykonania. Dzięki skupieniu się na implementacji całej usługi, a nie poszczególnych systemów, można znacznie skuteczniej identyfikować i przeciwdziałać ryzyku. Ponadto w zakresie obowiązków tego procesu leży zabezpieczenie wszystkich potrzebnych zasobów i zapewnienie wsparcia dla zespołów. Należy zwrócić uwagę, że ITIL nie narzuca sposobu prowadzenia poszczególnych projektów i może być używany razem z metodykami takimi jak PRINCE2[[20]](#footnote-20), czy agile. Wybór odpowiedniej, w zależności od rodzaju i rozmiaru projektu, leży w kompetencji firmy.

Gdy usługa zostanie zbudowana musi zostać dokładnie sprawdzona. Zajmuje się tym proces weryfikacji i testowania usług[[21]](#footnote-21) i proces oceny[[22]](#footnote-22). Pierwszy z nich bada zgodność usługi ze specyfikacją oraz potrzebami biznesowymi. Przygotowana usługa jest analizowana zarówno pod kątem użyteczności, jak i gwarancji opisanych w pakiecie projektu usługi. Dokładne testowanie bezpośrednio przekłada się na podniesienie jakości i pozwala uniknąć konsekwencji związanych z ewentualnymi awariami. Drugi sprawdza wydajność usługi względem kryteriów zdefiniowanych w SAC.

Gdy weryfikacja zakończy się pomyślnie, usługa zostaje uruchomiona z pomocą procesu zarządzania wersjami i wdrożeniami[[23]](#footnote-23). Krok ten wymaga zgromadzenia wszystkich potrzebnych zasobów i artefaktów w jeden spójny pakiet wersji[[24]](#footnote-24) i zaplanowania kolejnych etapów wdrożenia zgodnie z ustaloną polityką. Pozwala to skoordynować modyfikacje środowiska operacyjnego spowodowane jednoczesnym wdrażaniem i zmianą wielu usług, co minimalizuje ryzyko zakłóceń w dostawie działających usług. Proces ten może także zapewniać wczesne wsparcie[[25]](#footnote-25) dla wdrożonych usług, szybko rozwiązując problemy powstałe w początkowej fazie życia usługi.

Etap przekazania usług zajmuje się także małymi zmianami w działających usługach. Wszystkie potrzeby modyfikacji są zgłaszane jako wnioski o zmianę (RFC)[[26]](#footnote-26), które trafiają do procesu zarządzania zmianami[[27]](#footnote-27). Następnie są rejestrowane, oceniane i autoryzowane przez kompetentny organ. Umożliwia to podejmowanie rozważnych decyzji, minimalizację ryzyka i szybkie przywracanie usług do stanu sprawności po wykryciu problemów.

Pozostałe dwa procesy, mimo że znajdują się w obszarze przekazania usług, dotyczą całego cyklu życia usługi. Zarządzanie komponentami usług i konfiguracją[[28]](#footnote-28) odpowiada za przechowywanie w bazie zarządzania konfiguracją (CMDB)[[29]](#footnote-29) i uaktualnianie informacji na temat elementów konfiguracji[[30]](#footnote-30), czyli wszystkich obiektów związanych z usługami. Są to między innymi: usługi, składowe usług, zasoby strategiczne, dokumenty firmowe, dokumenty generowane przez inne procesy (SLA, RFC itp.), systemy, elementy infrastruktury IT, oprogramowanie, elementy infrastruktury budowlanej, obiekty potrzebne do świadczenia usług związane ze specyfiką branży i inne. Ponadto w CMDB są trzymane informacje na temat pracowników, kooperantów i klientów związanych z danymi elementami. Dane o wszystkich ważnych dla przedsiębiorstwa elementach możemy wykorzystać do pozyskiwania różnego rodzaju interesujących faktów. Do tego celu służy proces zarządzania wiedzą[[31]](#footnote-31), którego celem jest udostępnianie informacji wspomagających podejmowanie decyzji odpowiednim ludziom w odpowiednim czasie.

### Eksploatacja usług (Service Operation)

Obszar ten zajmuje się dostarczaniem usług klientom zgodnie z ustalonymi w SLA kryteriami. W jego obrębie znajdują się wszystkie wdrożone i działające usługi, które są pielęgnowane i utrzymywane w stanie operacyjnym. Wymaga to obsługi interakcji z użytkownikami, monitorowania stanu infrastruktury i poziomu świadczenia usług oraz ewentualnego podjęcia działań naprawczych.

Kluczowym procesem etapu eksploatacji usług jest zarządzanie zdarzeniami[[32]](#footnote-32). Zdarzenie[[33]](#footnote-33) jest wykrywalnym i rozpoznawalnym zjawiskiem, które może mieć bezpośredni lub pośredni wpływ na świadczenie usługi [18]. Rejestrowanie wszystkich zdarzeń i ocenianie ich wpływu umożliwia podjęcie w odpowiednim czasie, odpowiednich decyzji i kroków minimalizujących ryzyko zakłóceń w dostawie usługi. Większość zdarzeń jest generowana automatycznie przez odpowiednie narzędzia monitorujące.

Jeśli jednak ryzyko awarii zrealizuje się i nastąpi przerwa w świadczeniu lub obniżenie jakości usługi, mamy do czynienia z incydentem[[34]](#footnote-34). Zgłoszenia tego typu są gromadzone przez proces zarządzania incydentami[[35]](#footnote-35), który stara się jak najszybciej przywrócić, określone w SLA, oczekiwane działanie usługi. Incydenty są priorytetyzowane i kategoryzowane, co pozwala na znalezienie powtarzających się zgłoszeń. Źródła incydentów nazywane są problemami[[36]](#footnote-36). Za ich znalezienie, a następnie opracowanie planu naprawczego i jego implementację odpowiada proces zarządzania problemami[[37]](#footnote-37). Gdy problem zostanie zidentyfikowany, tworzony jest rekord[[38]](#footnote-38) w bazie znanych błędów[[39]](#footnote-39). Pomaga to podjąć szybkie działania, w sytuacji jeżeli problem powtarza się co jakiś czas. Ponadto dla znanych błędów może zostać sformułowane obejście problemu, pozwalające użytkownikom w pewnym stopniu zmniejszyć skutki awarii. Efektem końcowym tej procedury jest zgłoszenie wniosku o zmianę.

Innym ważnym procesem tej grupy jest zarządzanie uprawnieniami dostępu[[40]](#footnote-40), które stoi na straży polityki poufności i bezpieczeństwa. Proces ten odpowiada za udostępnienie właściwym użytkownikom, odpowiednich usług, zasobów lub informacji. Wymaga to mechanizmów weryfikacji tożsamości, kontroli uprawnień i śledzenia dostępu.

Wewnątrz obszaru eksploatacji usług mieści się funkcja Service Desk, która — tak jak w wersji drugiej ITIL — odpowiada za kontakt z użytkownikami usług. Może posiadać różne formy organizacji, jednak zdecydowanie najczęściej przyjmuje kształt centrum telefonicznego (ang. Call Center). Pracownicy działu obsługi klienta, oprócz wprowadzania do systemu zasygnalizowanych zgłoszeń incydentów, mogą spełniać drobne żądania — wnioski o usługę[[41]](#footnote-41). Są to małe zmiany, o niewielkim koszcie i ryzyku, z góry zatwierdzone np.: prośba o zmianę hasła, zainstalowanie aplikacji itp.; a także zapytania, skargi i zażalenia. Wykonaniem czynności związanych z ich obsługą i rozwiązaniem zajmuje się proces realizacji wniosków[[42]](#footnote-42). Pozwala to na utworzenie szybkiego kanału komunikacji i dostosowanie świadczonych usług do wymagań klienta.

### Ustawiczne doskonalenie usług (Continual Service Improvement)

Trzecim pierścieniem cyklu życia usługi, który wprawia w ruch wszystkie pozostałe, jest ustawiczne doskonalenie usług. Obszar ten jest odpowiedzialny za ciągłe dostosowywanie się firmy do zmieniających się potrzeb biznesu i pozwala sterować nią zgodnie z wyznaczonymi kierunkami. Aby było to możliwe należy zadać sobie sześć kolejnych pytań [19]:

1. Jaka jest wizja przedsiębiorstwa?
2. Gdzie jesteśmy teraz?
3. Gdzie chcemy być?
4. Jak tam się dostać?
5. Czy już się tam jesteśmy?
6. Jak utrzymać prędkość?

Odpowiedź wymaga uprzednio jasnego zdefiniowania celów i metryk. Podstawowym elementem obszaru ustawicznego doskonalenia usług są właśnie pomiary. Jedynie na podstawie wiarygodnych i miarodajnych informacji można kontrolować i zarządzać. Z pomocą przychodzi proces mierzenia usług[[43]](#footnote-43), który definiuje zasady wyznaczania metryk, zbiera dane z pomiarów, odpowiednio je filtruje i konsoliduje, a w końcu analizuje zgodnie z wyznaczonymi celami. Na podstawie tych badań proces raportowania usług[[44]](#footnote-44) tworzy odpowiednie zestawienia, które są wysyłane do kompetentnych osób podejmujących decyzje o ewentualnych usprawnieniach. Całością zarządza proces ustawicznego doskonalenia usług[[45]](#footnote-45) pracujący według siedmiokrokowego planu poprawy [19]:

1. Zdefiniuj co powinieneś mierzyć.
2. Zdefiniuj co możesz mierzyć.
3. Zbierz dane.
4. Przetwórz dane.
5. Analizuj dane.
6. Pokaż i wykorzystaj informacje.
7. Zaimplementuj akcję korygującą.

## Wyszczególnione procesy

Zakresem pracy objęto dwa procesy należące do obszaru przekazania usług. Są to: zarządzanie komponentami usług i konfiguracją oraz zarządzanie zmianą. Wsparcie dla obu zostało zaimplementowane w stworzonym systemie Service Desk. Ponadto system obsługuje procesy zarządzania incydentami i problemami, które zostały opisane i zaimplementowane w ramach pracy Piotra Kalańskiego [1].

### Zarządzanie komponentami usług i konfiguracją

Cel procesu

Zakres obowiązków tego procesu można streścić bardzo krótko, jest nim administrowanie bazą zarządzania konfiguracją (CMDB)[[46]](#footnote-46). Baza ta zawiera informacje na temat wszystkich ważnych z punktu widzenia przedsiębiorstwa elementów konfiguracji:

„Configuration Item (CI): Any Component that needs to be managed in order to deliver an IT Service […]. CIs typically include IT Services, hardware, software, buildings, people and formal documentation such as Process documentation and SLAs.”

Element konfiguracji (CI): Dowolny komponent, który musi być zarządzany w celu dostarczania usługi IT […]. Elementy konfiguracji zazwyczaj obejmują usługi IT, sprzęt, oprogramowanie, budynki, ludzi i formalną dokumentację, taką jak dokumentacja procesu i SLA.

Definicja elementu konfiguracji [20]

Posiadanie danych o wszystkich potrzebnych komponentach jest bardzo ważne z wielu powodów. Po pierwsze utrzymanie usługi w stanie operacyjnym wymaga identyfikacji i śledzenia stanu wszystkich wykorzystywanych elementów. Niesprawność jednego z nich może równać się zakłóceniom w dostawie całej usługi. Defekt może zostać wywołany przez fizyczną awarię, ale może wystąpić także z powodu nieświadomych lub nieskoordynowanych zmian. Dysponując informacjami możemy temu zapobiec poprzez wyszukanie zależności niosących duże prawdopodobieństwo awarii oraz koordynując zmiany dotyczące elementów używanych przez wiele usług.

Po drugie zbieranie danych znacząco ułatwia odnalezienie komponentów nie spełniających norm jakości: przeciążonych, wadliwych, zużytych; oraz nie spełniających norm prawnych, w szczególności oprogramowania bez ważnej licencji. Przeciwdziałanie tym zjawiskom redukuje koszty awarii i ewentualnych procesów sądowych oraz pośrednio wpływa na podniesienie jakości świadczonych usług.

Po trzecie wszystkie procesy decyzyjne występujące podczas całego cyklu życia usługi — od fazy strategii; przez etap projektowania, wdrażania, konsumpcji; do jej wycofania — nie są w stanie funkcjonować bez informacji o faktycznym stanie elementów konfiguracji. Wyborów świadomych i efektywnych można dokonywać jedynie posiadając obiektywne i miarodajne dane oraz odpowiednią wiedzę opartą o nie. Baza zarządzania konfiguracją stanowi pojedyncze i spójne źródło, które dostarcza tych informacji.

Z powyższych powodów zarządzanie komponentami usług i konfiguracją odgrywa podstawową i kluczową rolę w całym ITIL. Bez bazy zarządzania konfiguracją w praktyce niemożliwe jest udane wprowadzenie innych procesów.

Kluczowe pojęcia

Model konfiguracji[[47]](#footnote-47)

Baza zarządzania konfiguracją przechowuje model, będący odbiciem faktycznego stanu elementów konfiguracji. Przechowywane informacje mogą mieć dowolną szczegółowość, zależną od potrzeb przedsiębiorstwa. Jednak czasami wartość danych przewyższają koszty związane z utrzymaniem ich aktualnego stanu. Dlatego ważnym wyborem, podejmowanym przy wprowadzaniu procesu zarządzania komponentami usług i konfiguracją jest określenie śledzonych elementów. Na przykład komponenty takie jak myszki i klawiatury, ze względu na małą wartość i niskie ryzyko związane z ich awarią, mogą być pominięte w bazie zarządzania konfiguracją. Model konfiguracji ustala kryteria, według których elementy konfiguracji są identyfikowane, klasyfikowane i opisywane. Pozwala to zachować balans między otrzymywanym zyskiem i kosztem pozyskiwania informacji.

Element konfiguracji

Zgodnie z przytoczoną na wstępie definicją element konfiguracji to obiekt ważny dla funkcjonowania usługi. Każdy jest opisywany w bazie zarządzania konfiguracją przez zbiór atrybutów zgodny z ustalonym modelem konfiguracji. Ponadto posiada fizycznie dołączony unikalny identyfikator, którego format zależy od konwencji firmy, pozwalający go jednoznacznie odróżnić od pozostałych i odszukać w systemie. Elementami konfiguracji mogą być:

* Świadczone usługi, ich komponenty i wykorzystywane przez nie zasoby;
* Produkty cyklu życia usług takie jak projekty, plany przeciwdziałania awarii, wersje, zgłoszenia incydentów, żądania zmian, problemy, umowy SLA, SAC i inne;
* Dokumenty formalne określające np. strategię i politykę przedsiębiorstwa, wymagania wewnętrzne lub prawne, harmonogramy prac itp.;
* Ludzie: pracownicy, kooperanci i klienci;
* Elementy wirtualne używane do grupowania i kategoryzacji;
* Komponenty zewnętrzne, nie należące do firmy, w tym konsumowane usługi.

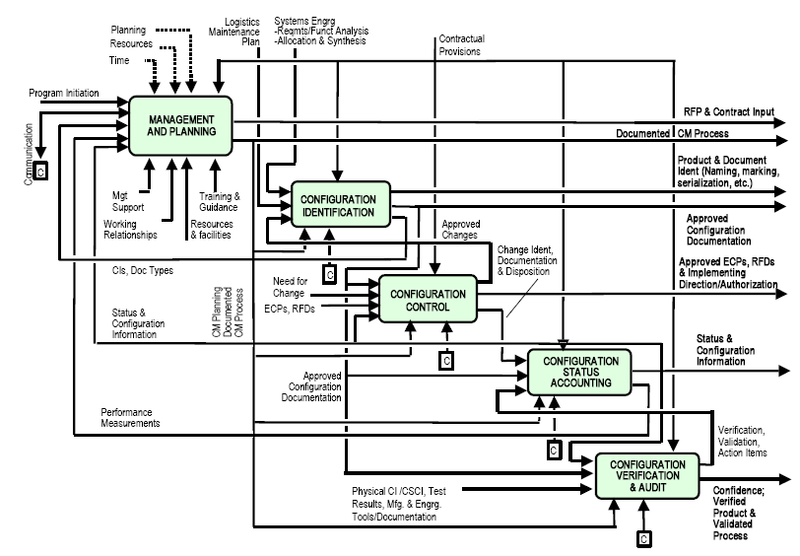
Istotną informację związaną z elementami konfiguracji niosą różnego typu relacje w jakie komponenty wchodzą ze sobą. Na przykład usługi mogą korzystać z systemów informatycznych, te z kolei mogą być złożone z ustalonych komponentów i administrowane przez konkretną grupę ludzi. Powiązane elementy konfiguracji tworzą sieć, w której znajduje się niepoliczalna liczba interesujących faktów i wiedzy. Dzięki niej można szacować wpływ zmian, obliczyć prawdopodobieństwo awarii i doszukiwać się wielu innych zjawisk.

System zarządzania konfiguracją[[48]](#footnote-48)

Jest to system informatyczny fizycznie przechowujący dane elementów konfiguracji. Pozwala na zarządzanie tymi danymi odpowiednio uprawnionym zarządzającym konfiguracją[[49]](#footnote-49). Może korzystać z wielu źródeł takich jak bazy danych, odpowiednio sformatowane dokumenty, narzędzia do automatycznego wykrywania konfiguracji, systemy działające w firmie (np. przechowujące dane pracowników lub klientów) oraz inne.

Czynności

Proces zarządzania konfiguracją zaczyna działanie podczas projektowania usługi w etapie przekazania. Pierwszym jego krokiem jest zarządzanie i planowanie, które polega na opracowaniu planu zarządzania konfiguracją. Zawiera on szczegółowy opis procedur potrzebnych do identyfikacji, kontroli, weryfikacji i audytu elementów konfiguracji danej usługi. Następnie, podczas identyfikacji konfiguracji, powstaje model konfiguracji i zgodnie z nim do systemu zostają wprowadzone dane wszystkich komponentów. Po tym kroku żaden element nie może być modyfikowany bez odpowiednich zezwoleń, a każda zmiana musi zostać odnotowana w bazie zarządzania konfiguracją. Te ograniczenia są niezbędne do zachowania ważności i spójności zapisanych informacji. Za autoryzację wprowadzanych zmian odpowiada kontrola konfiguracji, natomiast za uaktualnienie zapisanego stanu — raportowanie statusu konfiguracji. Ostatnią grupą czynności związaną z tym procesem jest weryfikacja i audyt konfiguracji. Jej celem jest sprawdzenie czy dane przechowywane w bazie zarządzania konfiguracją odzwierciedlają stan faktyczny. Audyty konfiguracji powinny być przeprowadzane w regularnych odstępach czasu oraz dodatkowo przed wydaniem głównych wersji[[50]](#footnote-50) usług.

Ilustracja . Model czynności zarządzania konfiguracją [21]

Powiązane procesy

Zarządzanie komponentami usług i konfiguracją stanowi źródło danych, z którego intensywnie korzysta proces zarządzania wiedzą. Wszystkie opracowania i raporty wytworzone przez ten proces bazują na informacjach zawartych w systemie zarządzania konfiguracją. SACM jest także blisko powiązany z procesami zarządzania zmianą oraz zarządzania wersjami i wdrożeniami. Oba kontrolują i autoryzują zmiany elementów konfiguracji. Ponadto każdy inny proces w pewnym stopniu korzysta z systemu zarządzania konfiguracją wykorzystując jego dane lub wprowadzając do niego swoje produkty.

### Zarządzanie zmianą

Cel procesu

Zmiany są nieodłącznym elementem związanym z świadczeniem usług i systemami IT. Powstają zarówno reaktywnie na skutek wykrycia błędów, jak i proaktywnie w celu redukcji kosztów lub poprawienia jakości. Każda niesie ze sobą ryzyko niepowodzenia i w konsekwencji obniżenia jakości świadczenia usługi.

„The addition, modification or removal of anything that could have an effect on IT Services. The Scope should include all IT Services, Configuration Items, Process, Documentation, etc.”

Dodanie, modyfikacja albo usunięcie czegokolwiek, co może wpływać na usługi IT. Zakres powinien obejmować wszystkie usługi IT, elementy konfiguracji, procesy, dokumentację itp.

Definicja zmiany [22]

Proces zarządzania zmianą wprowadza jednolite procedury obsługi wszystkich żądań zmian, opierające się na rejestracji, ocenie, autoryzacji i ewaluacji. Dzięki wprowadzeniu standardowych metod, wnioski te mogą być obsługiwane i implementowane znacznie szybciej. Pozwala to lepiej dostosować się do potrzeb klienta i biznesu. Ponadto systematyczne podejście umożliwia minimalizację ryzyka wprowadzenia zmian nieautoryzowanych, nieprzemyślanych, szkodliwych oraz ryzyka awarii wynikającego z braku koordynacji. Bezpośrednim skutkiem jest zmniejszenie liczby incydentów oraz przerw w świadczeniu usługi.

Kluczowe pojęcia

Wniosek o zmianę (RFC) [[51]](#footnote-51)

Formalny dokument zawierający propozycję zmiany i jej dokładny opis. RFC posiada odnotowaną datę złożenia, autora, identyfikator oraz opcjonalnie kategorię i wstępny priorytet. Duże zmiany mające znaczący wpływ na przedsiębiorstwo zawierają ponadto uzasadnienie podpisane przez odpowiednie kierownictwo. Wszystkie modyfikacje elementów konfiguracji muszą być zgłoszone w tej postaci. Wniosek o zmianę stanowi podstawę, a jego złożenie punkt początkowy procesu zarządzania zmianą.

Przy wprowadzaniu procesu należy określić zakres zarządzanych zmian. Rozważanie ogromnej ilości drobnych wniosków może negatywnie wpłynąć na wydajność całego przedsiębiorstwa. Z kolei rozpatrywanie zmian o charakterze strategicznym wymaga znacznie większego udziału najwyższego kierownictwa firmy i spełnienia dodatkowych wymagań bezpieczeństwa. RFC zasadniczo powinny dotyczyć usług i elementów konfiguracji.

Oprócz zwykłych wniosków o zmianę istnieją jeszcze dwa inne rodzaje: pilne[[52]](#footnote-52) oraz standardowe. Wnioski z pierwszej grupy stanowią zmiany, które trzeba wykonać natychmiast. Z racji na wysokie ryzyko i duży negatywny wpływ na usługę są one przetwarzane znacznie szybciej, z pominięciem wielu kroków kontrolnych procedury standardowej. Istnienie tego skrótu pozwala na szybszą reakcję na poważne i nieoczekiwane zmiany, w tym awarie. Natomiast zmiany standardowe to małe, częste i powtarzające się modyfikacje, niosące ze sobą niskie ryzyko. Do tego typu zmian kwalifikują się żądania zmiany hasła, wymiany klawiatury i temu podobne. Ich każdorazowa autoryzacja była by niepotrzebnym i nadmiernym obciążeniem dla całego procesu, dlatego odpowiedzialne organy poświadczają je z góry. Obsługą tego typu żądań zajmuje się Service Desk i proces realizacji wniosków.

Rekord zmiany[[53]](#footnote-53)

Wpis w systemie informatycznym wspomagającym proces zarządzania zmianą, tworzony na podstawie zgłoszonego wniosku o zmianę. Odzwierciedla bieżący stan zmiany oraz dokumentuje jej cykl życia, wraz z wszystkimi podjętymi decyzjami. Dodatkowo zawiera wskazania do elementów konfiguracji, na które proponowana zmiana ma wpływ.

Zarządzający zmianami[[54]](#footnote-54)

Rola, którą pełnią pracownicy procesu zarządzania zmianą. W jej kompetencji leży przeprowadzanie rekordów zmian przez cały proces, w tym: filtrowanie niepoprawnych wniosków, przydzielanie kategorii i priorytetu, szacowanie wpływu, koordynacja implementacji oraz recenzja i zamknięcie zmiany. Zarządzający zmianą mogą posiadać prawo do autoryzacji niektórych zmian.

Rada ds. Zmian (CAB)[[55]](#footnote-55)

Organ decyzyjny złożony z przedstawicieli IT, biznesu, kooperantów i użytkowników — jednym zdaniem wszystkich zainteresowanych, na których zmiany mogą mieć wpływ. Dokładny skład CAB zależy głównie od rodzaju rozważanych zmian. Każda kategoria zmian — np. dotycząca usługi A lub B — posiada własną radę, do której trafiają wszystkie rekordy zmian z tej kategorii. Rolą tego organu jest ich ostateczna autoryzacja, nadawanie priorytetów, harmonogramowanie oraz pomoc zarządzającym zmianą w ocenie i podejmowaniu decyzji. Zebrania rady mogą być organizowane elektronicznie, co znacząco oszczędza czas. Jednak w wypadku skomplikowanych i dużych zmian zalecane są bezpośrednie spotkania.

Nadzwyczajna Rada ds. Zmian (ECAB)[[56]](#footnote-56)

Czasami, gdy potrzeba szybkich decyzji, zorganizowanie zebrania Rady ds. Zmian może być niemożliwe. W tym celu istnieje znacznie mniejszy organ, złożony z jak najmniejszej możliwej ilości osób. Organem tym jest Nadzwyczajna Rada ds. Zmian. Posiada ona taką samą charakterystykę, co zwykłe CAB, a jej obowiązkiem jest zarządzanie zmianami pilnymi.

Czynności

Proces zarządzania zmianą zaczyna się od stworzenia i złożenia wniosku o zmianę. Autorem żądania może być każda zainteresowana osoba. Forma zgłaszania wniosków jest dowolna i zależy od przedsiębiorstwa. Może obejmować wnioski pisemne, nadsyłane pocztą elektroniczną oraz wprowadzane przez formularze internetowe.

Następnie wniosek jest odbierany przez zrządzającego zmianą i zapisywany w systemie w postaci rekordu zmiany. Rejestracja wszystkich wniosków jest bardzo ważna. Z jednej strony pozwala wykorzystać wszystkie możliwości ulepszenia usług, a z drugiej pozwala bardzo szybko wychwycić żądania nadmiarowe, które pojawiły się już wcześniej.

Kolejnym krokiem jest przegląd żądań. W tej fazie zarządzający zmianą musi zapoznać się z treścią wniosków i odrzucić wszystkie te, które nie spełniają wymagań formalnych, duplikują się, są niekompletne albo w oczywisty sposób niestosowne. Podjęcie decyzji o oddaleniu wniosku wymaga podania przyczyny, a autorowi żądania przysługuje procedura apelacyjna. Zarządzający konfiguracją może także przekazać wniosek do innej kategorii lub zmienić jego priorytet.

Podczas przeglądu kluczowe jest określenie typu wniosku i związanego z nim sposobu dalszego przetwarzania. Różne rodzaje żądań mogą wymagać innych procedur obsługi. Np. niektóre wnioski mogą wymagać autoryzacji przez wiele różnych osób, a inne mogą być szybko zatwierdzane przez samego zarządzającego zmianą. W tym miejscu proces zarządzania zmianą jest bardzo elastyczny i wszystko zależy od konkretnych potrzeb firmy. Dokładna procedura obsługi wniosku jest nazywana modelem zmiany.

ITIL szczegółowo opisuje sposób przetwarzania zwykłych zmian. Wnioski tego typu po przeglądzie zostają ocenione. Krok ten polega na oszacowaniu potencjalnych zysków, strat i ryzyka oraz wnikliwym przestudiowaniu wpływu zmiany na usługi i elementy konfiguracji. Dokładne jego oszacowanie pozwala uniknąć wielu incydentów i zakłóceń w dostawie usług. Dodatkowo należy wskazać osoby odpowiedzialne za ewentualne wdrożenie zmiany. Istnieje zasada siedmiu R, która pomaga oceniać wnioski [3]. Stawia ona następujące pytania:

1. Kto zgłosił (Raised) wniosek?
2. Jaka jest przyczyna (Reason) zmiany?
3. Jaki zwrot (Return) jest wymagany od zmiany?
4. Jakie są ryzyka (Risks) związane ze zmianą?
5. Jakie zasoby (Resources) są wymagane do dostarczenia zmiany?
6. Kto jest odpowiedzialny (Responsible) za budowanie, testowanie i implementację zmiany?
7. Jaki jest związek (Relationship) pomiędzy tą zmianą i innymi zmianami?

Odpowiedzi na te pytania stanowią ogólną bazę do oceny wniosku, którą można uzupełnić o informacje związane ze specyfiką przedsiębiorstwa, usługi lub samej zmiany. Precyzyjne oszacowanie wpływu jest trudne i często wymaga uwzględnienia stanowiska zainteresowanych grup, dlatego zarządzający zmianą może w każdej chwili poprosić Radę ds. Zmian o pomoc.

Posiadając komplet informacji dotyczący zmiany i wszystkich związanych z nią zjawisk nadchodzi moment, w którym należy podjąć decyzję o jej implementacji. W zależności od czynników takich jak typ, rozmiar, wpływ, czy ryzyko; zmiana jest kierowana do odpowiedniego organu autoryzującego. Najczęściej jest nim Rada ds. Zmian. Jednak przy małych zmianach organem tym może być sam zarządzający zmianą, natomiast przy zmianach o charakterze strategicznym może być to kierownictwo firmy. Organizacja zależy tutaj jedynie od zasad funkcjonowania przedsiębiorstwa.

Wszystkie zatwierdzone zmiany trafiają do Rady ds. Zmian, która jest odpowiedzialna za koordynowanie ich implementacji. Harmonogramowanie z wyprzedzeniem minimalizuje ryzyko awarii związane z jednoczesnym wprowadzaniem wielu zmian oraz, w przypadku potrzeby wyłączenia niektórych komponentów, pozwala na wcześniejsze poinformowanie użytkowników. Implementacją zmiany zajmuje się wskazany budowniczy. Przed przystąpieniem do prac musi on opracować plan pozwalający na ewentualne wycofanie zmiany. Procedura ta jest niezbędna biorąc pod uwagę ryzyko napotkania nieoczekiwanych problemów uniemożliwiających pełne wdrożenie. Po zaimplementowaniu zmiana jest sprawdzana przez testera, po czym CAB udziela zezwolenia na przekazywanie jej do środowiska operacyjnego.

Ostatnim krokiem procesu zarządzania zmianą jest podsumowanie i zamknięcie zmiany. Zarządzający zmianą musi spisać przegląd powdrożeniowy[[57]](#footnote-57), w którym podaje ostateczny status zmiany oraz sprawdza, czy zmiana spełnia stawiane wymagania i pozwala osiągnąć oczekiwane rezultaty lub podaje przyczyny niepowodzenia. Ponadto dokładnie dokumentuje wszystkie powstałe efekty uboczne oraz odchylenia ocen i planów. Wszystkie spisy są przeglądane i oceniane przez Radę ds. Zmian. Pozwala to na wyciągnięcie wniosków i precyzyjniejsze ocenianie w przyszłości. W tym momencie zmiana jest zamykana i proces się kończy.

Powiązane procesy

Zarządzanie zmianą jest bardzo blisko powiązane z zarządzaniem komponentami usług i konfiguracją. Wszystkie autoryzowane zmiany stanowią podstawę do modyfikowania elementów konfiguracji, a dane potrzebne do oceny wniosku o zmianę pochodzą z systemu zarządzania konfiguracją. Zarządzanie wdrożeniami i wersjami odpowiada za przekazanie zaimplementowanej zmiany do środowiska operacyjnego i pozwala grupować powiązane zmiany w wersje. Natomiast proces zarządzania problemami jest źródłem wielu wniosków o zmianę. Dodatkowo zarządzanie zmianą współpracuje z procesami zarządzania ciągłością usług, bezpieczeństwem, pojemnością i popytem. Wszystkie te aspekty muszą zostać uwzględnione podczas oceny wniosku o zmianę.

## Inne metodyki

1 strona

# Firma RedHost

Żeby dokładnie zobrazować ITIL powstała fikcyjna firma Redhost. Przedsiębiorstwo to zajmuje się świadczeniem usług IT i pragnie poprawić swoją konkurencyjność poprzez zastosowanie praktyk ITSM. Oczekuje poprawienia jakości świadczonych usług, redukcji czasu potrzebnego na naprawę awarii, a w konsekwencji wzrostu zadowolenia klientów. W tym rozdziale procesy zarządzania komponentami usług i konfiguracją oraz zarządzania zmianą zostaną skonkretyzowane i dostosowane do potrzeb firmy Redhost. W dalszej części pracy przedsiębiorstwo posłuży do uzasadnienia praktycznych zastosowań stworzonego systemu.

## Charakterystyka

Firma Redhost S.A. świadczy szeroką gamę profesjonalnych usług hostingowych. Jej klientami są zarówno różnej wielkości przedsiębiorstwa, jak i osoby prywatne. Posiada wysokiej klasy centrum danych[[58]](#footnote-58), złożone z czterech serwerowni i łącznie kilkuset serwerów. Należące do tych maszyn moc obliczeniowa, pamięć, powierzchnia dyskowa oraz łącza są udostępniane do dyspozycji klientów. Mogą oni je wykorzystywać do uruchamiania własnych aplikacji i stron www, dostępnych dla użytkowników przez Internet. Zadania firmy polegają na utrzymaniu serwerów w stanie operacyjnym, zgodnie z podpisanymi umowami SLA. Wymaga to zarządzania całą infrastrukturą złożoną z komputerów, sprzętu sieciowego i zasilającego, systemów chłodzących i przeciwpożarowych oraz wielu innych elementów. Kompletny katalog usług firmy Redhost S.A. stanowią:

* Hosting kolokacyjny;
* Hosting dedykowany;
* Wirtualny hosting dedykowany;
* Hosting współdzielony;
* Współdzielony hosting stron internetowych;
* Darmowy hosting stron internetowych;
* Hosting pośredni[[59]](#footnote-59).

Szczegółowy opis przedsiębiorstwa, którego jestem współautorem, znajduje się w [1].

## Wprowadzenie zarządzania konfiguracją

Czynnością niezbędną do wprowadzenia zarządzania komponentami usług i konfiguracją jest stworzenie planu zarządzania konfiguracją, szczegółowo dokumentującego proces (patrz: Ilustracja 2.4). Musi on określać jego cele, zakres, zasady, role i ich obowiązki. Następnie należy zidentyfikować elementy konfiguracji, stworzyć model konfiguracji i napełnić bazę zarządzania konfiguracją danymi. Kolejne fazy procesu dotyczące kontroli, weryfikacji, audytu, mierzenia, oceniania są opisane w przyjętym wcześniej planie.

### Plan zarządzania konfiguracją

Cele

* Zmniejszenie rocznej liczby awarii i czasu przestoju o 30% dzięki identyfikacji wadliwych, przeciążonych oraz niepoprawnie wykorzystywanych komponentów.
* Redukcja kosztów o 15% w skali roku, dzięki zmniejszeniu liczby awarii i lepszemu wykorzystaniu dostępnych zasobów.
* Dostarczenie danych ułatwiających dokładne oszacowanie wpływu awarii oraz zmian poszczególnych komponentów.
* Dostarczenie danych umożliwiających lepsze zarządzanie pojemnością.

Zasady

Ogólne

1. Niniejszy proces zarządzania konfiguracją musi być zgodny z zaleceniami ITIL v3.
2. Narzędziem użytym do wsparcia procesu i przechowywania danych będzie system Service Desk opracowany przez Piotra Kalańskiego i Adriana Wiśniewskiego.
3. Baza zarządzania konfiguracją musi stanowić wiarygodne, spójne i precyzyjne źródło informacji o elementach konfiguracji i zasobach firmy.
4. Będzie używana do:
   1. Określenia wpływu awarii i zmian;
   2. Wsparcia zarządzania pojemnością;
   3. W przyszłości ilość zastosowań się zwiększy.

Zakres

1. Przechowywane są w niej wszystkie elementy konfiguracji niezbędne do świadczenia usług oraz zasoby z wyłączeniem:
   1. Komponentów, które występują w dużych ilościach i są łatwo wymienialne i istnieje ich zapas i niosą ze sobą niskie ryzyko zakłócenia usługi i ich wartości nie przekracza 80 PLN (m. in. kabli zasilających, przewodów, czystych płyt);
   2. Myszek, klawiatur;
   3. Przenośnych pamięci USB;
   4. Narzędzi technicznych typu śrubokręty, lutownice, przecinaki itp.
2. Ponadto baza zarządzania konfiguracją musi zawierać dane:
   1. pracowników i klientów;
   2. usług świadczonych;
   3. usług wykupionych przez klientów wraz z umowami.

Kontrola i role

1. Za stan bazy zarządzania konfiguracją odpowiada proces zarządzania konfiguracją i jego właściciel.
2. Modyfikacji w niej można dokonywać tylko w ramach autoryzowanej zmiany. Modyfikacje są wykonywane przez uprawnionych zarządzający zmianą, mianowani przez właściciela procesu.
3. Każdy element konfiguracji musi posiadać osobę za niego odpowiedzialną.
4. Każdy element konfiguracji posiada unikalny identyfikator.
   1. W formacie: [tekst]/[rok]/[miesiąc]/[dzień]/[numer].
   2. Tekst zależy od typu elementu.
   3. Rok, miesiąc i dzień to data nabycia elementu, zawierająca poprzedzające zera.
   4. Numer jest numerem elementu w ewidencji zakupu.
   5. Np.: SerwerCache/2011/07/24/8.

Audyt i weryfikacja

1. Audyty będą przeprowadzane na zlecenie kierownictwa firmy systematycznie co pół roku. Mogą być przeprowadzane częściej, na odpowiednie żądanie.
2. Wszystkie odkryte nieścisłości muszą być poprawione niezwłocznie. Wykrycie niezgodności ze stanem faktycznym jest zgłaszane jako standardowa zmiana po zakończeniu audytu.

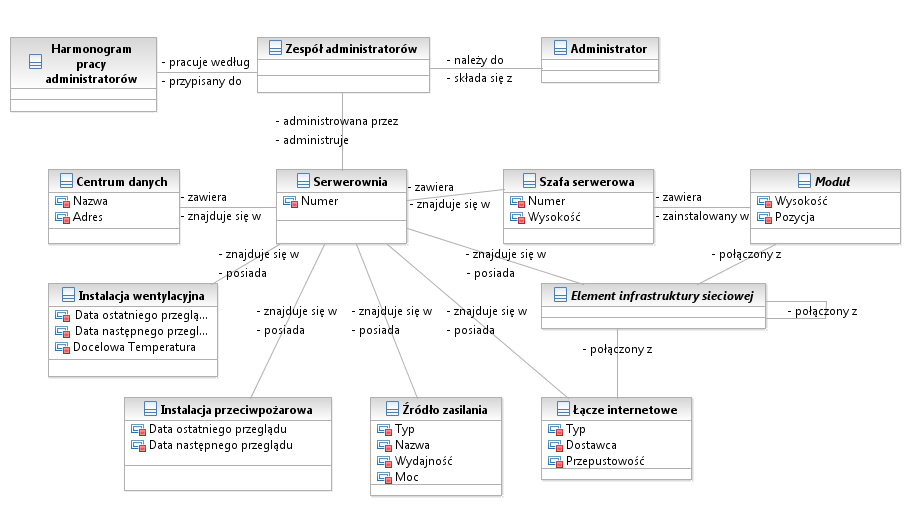
Mierzenie i ocena

1. Ocena sprawności procesu będzie opierała się o stopień wykonania celów.
2. Drugim jej składnikiem będzie mierzony będzie stopień odchyleń od stanu faktycznego, odkryty podczas przeprowadzania audytów.

### Model konfiguracji

Skrupulatne przeanalizowanie całej dziedziny usług hostingowych i dokładne sklasyfikowanie wszystkich elementów konfiguracji do niej należących mogło by zająć miejsce przeznaczone na całkiem grubą książkę. Dlatego w ramach tej pracy skupiono się na głównych komponentach. Można je pogrupować na następujące typy:

* Elementy infrastruktury budowlanej.
  + Centrum danych, serwerownia, skład sprzętu, skład oprogramowania.
* Elementy infrastruktury informatycznej.
  + Źródło zasilania, zasilacz awaryjny, szafa serwerowa, system wentylacyjny.
* Elementy infrastruktury sieciowej:
  + Łącze internetowe, przełącznik, router, mostek, brama, modem.
* Sprzęt:
  + Moduł serwera, pamięć RAM, macierz dyskowa, dysk twardy, procesor.
* Oprogramowanie:
  + System operacyjny, serwer HTTP, baza danych, program użytkowy.
* Komponenty usług:
  + Domena WWW, certyfikat HTTPS.
* Dokumentacja:
  + Umowa SLA, umowa OLA[[60]](#footnote-60), umowa UC[[61]](#footnote-61), licencja na oprogramowanie, RFC, zgłoszenie incydentu, problem, harmonogram pracy.



Ilustracja .. Przykładowe elementy konfiguracji firmy

## Wprowadzenie zarządzania zmianą

Podstawą zarządzania zmianą, jak i każdego innego procesu ITIL, jest formalna dokumentacja specyfikująca proces. Musi ona w szczególności precyzować format wniosków o zmianę, sposób ich zgłaszania i obsługi. Najważniejszym jej elementem jest określenie typów zmian. Typ zmiany determinuje sposób jej przetwarzania, nazywany modelem zmiany. Określenie kilku typów i odpowiadających im modeli zmian pozwala traktować różne zmiany w inny sposób. ITIL określa minimalny zestaw i podaje trzy predefiniowane typy zmian: zmiany zwykłe, pilne i standardowe. Nic nie stoi na przeszkodzie, aby uzupełnić ten zbiór w zależności od konkretnych potrzeb. W modelu zmian należy określić:

* Kolejne etapy przetwarzania zmiany i przejścia między nimi;
* Osoby odpowiedzialne za te etapy i ich obowiązki;
  + Uwagę należy zwrócić na autoryzację zmiany, która nie zawsze musi być wykonywana przez Radę ds. Zmian.
  + Zasady działania CAB lub ECAB, częstotliwość i formę zgromadzeń, ich porządek i skład (mogą być w postaci oddzielnego dokumentu).
* Ograniczenia czasowe i procedury eskalacji.

### Plan zarządzania zmianą

Cele

* Wyeliminowanie awarii spowodowanych zmianami elementów konfiguracji.
* Zredukowanie do zera ilości zmian nieautoryzowanych.
* Zmniejszenie o 80% ilości zmian niespełniających oczekiwań.
* Zmniejszenie o 50% średniego czasu potrzebnego do implementacji zmiany.
* Zrealizowanie powyższych celów w ciągu sześciu miesięcy.

Zasady

1. Niniejszy proces zarządzania zmianą musi być zgodny z zaleceniami ITIL v3.
2. Narzędziem użytym do wsparcia procesu i przechowywania danych będzie system Service Desk opracowany przez Piotra Kalańskiego i Adriana Wiśniewskiego.
3. Format wniosków jest zgodny z odpowiednim formularzem ww. systemu.
4. Wszystkie zmiany, z wyłączeniem standardowych, muszą być zgłaszane jako wnioski.
   1. Zmiany standardowe są obsługiwane przez proces realizacji wniosków.
5. Wnioski mogą być składane przez dowolnego pracownika lub klienta.
6. Podczas składania wniosku należy wybrać odpowiednią kategorię tematyczną.
   1. Kategorie są ustalane przez właściciela procesu.
   2. Klienci widzą jedynie specjalnie dla nich wydzielone kategorie.
7. Podczas składania wniosku należy wybrać odpowiedni priorytet służący do określenia wymagań czasowych.
   1. Priorytet może być niski, normalny, wysoki lub bardzo wysoki.
   2. Klienci nie mogą wybierać priorytetu. Automatycznie ustawia się normalny.
8. Wnioski przetwarzają zarządzający zmianą mianowani przez właściciela procesu.
   1. Zarządzający są przypisani do konkretnej kategorii, którą obsługują.
   2. W ramach każdej kategorii mianowany jest jeden główny zarządzający zmianą, nadrzędny w stosunku do pozostałych.
9. Zarządzający zmianą przypisują do siebie zgłoszone wnioski z obsługiwanej kategorii.
   1. Jeżeli zmiana nie zostanie przypisana w 2d/4g/1g/15m[[62]](#footnote-62) od zgłoszenia, to główny zarządzający zmianą niezwłocznie musi ją przypisać wybranemu zarządzającemu zmianą.
10. Po przypisaniu zarządzający przegląda zmianę w celu określenia jej typu.
    1. Jeżeli typ zmiany nie zostanie określony w 2d/4g/1g/15m od przypisania, to automatycznie ustawiany jest pilny.
    2. Priorytet bardzo wysoki zwykle implikuje zmianę pilną i nie powinien być używany do innych zmian.
11. Istnieją następujące typy zmian:
    1. Pilna – zmiany wymagające natychmiastowej lub jak najszybszej akcji;
    2. Standardowa – zmiany autoryzowane z góry przez właściciela procesu.
    3. Zwykłe – wszystkie inne, nie należące do powyższych kategorii.
12. Zarządzający może podczas przeglądu odrzucić wniosek z podaniem przyczyny. Należy odrzucać wnioski niekompletne, duplikaty i w oczywisty sposób niepraktyczne.
    1. Autorowi przysługuje prawo apelacji. Po poprawieniu wniosku może skierować go do ponownego rozpatrzenia przez tego samego zarządzającego.
13. Zarządzający może podczas przeglądu zmienić kategorię i priorytet wniosku.
    1. W przypadku zmiany kategorii, zarządzający przestaje prowadzić wniosek.
14. Dokładny przebieg całego procesu od momentu nadania typu wnioskowi, wraz z rolami i ich obowiązkami, reguluje odpowiedni, zatwierdzony model zmiany.
15. Ocena sprawności procesu będzie opierała się o stopień wykonania celów, a ponadto:
    1. Ilość zmian wykonanych w ramach dozwolonego czasu.
    2. Ilość przekroczonych poziomów eskalacji.

### Model zmian zwykłych

1. Wniosek po przypisaniu typu zwykłego musi zostać oceniony przez zarządzającego zmianą zgodnie z zasadą 7R opisaną przez ITIL w ciągu 5d/2d/4g/1h.
   1. Zarządzający zmianą może zwrócić się o pomoc w ocenie do Rady ds. Zmian.
   2. Po upłynięciu czasu typ zmiany zmienia się na pilny.
2. Po ocenieniu wniosek jest przesyłany do organu autoryzującego, który zatwierdza wniosek. Wybór organu zależy od wpływu wniosku na przedsiębiorstwo:
   1. Wnioski o małym wpływie autoryzuje zarządzający zmianą od ręki;
   2. O średnim wpływie autoryzuje Rada ds. Zmian na następnym posiedzeniu;
   3. O dużym wpływie autoryzuje zarząd firmy.
3. Wpływ wniosku należy określać w oparciu o dotychczas rozpatrzone wnioski.
   1. Gdy organ autoryzujący stwierdzi, że nie ma wystarczających uprawnień, wniosek zostaje przesłany do wyższej instancji.
4. Wszystkie autoryzowane wnioski są przesyłane do Rady ds. Zmian, której zadaniem jest harmonogramowanie zmian i przydzielanie zasobów do ich wykonania. W szczególności rada musi określić budowniczego zmiany.
   1. Każda kategoria wniosków posiada własną radę, a każda rada może obsługiwać więcej niż jedną kategorię.
   2. Szczegółowe zasady funkcjonowania i skład rady określa właściciel procesu.
5. Budowniczy zmiany odpowiada za zaimplementowanie i wdrożenie zmiany.
   1. Przed przystąpieniem do prac musi opracować plan wycofania, który określa kroki niezbędne do przywrócenia stanu sprzed wdrożenia zmiany w przypadku wystąpienia błędów.
6. Po wdrożeniu zmiany i odczekaniu czasu określonego przez radę, zarządzający zmianą musi sporządzić przegląd powdrożeniowy i zamknąć zmianę. Format przeglądu określa właściciel procesu.

### Model zmian pilnych

1. Model zmian pilnych jest podobny do modelu zmian zwykłych, z wyjątkami:
   1. Zarządzający zmianą musi jak najszybciej ocenić zmianę.
   2. Organem autoryzującym jest Nadzwyczajna Rada ds. Zmian, formowana przez właściciela procesu. Musi ona autoryzować zmianę jak najszybciej. Jej członkowie muszą być dyspozycyjni także poza godzinami pracy.

### Model zmian standardowych

1. Obsługa zmian standardowych nie leży w zakresie procesu zarządzania zmianą. Po napotkaniu zmiany takiego typu, powinien ją odrzucić i w wyjaśnieniu pokierować zgłaszającego do odpowiedniego formularza.

### Kategorie zmian

* Zgłaszane przez klienta:
  + Zmiana poziomu usługi;
  + Instalacja dodatkowego oprogramowania.
* Zgłaszane przez pracowników związane z usługami:
  + Hosting kolokacyjny;
  + Hosting dedykowany;
  + Wirtualny hosting dedykowany;
  + Hosting współdzielony;
  + Współdzielony hosting stron internetowych;
  + Darmowy hosting stron internetowych;
  + Hosting pośredni.
* Zgłaszane przez pracowników związane ze wspólną infrastrukturą:
  + Serwerownia 1;
  + Serwerownia 2;
  + Serwerownia 3;
  + Serwerownia 4.

# System Service Desk

Aby skutecznie wdrożyć ITIL w przedsiębiorstwie, niezbędny jest system informatyczny wspierający tą metodykę. Dzięki automatyzacji rutynowych czynności, procesy mogą być wykonywane znacznie szybciej i efektywniej, co pozwala zaoszczędzić zarówno czas jak i pieniądze. Aplikacja ta pomaga podejmować decyzje, poprzez gromadzenie i udostępnianie kluczowych danych. Przeprowadza użytkowników przez kolejne etapy procesów i koordynuje ich prace. Powiadamia o różnego rodzaju zdarzeniach np. ukończeniu prac lub przekroczeniu terminu. A ponadto kontroluje dostęp do informacji i autoryzuje wykonywane operacje.

Niniejszy rozdział opisuje ogólną architekturę oraz rdzeń systemu, udostępniający podstawową funkcjonalność. Kolejne dwa rozdziały zajmują się szczegółowym omówieniem modułów odpowiedzialnych za zarządzanie konfiguracją i zarządzanie zmianą.

## Wymagania

Przedstawiony zbiór wymagań stanowi podstawę do wykonania oprogramowania i określa jego zakres. Często pojawiający się zwrot „zarządzanie danymi” należy rozumieć jako możliwość wykonywania operacji CRUD[[63]](#footnote-63). Do weryfikacji wszystkich wymagań ilościowych i jakościowych stawianych systemowi służyć będzie serwer wyposażonego w:

* Czterordzeniowy procesor Intel Xeon X3330 2,66 GHz;
* 16 GB pamięci RAM DDR3 1333 MHz ECC;
* Macierz dyskową RAID 5 o pojemności 8 TB;
* Inne elementy sprzętowe niezbędne do działania serwera;
* Oraz komplet wymaganego oprogramowania.

### Technologia (TECH)

Wymagania technologiczne narzucają warunki na wybór stosowanych narzędzi, technik i języków programowania. Ograniczenia te kierowane są potrzebą stworzenie niezawodnego systemu wysokiej jakości, łatwego w obsłudze i taniego w utrzymaniu.

Spis wymagań niefunkcjonalnych

1. Zastosowanie sprawdzonych i powszechnie znanych technik programowania, aby zmniejszyć koszt i ułatwić utrzymanie systemu: paradygmatu obiektowego wraz ze wzorcami programowania opisanymi w [23] i [24].
2. Umożliwienie dostosowania do specyficznych potrzeb klienta poprzez użycie paradygmatu programowania komponentowego, który pozwala na wymianę nawet najmniejszych elementów systemu.
3. Nacisk na enkapsulację poprzez wyodrębnienie funkcjonalności pojawiających się w różnych warstwach systemu (zarządzania transakcjami, autoryzacji i innych), dzięki wykorzystaniu paradygmatu programowania aspektowego.
4. System musi być napisany w języku o statycznym typowaniu, co pozwoli na wczesne wykrycie wielu błędów już na etapie kompilacji.
5. Ilość błędów podczas pracy systemu musi być minimalizowana, poprzez wykorzystanie testów jednostkowych.
6. System do przechowywania informacji musi wykorzystywać relacyjne bazy danych.

### Architektura (ARCH)

Wymagania dotyczące architektury określają zgrubny kształt systemu oraz jego głównych elementów. Zarys ten pozwala wcześnie zdekomponować system na mniejsze elementy i rozdzielić ich funkcjonalność, a w efekcie upraszcza proces projektowania i implementacji.

Spis wymagań niefunkcjonalnych

1. Wykonanie systemu w postaci trójwarstwowej aplikacji sieciowej z dostępem zdalnym w celu zminimalizowania kosztów instalacji, wdrożenia i uaktualniania systemu.
2. Wsparcie wielu różnych warstw prezentacji.
   1. Podstawową warstwę prezentacji stanowi witryna internetowa dostępna przez zwykłą przeglądarkę internetową. Wsparcie dla wszystkich przeglądarek dostępnych na rynku nie jest potrzebne, wystarczy obsługa przeglądarki Firefox w wersji 5 lub wyższej.
   2. Drugą stanowią usługi sieciowe[[64]](#footnote-64), umożliwiające komunikację systemu z innymi aplikacjami i systemami.
   3. Musi istnieć możliwość późniejszego zaimplementowania innych warstw prezentacji takich jak np. interfejs okienkowy.
3. System musi być zbudowany modularnie i być rozszerzalny.
   1. Rdzeń systemu będzie udostępniał usługi nie związane bezpośrednio z procesami ITIL, lecz niezbędne do ich wykonania. Na przykład: autoryzacja, dostarczanie danych o pracownikach firmy, wysyłanie powiadomień do użytkowników itp.
   2. Z funkcjonalności zapewnianej przez rdzeń będą korzystać moduły wspierające procesy ITIL.
      1. Zaimplementowane zostaną moduły wspierające procesy zarządzania konfiguracją i zmianą
      2. Procesy te muszą współpracować z zaimplementowanymi przez Piotra Kalańskiego: zarządzaniem incydentami i problemami.
      3. Musi istnieć możliwość późniejszej implementacji kolejnych modułów wspierających następne procesów ITIL.
   3. Pożądana jest możliwość rozszerzenia systemu o moduły zapewniające funkcjonalność SCM[[65]](#footnote-65), CRM[[66]](#footnote-66) i ERP[[67]](#footnote-67).

### Rdzeń systemu (CORE)

Zbiór modułów świadczących szeroki wachlarz usług, nie związany bezpośrednio z procesami ITIL, ale uzupełniający je o niezbędną z punktu widzenia użytkownika funkcjonalność.

Spis wymagań funkcjonalnych

1. Uwierzytelnienie tożsamości i logowanie użytkowników do systemu.
   1. Użytkownicy mogą być pracownikami, klientami lub kooperantami.
   2. Uwierzytelnianie bazuje na nazwie użytkownika i haśle.
   3. Każdy użytkownik posiada przypisany zestaw uprawnień, określający dane do których użytkownik ma dostęp i operacje jakie może wykonać.
   4. Bez odpowiedniej autoryzacji nie można korzystać z systemu.
   5. Po zakończonej pracy użytkownicy mogą się wylogować.
2. Prezentowanie ważnych dla użytkownika informacji na jego stronie głównej widocznej zaraz po zalogowaniu. Strona ta pełni funkcję deski rozdzielczej (ang. dashboard), prezentującej najważniejsze zdarzenia i zmiany oraz pozwalającej na szybkie reagowanie.
3. Zapewnienie intuicyjnego mechanizmu nawigacji, umożliwiającego dostęp do funkcjonalności wszystkich modułów. Użytkownik powinien szybko i bez trudu znaleźć wszystkie potrzebne mu podstrony.
4. Udostępnienie narzędzi komunikacji z użytkownikami
   1. Wysyłanie ogłoszeń do wszystkich użytkowników. Ogłoszenia zawierają ważne informacje, skierowane do wszystkich użytkowników.
   2. Wysyłanie użytkownikom automatycznych powiadomień o przebiegu wykonywanych przez nich prac. Na przykład informacje o zmianie stanu wniosku umożliwiającej dalsze jego przetwarzanie itp.
      1. Powiadomienia po ich wybraniu przenoszą użytkownika do odpowiedniego formularza pozwalającego mu na kontynuację pracy nad obiektem, który dane powiadomienie wygenerował.
5. Udostępnianie informacji na temat firmy.
   1. Danych pracowników.
   2. Danych klientów.
   3. Danych kooperantów.
   4. Należy mieć na uwadze to, że dane te mogą pochodzić z różnych źródeł, w tym istniejących baz danych lub innych systemów informatycznych.

Spis wymagań niefunkcjonalnych

1. System musi zapewnić miejsce do przechowywania danych o co najmniej dziesięciu tysiącach użytkowników.
2. System musi mieć możliwość przechowywania łącznie co najmniej dziesięciu tysięcy ogłoszeń.
3. Ogłoszenia mogą do systemu dodawać jedynie uprawnieni administratorzy,
4. System musi mieć możliwość przechowywania łącznie co najmniej dziesięciu milionów powiadomień.

## Wykorzystane narzędzia

Java 2 Enterprise Edition

Istnieje bardzo wiele różnych języków programowania. Jednakże do napisania dużych systemów biznesowych nadaje się zaledwie kilka. Przy wyborze tego kluczowego narzędzia z góry zostały odrzucone wszystkie, obecnie bardzo popularne, języki typowane dynamicznie takie jak PHP, Python, czy Ruby. Ich główna zaleta jest zarazem ich główną wadą. Dzięki braku kontroli nad typami zmiennych, umożliwiają one szybkie wytwarzanie oprogramowania, jednak kosztem znacznie obniżonej kontroli, co przekłada się na większą ilość błędów czasu wykonania. Języki o statycznym typowaniu potrafią wychwycić te błędy już na etapie kompilacji, albo nawet wcześniej – podczas pisania kodu, dzięki podpowiedziom środowiska programistycznego. Jednym z głównych wymagań implementowanego systemu jest jego niezawodność, stąd ta radykalna decyzja. Z języków statycznie typowanych tylko dwa posiadają wystarczająco rozwinięte biblioteki do pisania aplikacji sieciowych. Są to Java i C#, przy czym ten drugi ewidentnie zostaje w tyle. Java posiada wiele nowoczesnych i rozwiniętych narzędzi oraz bibliotek pozwalających tworzyć bardzo duże i bezpieczne systemy. Ponadto posiada imponujące wsparcie ze strony środowisk programistycznych, w tym sprawdzanie poprawności kodu podczas pisania i możliwość przeprowadzania zaawansowanych refaktoryzacji.

Spring Framework

Biblioteka Javy do pisania aplikacji sieciowych, która opakowuje mechanizm servletów, udostępniając interfejs programistyczny wysokiego poziomu, pozwalający na szybkie pisanie aplikacji sieciowych. Jej najważniejszym elementem, intensywnie wykorzystywanym podczas implementacji systemu, jest kontener odwróconego sterowania[[68]](#footnote-68), który wspomaga paradygmat programowania komponentowego. Pozwala on na deklaratywne określenie zależności między komponentami aplikacji (zarówno przez piliki xml, jak i adnotacje), a w efekcie umożliwia wymianę nawet najmniejszych części systemu, gdy zajdzie taka potrzeba. Biblioteka posiada szereg modułów m.in. dostępu do baz danych, zarządzania transakcjami, kontroli bezpieczeństwa, wsparcia dla usług sieciowych.

Hibernate

Jest to biblioteka do mapowania obiektowo-relacyjnego[[69]](#footnote-69). Stanowi warstwę łączącą obiektowy świat Javy z relacyjnymi bazami danych. Automatycznie tworzy i wykonuje zapytania SQL podczas dostępu do właściwości obiektów, co pozwala programiście zaoszczędzić czas, który musiałby przeznaczyć na pisanie tych zapytań.

Maven

Jest to system automatycznego budowania aplikacji w języku Java, podobnie jak popularny make dla C i C++. Narzędzie to jest jednak znacznie bardziej rozwinięte. Do głównych jego zalet należy zarządzanie zależnościami projektów. Po określeniu wymaganych przez aplikację bibliotek, narzędzie jest w stanie samo je wszystkie odszukać i pobrać, łącznie w wszystkimi zależnościami drugiego i dalszych rzędów. Wszystkie zależności wyświetlane są w postaci grafu, dzięki czemu można bardzo łatwo rozwiązywać ewentualne konflikty wersji. Drugą ogromną zaletą narzędzia jest system dodatków umożliwiający dołożenie do procesu budowania aplikacji kroków takich jak uruchomienie testów, generowanie dokumentacji i raportów, czy automatyczna kompresja i zaciemnianie skryptów języka Javascript czy arkuszy stylów Css. Narzędzie pozwala ponadto automatycznie pakować aplikacje Javy do archiwów war i wdrażać je na docelowy serwer.

Mercurial

Repozytorium kodu umożliwiające równoczesną pracę wielu osobom. Posiada wiele usprawnień w stosunku do systemów CVS i SVN.

Oracle

Wykorzystywana przez implementowany system baza danych. Aplikacja korzysta z pewnych zaawansowanych możliwości Oracle, takich jak np. klastry, jednak są one używane jedynie w celu poprawienia wydajności i nie są niezbędne do uruchomienia aplikacji. Dlatego w razie potrzeby może zostać wymieniona na dowolną inną np. PostgreSQL czy MySql. Wystarczy zmienić konfigurację Hibernate.

Tomcat

Serwer www, implementujący mechanizm servletów Javy.

IBM Rational Software Architect

Narzędzie wykorzystane do projektowania systemu i tworzenia diagramów UML.

JDeveloper i Oracle SQL Developer Data Modeler

Narzędzia wykorzystane do projektowania schematów bazy danych.

Netbeans i Spring Source Tool Suite

Środowiska programistyczne, w których powstał implementowany system. Spring Source Tool Suite posiada szereg ciekawych narzędzi poświęconych specjalnie bibliotece Spring, jednak jest bardzo niestabilnym narzędziem, dlatego wykorzystano także Netbeans.

## Ogólny projekt architektury

System posiada jednolitą architekturę, która jest narzucana wszystkim dołączanym modułom. Pozwala to na utrzymanie lekkich zależności i czystych interfejsów. Opiera się ona na klasycznej architekturze trójwarstwowej. W systemie wyróżniamy warstwę danych, która odpowiada za przechowywanie i udostępnianie wszystkich niezbędnych informacji. Stanowi ją oprogramowanie relacyjnej bazy danych z interfejsem w postaci połączenia JDBC[[70]](#footnote-70), po którym przesyłane są zapytania w języku SQL i odbierane ich wyniki.

Drugą warstwę stanowi logika aplikacyjna, przeprowadzająca wszystkie procesy biznesowe. Stanowi ona serce systemu, które przetwarza wszystkie żądania użytkowników. Warstwa ta udostępnia interfejs składający się ze zbioru dziesiątek usług, pozwalających pobierać i zmieniać dane przechowywane w bazie. Logika aplikacyjna kontroluje dostęp do informacji, autoryzuje wykonywane operacje i odpowiada za zachowanie reguł biznesowych.

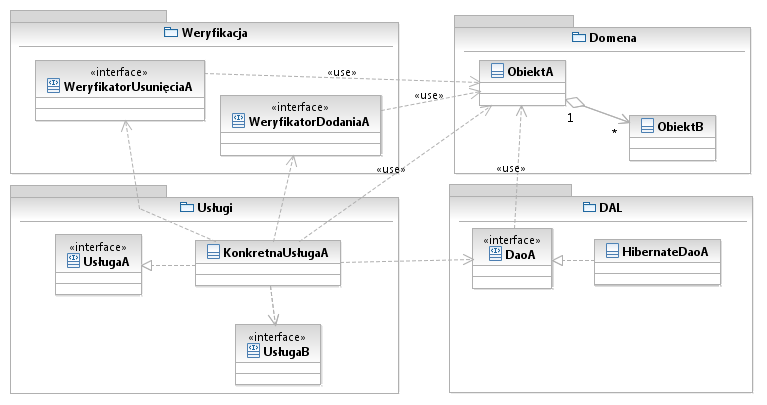


Ilustracja . Warstwy systemu

Warstwa prezentacji korzysta z usług świadczonych przez logikę aplikacyjną i udostępnia przyjazny interfejs dla użytkownika. Implementowany system posiada aktualnie dwie niezależne warstwy prezentacji. Główną jest witryna internetowa dostępna dla użytkowników, składająca się ze zbioru stron i formularzy umożliwiających interakcję z warstwą logiki aplikacyjnej. Natomiast drugą stanowią usługi sieciowe, które umożliwiają integrację systemu z innymi aplikacjami.

### Warstwa logiki aplikacji

Warstwa logiki aplikacji dzieli się na cztery główne podwarstwy. Pierwsza zawiera obiekty reprezentujące encje domeny, modelujące przedmioty świata rzeczywistego i przechowujące związane z nimi dane. Podwarstwa ta jest niezależna od pozostałych. Encje domeny pełnią rolę nośników informacji i są wykorzystywane oraz przetwarzane przez wszystkie części systemu. Mogą zawierać także elementy logiki biznesowej nierozerwalnie związane z domeną.



Ilustracja . Schemat podziału warstwy aplikacji na podwarstwy

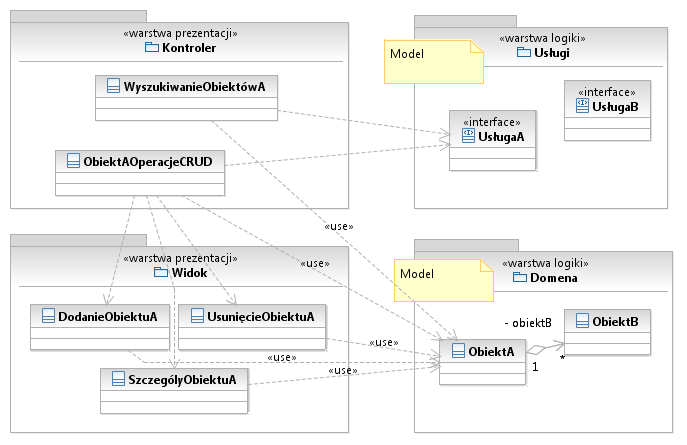
Drugą podwarstwą jest tak zwana warstwa dostępu do danych (ang. Data Access Layer – DAL), która odpowiada za składowanie obiektów domenowych w trwałym magazynie. Podwarstwa zawiera obiekty dostępu do danych (ang. Data Access Object – DAO), których jest tyle ile typów encji domenowych. Obiekty DAO potrafią pobierać, uaktualniać i zapisywać dane encji domenowych powiązanego typu. Warstwa ta abstrahuje sposób przechowywania danych i sprawia, że system jest od niego niezależny. W szczególności oznacza, że dane mogą być zapisywane w magazynie innym niż relacyjna baza danych. Jest to pomocne podczas testowania, ponieważ można wtedy użyć komponentów czytających i zapisujących dane w pamięci, co znacznie upraszcza całą procedurę. W przypadku implementowanego systemu obiekty DAO obudowują wywołania biblioteki Hibernate.

Kolejną podwarstwą jest podwarstwa weryfikacji reguł biznesowych zawierająca obiekty – weryfikatory sprawdzające niezmienniki reguł biznesowych. Obiekty te analizują stan encji domenowych i orzekają, czy można dla nich wykonać żądane operacje np. dodania bądź usunięcia. W przypadku decyzji negatywnej mogą przekazać informację wyjaśniającą dlaczego operacja została odrzucona. Reguły biznesowe mogą się czasami zmieniać, dlatego bardzo ważne jest wyodrębnienie ich do oddzielnych, łatwo modyfikowalnych obiektów.

Ostatnia podwarstwa nazywana jest warstwą usług. Zawiera ona komponenty wykonujące konkretne procesy biznesowe. Każdy komponent odpowiada grupie powiązanej funkcjonalności, a każda jego metoda realizuje jeden przypadek użycia. Wzorzec ten jest określany jako skrypty transakcyjne [24]. Podwarstwa modyfikuje encje domenowe, wykorzystuje DAL do ich pobierania i zapisywania zmian oraz korzysta z oceny weryfikatorów. Jako jedyna jest widoczna przez warstwy prezentacji, poprzez szereg udostępnianych interfejsów.

### Warstwa prezentacji: witryna internetowa

Warstwa prezentacji wykonana jest z użyciem wzorca Model-Widok-Kontroler sieciowego[[71]](#footnote-71) (ang. Web MVC[[72]](#footnote-72)). Jej zadaniem jest udostępnienie ergonomicznego interfejsu użytkownika, który umożliwia wykonywanie operacji logiki biznesowej. Pierwsza litera w skrócie MVC oznacza model, czyli ogół przetwarzanych danych i operacji na nich wykonywanych. W przypadku implementowanego systemu model stanowi warstwa usług wraz z encjami domenowymi na których ona operuje. Należy zwrócić uwagę, że warstwa prezentacji widzi jedynie interfejsy usług znajdujących się w warstwie logiki biznesowej, co pozwala wymieniać implementacje usług bez potrzeby zmiany warstwy prezentacji (oczywiście przy zachowaniu interfejsu). Zastanawiający może być fakt, że warstwa prezentacji korzysta bezpośrednio z encji domenowych, znajdujących się głęboko w warstwie logiki aplikacji. Działanie to jest zamierzone i uznawane za dobrą praktykę. Jedyną realną alternatywą jest przekazywanie przez warstwę usług obiektów transferu danych (ang. Data Transfer Object – DTO). Są to bardziej struktury niż klasy, które stanowią odbicie encji domenowych i zawierają część ich pól. Klas obiektów DTO powstaje bardzo dużo, czasami jedna na każdy przypadek użycia, a redundancja powoduje kaskadowe modyfikacje, gdy zmieni się jedno pole encji domenowej. Z tych powodów obiekty DTO zaczęto postrzegać jako antywzorzec.



Ilustracja . Schemat budowy warstwy prezentacji

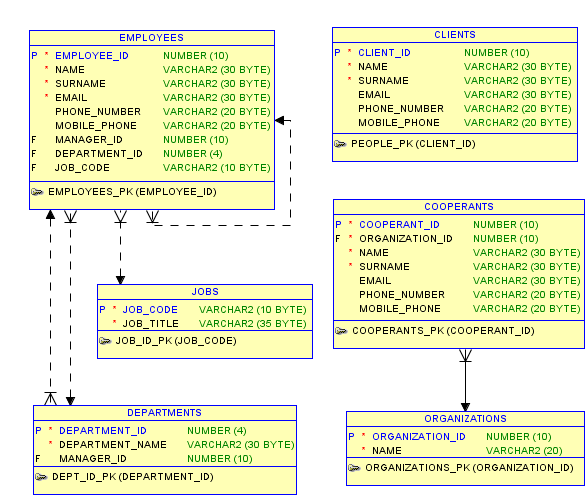
Drugim elementem wzorca MVC jest widok, który jest odpowiedzialny za bezpośrednią obsługę interfejsu użytkownika. W przypadku aplikacji sieciowych jest to przygotowanie kodu html odwiedzanej strony i wiązanie nadesłanych w formularzach danych z obiektami przetwarzanych encji. Ostatnim elementem wzorca jest kontroler, który odpowiada na żądania użytkownika, uaktualnia model i wiąże go z odpowiednim widokiem. Częstym błędem popełnianym przy pisaniu aplikacji sieciowych jest umieszczanie logiki biznesowej wewnątrz kontrolerów. Takie postępowanie dosłownie cementuje zależność między warstwą prezentacji i logiki aplikacyjnej. Rolą kontrolera jest jedynie koordynacja podległych mu elementów.

## Projekt rdzenia systemu

Opis projektu rdzenia systemu jest ograniczony do elementów, które zostały zmienione lub dodane przez autora pracy. Informacje dotyczące pozostałych elementów znajdują się w pracy dyplomowej Piotra Kalańskiego [1].

### Schemat bazy danych

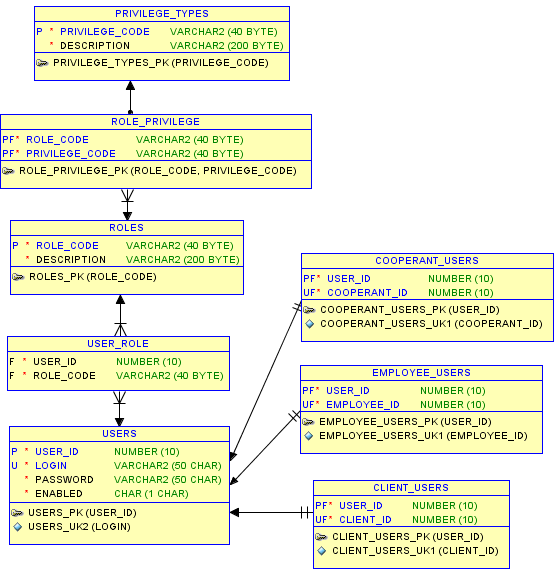
Dane firmy



Ilustracja . Tabele związane z danymi firmy

Dane firmy sprowadzają się obecnie do przechowywania informacji o klientach, kooperantach, powiązanych organizacjach i pracownikach. Pracownicy należą do departamentów i są na określonym stanowisku. Wszystkie wymienione encje posiadają odpowiadające im tabele i nie ma w nich nic nadzwyczajnego.

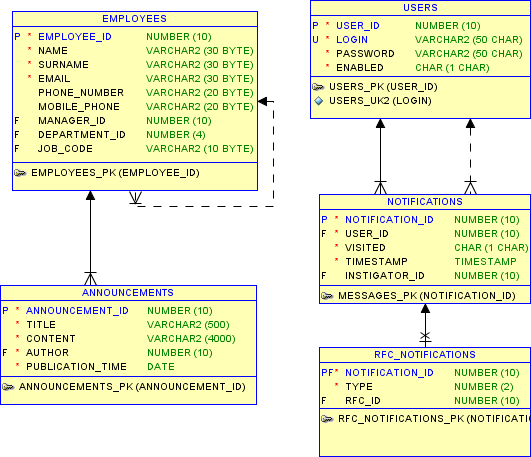
Uwierzytelnianie i autoryzacja użytkowników



Ilustracja . Tabele związane z autoryzacją użytkowników i danymi firmy

Autoryzacja w aplikacji opiera się o system uprawnień. aby wykonać daną operację, użytkownik musi posiadać określone uprawnienie. Uprawnienia potrzebne do wykonania powiązanych ze sobą czynności grupowane są w role. Ułatwia to zarządzanie autoryzacją, ponieważ uprawnienia ról w odróżnieniu od uprawnień użytkowników zmieniają się rzadko. Role są następnie przypisywane użytkownikom. Każdy użytkownik może pełnić wiele ról naraz i posiada wszystkie uprawnienia pełnionych ról. Informacje potrzebne do uwierzytelnienia użytkownika znajdują się w tabeli USERS, która modeluje abstrakcyjną encję użytkownika. Istnieją trzy konkretne typy użytkowników: klienci, kooperanci i pracownicy. Użytkownik może zostać zablokowany przez administratora, co uniemożliwia jego uwierzytelnienie.

Ogłoszenia i powiadomienia



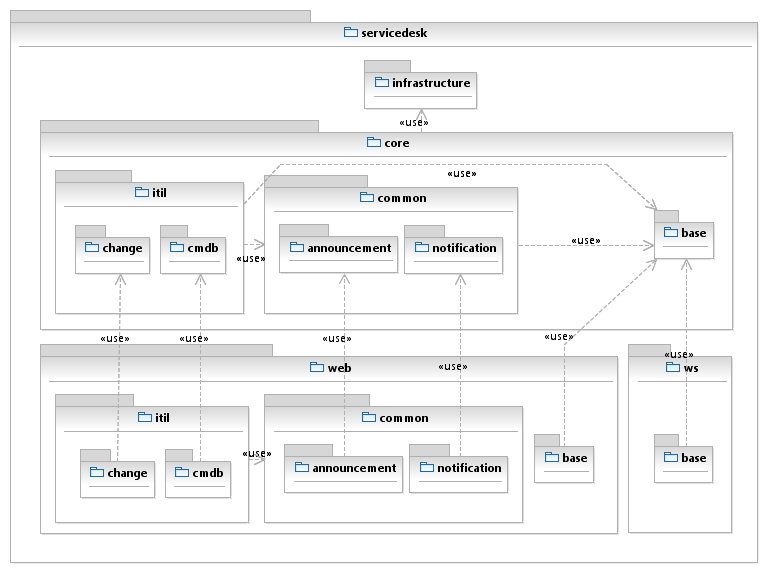
Ilustracja . Tabele związane z ogłoszeniami i powiadomieniami

Ogłoszenia mogą być tworzone jedynie przez uprawnionych pracowników (nie mogą tego robić klienci i kooperanci, którzy także mają dostęp do systemu). Posiadają krótki tytuł i treść. Można ustawić data publikacji w przyszłość, co spowoduje ukrycie ogłoszenia i jego opublikowanie w wyznaczonym czasie.

Powiadomienia są tworzone automatycznie przez system. Każde powiadomienie należy do pojedynczego użytkownika. Posiada odnotowany czas zgłoszenia, flagę określającą, czy zostało już odwiedzone oraz opcjonalnie identyfikator użytkownika, który je spowodował (niektóre powiadomienia, np. o przekroczeniu czasu nie posiadają inicjatora). Encja powiadomienie jest abstrakcyjna. Każdy moduł może stworzyć własny podtyp powiadomień i przechowywać w nim wymagane informacje i dowolnie nimi zarządzać (np. usuwać powiadomienia nieaktualne). Moduł musi także udostępnić komponent, który posłuży do przetłumaczenia pól powiadomienia na tekst wyświetlany użytkownikowi (przechowywanie powtarzających się ciągów byłoby skrajnie nieefektywne) i przekierowujący użytkownika do odpowiedniego formularza lub okna w zależności od warstwy prezentacji.

## Implementacja rdzenia systemu

### Pakiety



Ilustracja . Diagram pakietów systemu Service Desk

Wszystkie pakiety aplikacji zaczynają się od jednolitego rdzenia **servicedesk**, który stanowi wspólną przestrzeń nazw. Głównymi pakietami systemu są:

* **infrastructure** – zawiera klasy biblioteczne, ogólnego przeznaczenia.
* **core** – warstwa logiki aplikacji
  + **itil** – usługi związane z procesami metodyki
    - **change** – implementacja wsparcia procesu zarządzania zmianą
    - **cmdb** – implementacja wsparcia procesu zarządzania konfiguracją
  + **common** – usługi rdzenia systemu
  + **base** – wewnętrzne usługi nie stanowiące funkcjonalności systemu
* **web** – witryna internetowa (warstwa prezentacji)
* **ws** – usługi sieciowe (warstwa prezentacji)

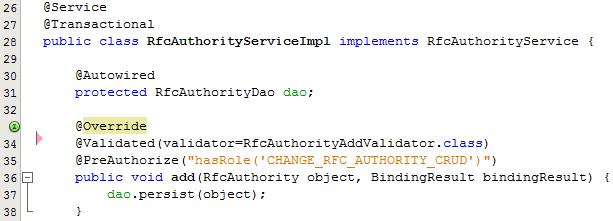
### Kontener i komponenty

System został zaimplementowany w duchu programowania komponentowego. Podstawową jednostką systemu jest komponent. Funkcjonalność komponentu musi spełniać kontrakt zapisany w implementowanym interfejsie (lub ich zbiorze). Komponent może komunikować się z innymi komponentami jedynie za pomocą interfejsów. Zasady te stwarzają sytuację, w której żaden komponent nie zależy bezpośrednio od innego komponentu, a jedynie od deklarowanych interfejsów. Dzięki temu można w dowolnej chwili wymienić każdy komponent na inny implementujący ten sam interfejs. Oznacza to, że w razie potrzeby można wymienić nawet najmniejszy element systemu, dostosowując działanie do własnych potrzeb.

Kontener jest tworem, który tworzy komponenty i zarządza ich zależnościami. Technik dostarczenia komponentom referencji do konkretnych implementacji w miejsce używanych interfejsów jest kilka. Kontener Spring używa wstrzykiwania zależności[[73]](#footnote-73). Za pomocą refleksji automatycznie ustawia odpowiednie pola klas podczas ich tworzenia. Konfiguracja kontenera jest w pełni deklaratywna, opis zależności można podać w postaci plików xml i adnotacji bezpośrednio w kodzie.

### Aspekty

Wykonanie funkcjonalności systemu pociąga za sobą potrzebę realizacji pewnych pobocznych zagadnień. Na przykład przed wykonaniem pewnych operacji należy autoryzować użytkownika. Jest to czynność wykonywana rutynowo. Pojawia się w wielu miejscach, często przecina kod logiki biznesowej, zaciemniając go i tworząc wiele zbędnych zależności. Z tego powodu podczas implementacji systemu skorzystano z paradygmatu programowania aspektowego [25], który potrafi rozdzielić przecinające się zagadnienia w sposób deklaratywny, zwiększając modularyzację i czystość kodu. Poniższy listing przedstawia jednolinijkową metodę, która realizuje przypadek użycia zapisania obiektu w bazie danych. Przed wywołaniem aspekty autoryzują użytkownika, rozpoczynają transakcję, sprawdzają poprawność przekazywanego obiektu. A po zatwierdzają transakcję lub w razie wyjątku ją cofają.



Listing . Przykład użycia aspektów do implementacji autoryzacji, weryfikacji i zarządzania transakcjami

Autoryzacja

Do uwierzytelniania użytkowników i autoryzacji system wykorzystuje bibliotekę Spring Security. Uwierzytelnianie następuje w warstwie prezentacji za pomocą formularza logowania. Dostęp do jakiejkolwiek strony witryny systemu wymaga uwierzytelnienia. Autoryzacja odbywa się w warstwie logiki aplikacji przed lub po wywołaniu metod oznaczonych specjalnymi adnotacjami *@PreAuthorize* i *@PostAuthorize* z argumentem określającym kryterium autoryzacji w postaci wyrażeni języka SpEL[[74]](#footnote-74). Wyrażenie może zawierać różne predykaty i odwołania do argumentów metody. Najczęściej używanym predykatem jest „hasRole” sprawdzające, czy użytkownik ma dane uprawnienie. Podczas implementacji systemu rozszerzono bibliotekę o dodatkowe predykaty np. „isOwner” sprawdzający czy użytkownik jest właścicielem obiektu.

Weryfikacja

Obiekty przekazywane przez warstwę prezentacji do warstwy logiki są narażone na niedozwolone manipulacje ze strony użytkownika[[75]](#footnote-75). Użytkownik może dowolnie kształtować odpowiedzi protokołu http, a narzędzia do tego typu czynności są powszechnie dostępne. Dlatego nie można ufać, że dane wysyłane do warstwy logiki aplikacyjnej są rzetelne. Lub też użytkownik może najzwyczajniej w świecie się pomylić czy zapomnieć wpisać wartość do formularza. Dlatego wywołania metod komponentów należących do warstwy usług są obwarowane napisaną przez autora adnotacją *@Validated*, wychwytywaną przez aspekt weryfikacyjny. Implementacja aspektu przekazuje parametr wywołania metody do wskazanego obiektu weryfikatora, który sprawdza warunki reguł biznesowych i w razie ich złamania rzuca wyjątek *BusinessConstraintViolationException* zawierający zbiór nazw złamanych reguł, co umożliwia wyświetlenie w warstwie prezentacji odpowiednich komunikatów informujących użytkownika o przyczynie błędu.

Zarządzanie transakcjami

W aplikacji sieciowej należy określić miejsca w których aplikacja nawiązuje połączenie z bazą danych i zaczyna transakcję. Pojedynczy przypadek użycia może wymagać wielu spójnych dostępów do bazy w ramach jednej transakcji. Nie można tworzyć ich na potrzeby pojedynczych zapytań, ponieważ funkcjonują w szerszym kontekście. Implementowany system na każde żądanie użytkownika tworzy zwykle dwie transakcje. Pierwszą przy wywołaniu metod warstwy usług oznaczonych standardową adnotacją *@Transactional*, umożliwiającą wykonanie zapytań potrzebnych do realizacji przypadku użycia. Druga transakcja jest otwierana podczas przygotowywania widoku. Umożliwia doczytanie potrzebnych danych. Jest to transakcja tylko do odczytu i nie można podczas niej wykonywać zapytań DML[[76]](#footnote-76).

Inny problem powstaje, gdy przypadek użycia rozciąga się na kilka żądań użytkownika. Zapewnienie spójności poprzez trzymanie otwartej sesji podczas czasu namysłu użytkownika jest nieakceptowalne ze względów wydajnościowych, ponieważ powodowałoby długie blokady zasobów. Dlatego transakcje są wykonywane jedynie podczas przetwarzania żądań i zamykane najwcześniej jak to tylko możliwe. Żeby zachować spójność między kilkoma żądaniami użytkownika, każda główna encja[[77]](#footnote-77) posiada pole wersji będące liczbą całkowitą. Za każdym razem gdy dochodzi do modyfikacji encji wersja ta jest zwiększana. Gdy kilkoro użytkowników próbuje naraz wykonać operację na jednym obiekcie, tylko jednemu z nich powiedzie się. Znacznik wersji jest sprawdzany bezpośrednio przed modyfikacją obiektu i jeżeli jest mniejszy niż obecny, transakcja jest wycofywana, a warstwa dostępu do danych rzuca wyjątek *StaleStateException*. W przeciwnym wypadku żądania jednoczesne byłyby przetwarzane kolejno, co powodowałoby nadpisywanie danych bez wiedzy użytkownika.

### Infrastruktura!

# Moduł zarządzania konfiguracją

## Wymagania

Zadaniem modułu jest zarządzanie CMDB, czyli przechowywanie danych konfiguracji firmy zgodnie z procesem zarządzania konfiguracją ITIL w wersji 3. Zawartość CMDB stanowią informacje o elementach konfiguracji, ich atrybutach i relacjach między nimi. Elementy konfiguracji dzielą się na dwie zasadnicze grupy: elementy wbudowane i uniwersalne.

Do pierwszej grupy należą elementy konfiguracji zarządzanie przez inne moduły takie jak zgłoszenia incydentów, problemów, żądania zmian, ale także pracownicy i klienci. Elementy te są wyświetlane w module konfiguracji w trybie do odczytu. Za edycję elementów wbudowanych odpowiedzialne są ich macierzyste moduły, co pozwala na egzekwowanie odpowiednich uprawnień i ograniczeń wewnątrz tych modułów.

Drugą grupę stanowią elementy uniwersalne posiadające dowolny zestaw atrybutów, określany przez użytkownika. Ich zadaniem jest przechowywanie informacji o elementach konfiguracji specyficznych dla danej branży. Przykładami są: monitory, serwery, elementy infrastruktury, komponenty niezbędne do funkcjonowania firmy.

Spis wymagań funkcjonalnych

1. Przechowywanie, wyszukiwanie i prezentacja danych wszystkich elementów konfiguracji (zarówno wbudowanych jak i uniwersalnych).
   1. Element konfiguracji posiada przypisany identyfikator oraz zbiór wartości atrybutów.
   2. Format identyfikatora jest ustalany przez firmę (np. dla monitora LCD zakupionego w roku 2010 roku: HW/MON/LCD/4/30/2010/8).
   3. Dodatkowo element posiada krótkie podsumowanie złożone z wybranych wartości atrybutów, dzięki któremu znacznie łatwiej odnaleźć szukany element na liście (np. dla tego samego monitora: Samsung SME1920N 19" 1360x768 TFT/TN).
2. Zarządzanie uniwersalnymi elementami konfiguracji.
   1. Uniwersalny element konfiguracji posiada dodatkowo typ. Typ podawany jest podczas tworzenia elementu konfiguracji i nie może być zmieniony.
3. Zarządzanie typami elementów uniwersalnych.
   1. Typ elementu uniwersalnego posiada unikalną nazwę, opis oraz zestaw atrybutów.
   2. Typ może być abstrakcyjny, co oznacza, że nie można stworzyć elementu konfiguracji tego typu.
   3. Typy tworzą hierarchię. Podtypy dziedziczą atrybuty typu nadrzędnego.
   4. Kluczowym wymaganiem jest, aby można było dodawać nowe typy z poziomu interfejsu użytkownika, bez potrzeby modyfikacji systemu.
4. Zarządzanie relacjami między elementami konfiguracji.
   1. Relacja łączy dwa elementy konfiguracji (wbudowane lub uniwersalne). Ponadto tak jak w przypadku uniwersalnych elementów konfiguracji posiada typ oraz zbiór wartości atrybutów.
5. Zarządzanie typami relacji. Wszystkie wymagania dotyczące typów uniwersalnych elementów konfiguracji odnoszą się także do typów relacji. A ponadto klasa relacji:
   1. określa czy relacja jest skierowana, czy nie;
   2. wyznacza podpis relacji w obu kierunkach (np. obsługuje, jest obsługiwany przez);
   3. nakłada ograniczenia na typy elementów obu stron relacji.
6. Zarządzanie atrybutami typów elementów uniwersalnych i relacji.
   1. Atrybut posiada nazwę, opis oraz swój format.
      1. W obecnej wersji jedynym dostępnym formatem atrybutu jest napis. W kolejnych wersjach będą dodawane nowe formaty.
7. Zarządzanie wartościami atrybutów elementów uniwersalnych i relacji.
   1. Wartości atrybutów to pary typu atrybut-wartość. Pierwszy element pary musi być atrybutem należącym do typu elementu lub relacji, natomiast drugi musi posiadać format określony przez dany atrybut.

Spis wymagań niefunkcjonalnych

1. Zarządzać elementami konfiguracji i relacjami oraz wartościami ich atrybutów mogą jedynie użytkownicy posiadający rolę zarządzającego konfiguracją.
2. Zarządzać typami elementów konfiguracji i relacji oraz ich atrybutami mogą jedynie użytkownicy posiadający rolę administratora konfiguracji.
3. System musi zapewnić miejsce do przechowywania stu tysięcy elementów konfiguracji. Elementy testowe będą posiadać dwadzieścia wartości atrybutów o średniej długości dwustu bajtów.
4. System musi zapewnić miejsce do przechowywania stu typów elementów uniwersalnych i stu typów relacji. Typy testowe będą posiadać dwadzieścia atrybutów o średniej długości wynoszącej jeden kilobajt, ponadto ich nazwa i opis nie przekroczą dwóch kilobajtów.
5. Przy bazie danych zapełnionej w ilości wskazanej w punktach 3. i 4. system powinien odpowiadać na wszystkie zapytania w czasie poniżej pięciu sekund.

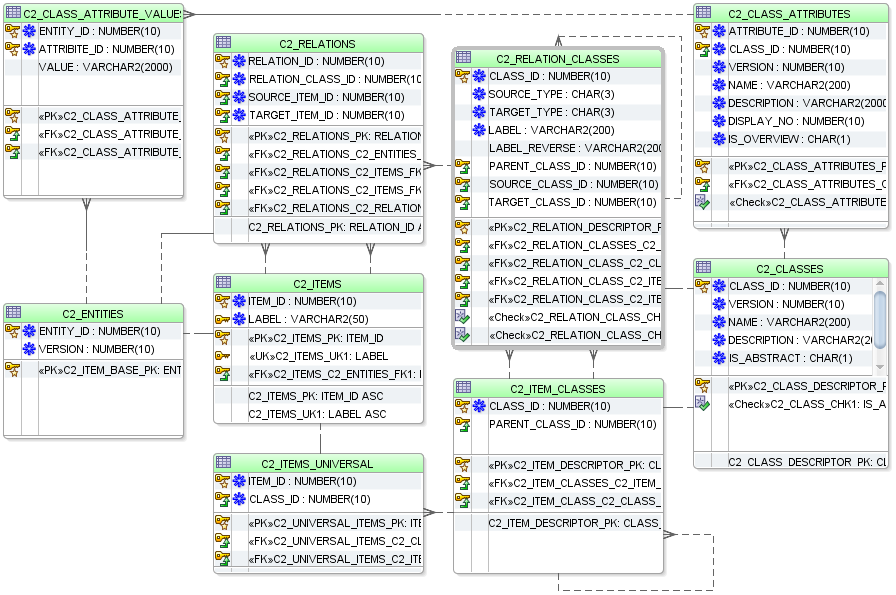
## Projekt

Moduł jest podzielony na trzy warstwy. Posiada własny zbiór tabel w bazie danych. W pakiecie *servicedesk.core.itil* znajduje się implementacja logiki biznesowej, a w pakiecie *servicedesk.web.itil* implementacja warstwy prezentacji modułu. Struktura modułu jest zgodna z opisaną w rozdziale czwartym ogólną architekturą systemu. Moduł nie korzysta z innych modułów ITIL, natomiast jest przez nie wykorzystywany.

### Schemat bazy danych

Uniwersalne elementy konfiguracji

Głównym wymaganiem jest umożliwienie dodawania nowych typów elementów konfiguracji i relacji bez konieczności ingerencji w bazę danych i aplikację. Z tego powodu tradycyjna metodyka projektowania baz danych, która dla każdej modelowanej encji tworzy oddzielną tabelę, nie spełnia wymogów postawionych przed modułem. Informacje o typach elementów muszą być przechowywane na równi z danymi, dlatego że musi istnieć możliwość ich modyfikacji. Te metadane opisują nazwę typu, jego powiązania z innymi typami oraz zbiór atrybutów posiadanych przez obiekty danego typu. Schemat, który pozwala dynamicznie modelować typy encji za pomocą metadanych nazywany jest schematem generycznym lub schematem uniwersalnym [26]. Jego zaletą jest to, że może zawierać dane dowolnych obiektów. Do wad należy znacznie bardziej skomplikowana struktura i wydajność, która może być gorsza w porównaniu z modelami tradycyjnymi. Jednak do rozwiązania problemu przechowywania danych elementów konfiguracji schemat generyczny jest narzędziem wręcz idealnym.



Ilustracja . Tabele przechowujące dane o uniwersalnych elementach konfiguracji

Zaprojektowany schemat intensywnie wykorzystuje modelowanie dziedziczenia z oddzielną tabelą na każdy podtyp [27]. Tabele posiadają pola wersji, których przeznaczenie zostało opisane w podrozdziale 4.5.2. Podczas modelowania metamodelu słowa takie jak typ i klasa, czy atrybut mają podwójne znaczenie, ponieważ modelowane encje domenowe posiadają dokładnie takie nazwy. Z tego powodu w kliku kolejnych akapitach dla czytelności wytłuszczono nazwy encji domenowych. Dla przykładu **klasa** oznacza encję domenową, a nie kryterium klasyfikacji w sensie pierwotnym tego słowa.

Schemat pozwala modelować różne typy zarówno elementów jak i relacji, dlatego oba te twory zostały wyprowadzone z tego samego typu abstrakcyjnego: **encji** (tabela *C2\_ENTITIES*). **Encja** posiada własny sztuczny identyfikator oraz zbiór atrybutów. Konkretyzacją **encji** jest **element** (C2\_ITEMS) lub **relacja** (C2\_RELATION). Każdy **element** posiada etykietę będącą identyfikatorem naturalnym o formacie zależnym od przedsiębiorstwa. **Element** jest także tworem abstrakcyjnym i posiada kilka różnych konkretyzacji, jednak na razie interesuje nas jedna z nich: **element uniwersalny** (C2\_ITEMS\_UNIVERSAL). Każdy **element uniwersalny** posiada przypisaną **klasę elementu** (*C2\_ITEM\_CLASSES*), która determinuje zbiór jego atrybutów. **Relacje** z kolei posiadają przypisaną **klasę relacji** (*C2\_RELATION\_CLASSES*). Rozróżnienie na dwa typy **klas** zostało zaprojektowane po to, aby uniemożliwić przypisywanie **klas** **relacjami** **elementom** i vice versa. **Relacje** ponadto zawierają wskazania na uczestniczące w niej **elementy** źródłowy i docelowy.

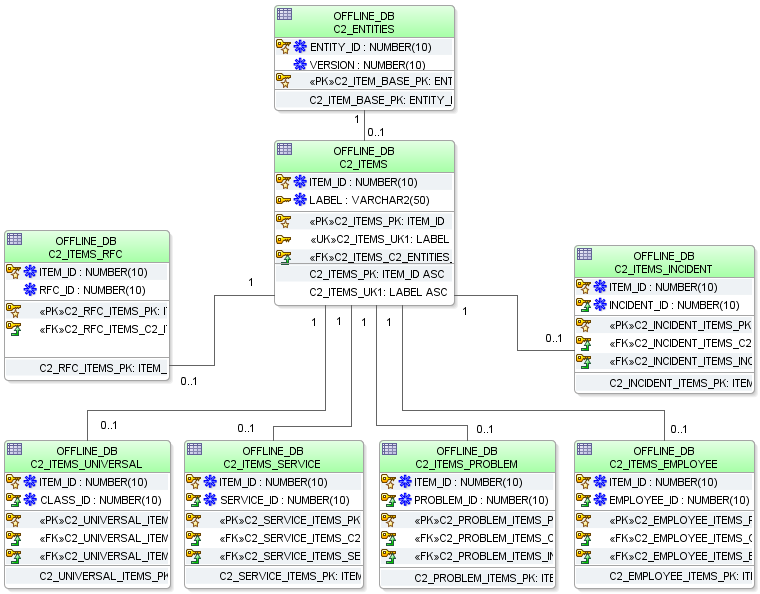
**Klasy elementów** i **klasy** **relacji** dziedziczą z abstrakcyjnego typu **klasy** (*C2\_CLASSES*). Abstrakt ten posiada unikalną nazwę, opis i flagę wskazującą, czy można tworzyć obiekty danej **klasy**. **Klasy** mogą dziedziczyć atrybuty po innych **klasach**, dlatego flaga ta jest bardzo przydatna przy modelowaniu **klas** bazowych. Należy zwrócić uwagę na to, że wskazanie na **klasę** dziedziczoną znajduje się w konkretyzacjach, przez co niemożliwe jest odziedziczenie **klasy relacji** przez **klasę elementu**. **Klasa elementu** nie rozszerza funkcjonalności abstrakcyjnej, natomiast **klasa relacji** pozwala nałożyć ograniczenie na to jakie **elementy** mogą wchodzić w **relację** tej **klasy**. Restrykcja dotyczy zarówno typu **elementu** (na razie przedstawiono typ uniwersalny, inne będą przedstawione później) jak i jego **klasy**. **Klasa relacji** pozwala także definiować etykiety w obu kierunkach relacji. Domyślnie **relacje** są skierowane, co wynika z podziału na element źródłowy i docelowy. Jednak jeśli w **klasie relacji** wartość pola LABEL\_REVERSE nie będzie ustawiona to należy traktować **relacje** tej **klasy** jako symetryczne dwukierunkowe.

**Klasa** agreguje zbiór **atrybutów** (*C2\_CLASS\_ATTRIBUTES*). Każdy **atrybut** oprócz identyfikatora posiada nazwę, opis, numer kolejności wyświetlania i flagę określającą, czy jego wartość jest wyświetlana w podglądzie **elementu**. Intencją podglądu jest umożliwienie użytkownikowi szybkiego, wzrokowego odnalezienia szukanych **elementów** na listach, dlatego powinny do niego wchodzić **atrybuty** o charakterze klucza naturalnego. **Atrybut** musi posiadać unikalną nazwę w ramach wszystkich **atrybutów** **klasy** (wraz odziedziczonymi). Ograniczenie to jest sprawdzane aplikacyjnie, poza bazą danych.

Istotę schematu generycznego stanowi tabela *C2\_CLASS\_ATTIBUTE\_VALUES* przechowująca **wartości atrybutów**. Sprzęga ona **elementy** z **atrybutami klas** do których należą. Każdy **element** może posiadać **wartości atrybutów** związane z **atrybutami** przypisanej mu **klasy** i jej podklas. Ograniczenie to jest także egzekwowane przez aplikację. Co prawda istnieje możliwość zaprojektowania schematu generycznego, który zachowuje spójność z użyciem kluczy obcych, jednak niemożliwe jest wtedy dziedziczenie. **Wartość atrybutu** jest przechowywana w postaci łańcucha znaków o zmiennej długości, jako że format ten jest najbardziej ogólny. Schemat w obecnej wersji nie przewiduje innych typów **wartości atrybutów** poza łańcuchem znaków, jednak ich dodanie nie stanowi problemu. Najprostszym sposobem jest dołączenie do **wartości atrybutu** pole wyliczeniowe określającego typ wartości i odpowiednie szeregowanie (ang. serialization) do i z łańcucha znaków.

Elementy konfiguracji innych modułów

W ten sposób implementowany system przechowuje elementy definiowane przez użytkownika. CMDB musi zawierać także elementy konfiguracji będące produktami innych procesów ITIL takie jak incydenty, problemy czy wnioski o zmianę. Zapisywanie ich w postaci generycznej uniemożliwiło by korzystanie z więzów, wyzwalaczy, indeksów i innych mechanizmów bazodanowych. Działanie systemu w stosunku do elementów uniwersalnych ogranicza się do ich edycji i wyświetlania, dlatego w ich przypadku nie ma potrzeby korzystania z tej funkcjonalności. Inne moduły ITIL mogą jednak wymagać tych mechanizmów, dlatego w trosce o ich bezpieczeństwo i wydajność, podjęto decyzję, że moduły te same będą odpowiedzialne za przechowywanie swoich danych. Moduł konfiguracji nie może w żaden sposób ich ograniczać poprzez narzucenie z góry pewnych założeń. Tak więc moduł konfiguracji musi być niezależny od pozostałych, a one z kolei od modułu konfiguracji. Tylko, że w tej sytuacji nie ma mowy o wzajemnej komunikacji, a dane muszą w końcu znaleźć się w CMDB.



Ilustracja . Tabele przechowujące dane pozostałych typów elementów konfiguracji

Problem został rozwiązany z użyciem mechanizmu adapterów. Każdy moduł zyskuje maksymalną elastyczność i może być tworzony niezależnie od modułu konfiguracji. Jednak po ukończeniu tworzenia modułu należy dla każdego typu zarządzanych przez niego elementów konfiguracji stworzyć odpowiedni łącznik z CMDB. Adapter istnieje ponad modułami i zna ich strukturę, dlatego potrafi przetłumaczyć encje modułu na elementy konfiguracji. W praktyce tworzy on podtyp abstrakcyjnego elementu konfiguracji (*C2\_ITEMS*) będący pełnoprawnym elementem konfiguracji wczytywanym przez moduł konfiguracji. Nowy typ elementu zawiera zwykle jedynie wskazanie na encję znajdującą się w innym module. Ponadto adapter udostępnia komponent, który wypisuje właściwości elementów nowego typu. Moduł konfiguracji potrafi dzięki temu pokazywać elementy nowych typów elementów konfiguracji wraz z ich właściwościami. Nie potrafi jednak ich edytować. Jest to całkowicie zamierzone, ponieważ tylko macierzyste moduły są w stanie wyegzekwować reguły biznesowe dotyczące tych elementów. Ponadto elementy dowolnego typu mogą wchodzić w relacje z elementami innych typów. Przykładami zaimplementowanych w systemie typów elementów konfiguracji są: pracownicy, usługi, problemy, incydenty i wnioski o zmianę. Warto zauważyć, że element uniwersalny to także jeden z podtypów elementu. Za pomocą adapterów można także dołączyć do CMDB dane z innych źródeł np.: ustrukturyzowanych plików tekstowych, usług sieciowych itp.

Kwestie wydajności

Projekt schematu wykorzystuje dziedziczenie z oddzielną tabelą na podklasę, ponieważ jest to jedyny z trzech sposobów, który pozwala na zachowanie więzów referencyjnych. Skutkuje to koniecznością wykonywania wielu złączeń podczas wykonywania zapytań na bazie danych. Dodatkowo schemat generyczny także wymaga złączenia aby pobrać wartości atrybutów elementu. Ile zatem odczytów potrzeba łącznie, aby pobrać wartości atrybutów elementu uniwersalnego o znanym identyfikatorze? Odpowiedź jest na to pytanie jest zaskakująca. Wystarczy odczytanie dokładnie jednego indeksu i jednego bloku danych[[78]](#footnote-78). Jest to możliwe dzięki wykorzystaniu mechanizmu klastrów. Klaster pozwala umieścić wiersze różnych tabel posiadające tą samą wartość klucza w jednym bloku danych. Zakładając klaster na identyfikatorze elementu konfiguracji i dodając do niego odpowiednie tabele, można za pomocą jednego odczytu od razu otrzymać całą hierarchię dziedziczenia typu elementu wraz z atrybutami.

## Implementacja!

Moduł zarządzania konfiguracją składa się z ośmiu pakietów

|  |  |
| --- | --- |
| Nazwa usługi | Funkcja |
| EntityClassService | Odczyt klas encji (zarówno relacji jak i elementów) |
| ItemClassService | Odczyt klas elementów konfiguracji |
| UniversalItemClassService | Odczyt i edycja klas uniwersalnych elementów konfiguracji |
| RelationshipClassService | Odczyt i edycja klas relacji |
| ItemService | Odczyt elementów konfiguracji (wszystkich typów) |
| UniversalItemService | Odczyt i edycja uniwersalnych elementów konfiguracji |
| RelationshipService | Odczyt i edycja relacji |
| AttributeService | Odczyt i edycja atrybutów encji |

Tabela . Usługi udostępniane przez moduł zarządzania konfiguracją

## Testy!

## Rozwój

W kolejnych wersjach moduł zarządzania konfiguracją powinien zostać rozszerzony o obsługę zewnętrznych źródeł danych między innymi serwerów LDAP. Szczególnie interesujące jest wykorzystanie programów zbierających dane o komputerach znajdujących się w sieci i oprogramowaniu na nich zainstalowanym. Dzięki integracji z takimi programami moduł zarządzania konfiguracją miałby dostęp do bieżących danych i automatycznie uwzględniałby wszystkie modyfikacje. Ponadto moduł mógłby monitorować stan maszyn, co pozwoliło by na implementację procesów zarządzania ciągłością i dostępnością usług.

# Moduł zarządzania zmianą

## Wymagania

Moduł służy do przechowywania i przetwarzania żądań zmian (RFC) zgodnie z ITIL. Ocenione i autoryzowane zmiany są podstawą do modyfikacji CMDB przy użyciu modułu konfiguracji.

Spis wymagań funkcjonalnych

1. Przyjmowanie nowych żądań zmian.
   1. Zgłaszane nowe żądania zmian posiadają unikalny identyfikator, krótki tytuł, dokładny opis, wstępny priorytet, wstępną kategorię oraz odnotowane: autora i czas złożenia.
   2. Żądanie stanowi element konfiguracji i może być powiązane różnymi relacjami z innymi elementami.
   3. Ponadto żądanie może posiadać komentarze.
   4. Użytkownik może usuwać swoje nowe żądania zmian.
   5. Zarządzający konfiguracją może przejąć nowe żądanie zmiany, zmieniając jego stan na nieautoryzowane żądanie zmiany.
2. Zarządzanie nieautoryzowanymi żądaniami zmian.
   1. Nieautoryzowane żądania zmian mogą być filtrowane (odrzucane) przez kierującego nimi zarządzającego zmianą po podaniu przyczyny.
   2. Zarządzający zmianą może zmienić priorytet i kategorię ustawioną przez autora żądania.
   3. Zarządzający określa typ zmiany (np. rutynowa, mała, ważna, duża).
   4. Następnie żądanie jest wysyłane do organu autoryzującego zmiany. Organ może różnić się w zależności od kategorii, priorytetu (wraz z pilnością) i typu.
   5. Organ autoryzujący żądanie autoryzuje bądź odrzuca podając przyczynę.
      1. Przy pozytywnym rozpatrzeniu może opcjonalnie podać proponowany czas implementacji zmiany.
3. Zarządzanie autoryzowanymi zmianami.
   1. Wszystkie autoryzowane zmiany trafiają do powiązanej z ich kategorią rady ds. zmian lub w przypadku zmian pilnych — nadzwyczajnej rady ds. zmian.
   2. Rada może zmiany odrzucić lub zmienić ich priorytet i typ.
   3. Rada wpisuje zmiany do harmonogramu zmian oraz określa osobę odpowiedzialną za budowanie i testowanie zmiany. Stan żądania zmiany zmienia się na implementowane.
4. Zarządzanie implementowanym żądaniami zmianami.
   1. Budowniczy po wykonaniu prac zmienia status żądania na gotowe do wdrożenia.
   2. Zarządzający zmianą koordynuje wdrożenie i odnotowuje jego wynik w żądaniu zmiany.
   3. W przypadku nieudanego wdrożenia zmiana zostaje wycofana i przekazana z powrotem do rady ds. zmian. Natomiast w przypadku udanego wdrożenia, zarządzający zmianą dołącza do żądania sprawozdanie z implementacji i zamyka żądanie.
5. Zarządzanie odrzuconymi żądaniami zmian.
   1. Dowolny użytkownik może zgłosić apelację od odrzucenia żądania po podaniu odpowiedniego uzasadnienia.
   2. Apelacja może zostać uznana przez zarządzającego zmianą.
   3. Zmiany, których apelacja została uznana, stają się zmianami nieautoryzowanymi.
6. Wyświetlanie harmonogramu zmian.
7. Zarządzanie poziomami priorytetów zmian.
   1. Priorytet ma unikalną nazwę.
   2. Niektóre poziomy mogą być oznaczone jako pilne. Zmiany o priorytecie oznaczonym jako pilny trafiają do nadzwyczajnej rady ds. zmian.
8. Zarządzanie kategoriami żądań zmian.
   1. Kategoria posiada unikalną nazwę.
   2. Kategorie tworzą strukturę hierarchiczną.
   3. Kategoria może posiadać przypisaną zwykłą i nadzwyczajną radę ds. zmian, do której kierowane są wszystkie autoryzowane zmiany danej kategorii. W przeciwnym przypadku zmiany te są wysyłane do rady związanej z kategorią nadrzędną.
9. Zarządzanie typami zmian.
   1. Typ zmiany posiada unikalną nazwę.
10. Zarządzanie organami autoryzującymi.
    1. Organ autoryzujący posiada unikalną nazwę oraz może składać się z jednego lub więcej użytkowników systemu.
    2. Organ otrzymuje zmiany wymagające autoryzacji określonego typu z określonej kategorii.
    3. Szczególnymi rodzajami organu autoryzującego jest Rada ds. Zmian (CAB) i nadzwyczajna Rada ds. Zmian (CAB/EC).

Spis wymagań niefunkcjonalnych

1. Żądania zmian mogą być składane tylko przez uprawnionych użytkowników.
2. Żądania zmian mogą być zarządzane tylko przez uprawnionych zarządzających zmianą.
3. Poziomy priorytetów, kategorie zmian i zakresy wpływu mogą być zarządzane tylko przez uprawnionych administratorów zmian.
4. System musi przechowywać nie mniej niż sto tysięcy wniosków zmian.
5. Czas odpowiedzi systemu na zapytania przy takim obciążeniu nie powinien przekroczyć trzech sekund.

## Projekt

### Schemat bazy danych

## Implementacja

## Testy

## Rozwój

# Użytkowanie systemu

Rozdział ten ma na celu zobrazowanie zastosowań zaimplementowanego systemu po wdrożeniu go w firmie RedHost. Opisuje funkcjonalność systemu podczas realizacji kilku wybranych przypadków użycia na poziomie systemowym, skupiając się na przedstawieniu jej z perspektywy użytkownika. Rozdział zawiera wiele ilustracji pokazujących wygląd interfejsu graficznego systemu.

## Dodawanie klas elementów konfiguracji

Przegląd przypadku użycia

|  |  |
| --- | --- |
| Aktor | Administrator konfiguracji |
| Opis | Użytkownik pragnie dodać do systemu nową klasę elementów konfiguracji o określonej. |
| Warunki wstępne | Użytkownik został uwierzytelniony i posiada uprawnienia administratora konfiguracji. |
| Warunki końcowe | Nowa klasa jest zapisana w systemie. |

Przebieg przypadku użycia

1. Użytkownik wybiera z menu opcję dodania klasy elementów konfiguracji.
2. System wyświetla formularz dodania nowej klasy elementów konfiguracji
3. Użytkownik wypełnia formularz niezbędne dane i wysyła formularz.
4. System wyświetla komunikat o sukcesie operacji utworzenia nowej klasy i przekierowuje użytkownika do strony z jej szczegółami.

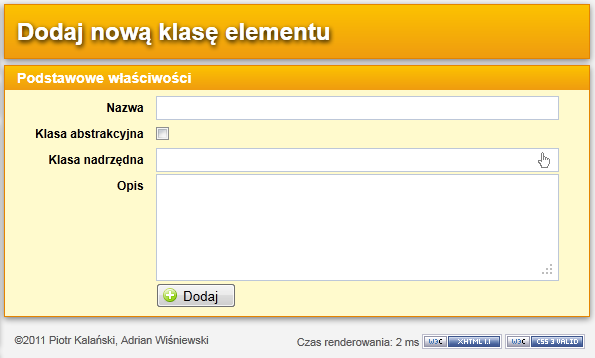
Szczegóły przebiegu

Użytkownik po zalogowaniu widzi swoją stronę główną, która gromadzi wszystkie ważne informacje. Po lewej stronie ma do dyspozycji menu, z którego może wybierać interesujące go funkcje. Zostały one podzielone na grupy odpowiadające procesom ITIL, więc znalezienie odpowiedniej nie nastręcza żadnych problemów.



Ilustracja . Panel główny systemu

Po wybraniu opcji *Dodaj* z grupy klas elementów konfiguracji, system wyświetla formularz dodawania nowej klasy elementu. Użytkownik podaje w nim wszystkie dane nowej klasy łącznie z klasą nadrzędną, po której dziedziczy. W przypadku tej właściwości użyta została specjalna kontrolka wyświetlająca po kliknięciu okienko zawierające drzewo wszystkich klas elementów. Po wypełnieniu danych, użytkownik wysyła formularz naciskając przycisk *dodaj*.



Ilustracja . Formularz dodawania nowej klasy elementu konfiguracji



Ilustracja . Widok szczegółów klasy elementów

Po wysłaniu formularza, system dodaje nową klasę i przekierowuje użytkownika do strony z jej szczegółami. U góry ekranu pojawia się jest komunikat świadczący o powodzeniu wykonania operacji. Nowo dodana klasa elementu nie posiada własnych atrybutów. Można je utworzyć klikając przycisk dodaj znajdujący się w ramce *Atrybuty klasy*. Dziedziczy natomiast wszystkie atrybuty po klasie bazowej – są one pokazane w ramce *Wszystkie atrybuty*. Jeżeli klasa została utworzona jako abstrakcyjna nie można tworzyć jej obiektów. Można za to klasę taką wykorzystać do grupowania wspólnych atrybutów innych klas. Dzięki temu nie trzeba kopiować powtarzających się atrybutów, co znacznie upraszcza zarządzanie klasami konfiguracji.

## Dodawanie relacji między elementami konfiguracji

## Składanie wniosków o zmianę

## Zarządzanie wioskami o zmianę

# Podsumowanie

Na powyższej liście znajdują się jedynie główne pakiety aplikacji. W sumie aplikacja składa się z 110 pakietów, 436 klas co łącznie daje ponad 20 tysięcy linii kodu (KLOC[[79]](#footnote-79)). .

Bibliografia

1. **Kalański Piotr.** *System Service Desk zgodny z zaleceniami ITIL.* 2010.

2. **OGC.** ITILv3 Glossary Polish. *ITIL Official Site.* [Online] 2008. [Zacytowano: 10 Grudnia 2010.] http://www.itil-officialsite.com/nmsruntime/saveasdialog.aspx?lID=921&sID=242.

3. —. *Service Transition Book (ITIL).* Londyn : The Stationery Office, 2007.

4. **Knowledge Transfer.** Słownik pojęć ITIL. [Online] [Zacytowano: 20 Lipiec 2011.] http://www.knowledgetransfer.net/dictionary/ITIL/en/IT\_Service\_Management.htm.

5. —. Słownik pojęć ITIL - hasło Function. [Online] [Zacytowano: 20 Lipiec 2011.] http://www.knowledgetransfer.net/dictionary/ITIL/en/Function.htm.

6. **Global Journals Inc.** *Global Journal of Computer Science and Technology.* [Online] Grudzień 2010. [Zacytowano: 18 Lipiec 2011.] http://computerresearch.org/stpr/index.php/gjcst/article/download/465/424.

7. ITIL Service Management - A Brief History of ITIL. *ITIL Service Management.* [Online] 18 Lipca 2011. [Zacytowano: 18 Lipca 2011.] http://itservicemngmt.blogspot.com/2007/09/brief-history-of-itil.html.

8. IT Life - Korzenie ITIL. *IT Life.* [Online] [Zacytowano: 18 Lipiec 2011.] http://itsm.itlife.pl/content/view/10012/57/.

9. **Sante Tom i Jeroen Ermers.** *TOGAF™ 9 and ITIL® V3 Two Frameworks Whitepaper.* 2009.

10. **Ratcliffe David.** Pink Elephant - IT Service Management 2007. *Pink Elephant.* [Online] [Zacytowano: 18 Lipiec 2011.] http://www.pinkelephant.com/articles/MicrosoftPowerPointDavidRatcliffePinkPerspectiveBM.pdf.

11. *actis-ingenierie.* [Online] http://www.actis-ingenierie.com.

12. Projekt ITIL Refresh. *ITLife.pl.* [Online] [Zacytowano: 19 Lipiec 2011.] http://itsm.itlife.pl/content/view/10071/138/.

13. **OGC.** Scope and Development Plan: ITIL® V3 Update. [Online] February 2010. [Zacytowano: 21 Lipiec 2011.] http://www.best-management-practice.com/gempdf/Scope\_and\_Development\_Plan\_ITIL\_V3\_Update.pdf.

14. IT Life - Co to jest ITIL? *IT Life.* [Online] [Zacytowano: 7 Lipiec 2011.] http://itsm.itlife.pl/.

15. IT frameworks wiki - Hasło ITIL. *IT frameworks wiki.* [Online] [Zacytowano: 7 Lipiec 2011.] http://www.itframeworks.org/wiki/Information\_Technology\_Infrastructure\_Library.

16. **OGC.** *Service Strategy Book (ITIL).* brak miejsca : The Stationery Office, 2007.

17. **itSMF Ltd.** *An Introductory Overview of ITIL® V3.* brak miejsca : The UK Chapter of the itSMF, 2007.

18. **OGC.** *Service Operation Book (ITIL).* brak miejsca : The Stationery Office, 2007.

19. —. *Continual Service Improvement Book (ITIL).* brak miejsca : The Stationery Office, 2007.

20. **Knowledge Transfer.** Słownik pojęć ITIL. [Online] [Zacytowano: 20 Lipiec 2011.] http://www.knowledgetransfer.net/dictionary/ITIL/en/Configuration\_Item.htm.

21. **Wikipedia.** Configuration management - Wikipedia, the free encyclopedia. [Online] [Zacytowano: 20 Lipiec 2011.] http://en.wikipedia.org/wiki/File:ConfiurationActivityModel.png.

22. **Knowledge Transfer.** Słownik pojęć ITIL. [Online] [Zacytowano: 20 Lipiec 2011.] http://www.knowledgetransfer.net/dictionary/ITIL/en/Change.htm.

23. **Gamma Erich , i inni.** *Wzorce projektowe. Elementy oprogramowania obiektowego wielokrotnego użytku.* brak miejsca : Wydawnictwa Naukowo Techniczne, 2005.

24. **Fowler Henry.** *Architektura systemów zarządzania przedsiębiorstwem. Wzorce projektowe.* brak miejsca : Helion, 2005.

25. **Stochmiałek Michał.** Wprowadzenie do programowania aspektowego. [Online] http://stochmialek.pl/papers/aop-intro-seminar.pdf.

26. **Collins Dan.** Generic Data Modeling. [Online] http://www.dama-nj.org/presentations/Kalido\_Generic\_Data\_Modeling.pdf.

27. **JBoss.** Dokumentacja Hibernate. [Online] http://docs.jboss.org/hibernate/core/3.3/reference/en/html/inheritance.html.

28. **Keogh Patric.** The Value of Cloud Computing to ITSM. [Online] [Zacytowano: 14 Lipiec 2011.] http://www.slideshare.net/patrick\_keogh/the-value-of-cloud-computing-to-itsm.

29. **Scott Noel.** Case Study: Using ITIL® and PRINCE2™ Together. [Online] 2010. http://www.best-management-practice.com/gempdf/Using\_ITIL\_and\_PRINCE2\_Together\_August\_2010.pdf.

Zawartość płyty CD

Uruchamianie systemu

Wykaz skrótów

Spis ilustracji

Ilustracja 2.1. Obszary ITIL v2 [11] 17

Ilustracja 2.2. Cykl życia usługi w ITIL [14] 18

Ilustracja 2.3. Procesy ITIL pogrupowane wg hierarchii zarządzania [15] 19

Ilustracja 2.4 Model czynności zarządzania konfiguracją [21] 29

Ilustracja 3.1. Przykładowe elementy konfiguracji firmy 40

Ilustracja 4.1 Warstwy systemu 52

Ilustracja 4.2 Schemat podziału warstwy aplikacji na podwarstwy 52

Ilustracja 4.3 Schemat budowy warstwy prezentacji 54

Ilustracja 4.4 Tabele związane z danymi firmy 55

Ilustracja 4.5 Tabele związane z autoryzacją użytkowników i danymi firmy 56

Ilustracja 4.6 Tabele związane z ogłoszeniami i powiadomieniami 57

Ilustracja 4.7 Diagram pakietów systemu Service Desk 58

Ilustracja 5.1 Tabele przechowujące dane o uniwersalnych elementach konfiguracji 65

Ilustracja 5.2 Tabele przechowujące dane pozostałych typów elementów konfiguracji 68

Ilustracja 7.1 Panel główny systemu 75

Ilustracja 7.2 Formularz dodawania nowej klasy elementu konfiguracji 75

Ilustracja 7.3 Widok szczegółów klasy elementów 76

1. Office of Government Commerce — komórka rządu Wielkiej Brytanii. W momencie stworzenia ITIL działająca pod nazwą Central Computer and Telecommunications Agency (CCTA). [↑](#footnote-ref-1)
2. ang. Projects In Controlled Environment — metodyka zarządzania projektami [↑](#footnote-ref-2)
3. ang. Managing Successful Programmes — metodyka zarządzania programami [↑](#footnote-ref-3)
4. ang. Management of Risk — metodyka zarządzania ryzykiem [↑](#footnote-ref-4)
5. ang. Cloud Computing, [28] [↑](#footnote-ref-5)
6. ang. Black Box [↑](#footnote-ref-6)
7. ang. Process Owner [↑](#footnote-ref-7)
8. ang. Service Level Agreement — umowa określająca warunki świadczenia usługi, w szczególności mierzalne parametry dotyczące wydajności i awaryjności. [↑](#footnote-ref-8)
9. ang. Service Portfolio [↑](#footnote-ref-9)
10. ang. Service Catalogue [↑](#footnote-ref-10)
11. ang. Service Catalogue Management [↑](#footnote-ref-11)
12. ang. Service Level Management [↑](#footnote-ref-12)
13. ang. Service Level Requirement — dokument zawierający biznesowe wymagania dotyczące usługi [↑](#footnote-ref-13)
14. ang. Information Security Management [↑](#footnote-ref-14)
15. ang. IT Service Continuity Management (ITSCM) [↑](#footnote-ref-15)
16. ang. Supplier Managment [↑](#footnote-ref-16)
17. ang. Service Design Package (SDP) [↑](#footnote-ref-17)
18. ang. Service Acceptance Criteria (SAC) [↑](#footnote-ref-18)
19. ang. Transition Planning and Support [↑](#footnote-ref-19)
20. ang. PRojects IN Controlled Environments 2 [↑](#footnote-ref-20)
21. ang. Service Validation and Testing [↑](#footnote-ref-21)
22. ang. Evaluation [↑](#footnote-ref-22)
23. ang. Release and Deployment Management [↑](#footnote-ref-23)
24. ang. Release Package [↑](#footnote-ref-24)
25. ang. Early Life Support (ELS) [↑](#footnote-ref-25)
26. ang. Request for Change [↑](#footnote-ref-26)
27. ang. Change Management [↑](#footnote-ref-27)
28. ang. Zarządzanie komponentami usług i konfiguracją (SACM) [↑](#footnote-ref-28)
29. ang. Configuration Management Database [↑](#footnote-ref-29)
30. ang. Configuration Item (CI) [↑](#footnote-ref-30)
31. ang. Knowledge Management [↑](#footnote-ref-31)
32. ang. Event Management [↑](#footnote-ref-32)
33. ang. Event [↑](#footnote-ref-33)
34. ang. Incident [↑](#footnote-ref-34)
35. ang. Incident Management [↑](#footnote-ref-35)
36. ang. Problem [↑](#footnote-ref-36)
37. ang. Problem Management [↑](#footnote-ref-37)
38. ang. Known Error Record (KER) [↑](#footnote-ref-38)
39. ang. Known Error Database (KEDB) [↑](#footnote-ref-39)
40. ang. Access Management [↑](#footnote-ref-40)
41. ang. Service Request [↑](#footnote-ref-41)
42. ang. Request Fulfilment [↑](#footnote-ref-42)
43. ang. Service Measurement [↑](#footnote-ref-43)
44. ang. Service Reporting [↑](#footnote-ref-44)
45. mimo takich samych nazw należy rozróżnić proces CSI od obszaru CSI [↑](#footnote-ref-45)
46. ang. Configuration Management Database [↑](#footnote-ref-46)
47. ang. Configuration Model [↑](#footnote-ref-47)
48. ang. Configuration Management System (CMS) [↑](#footnote-ref-48)
49. ang. Configuration Manager [↑](#footnote-ref-49)
50. ang. Major Release [↑](#footnote-ref-50)
51. ang. Request For Change [↑](#footnote-ref-51)
52. ang. urgent [↑](#footnote-ref-52)
53. ang. Change Record [↑](#footnote-ref-53)
54. ang. Change Manager [↑](#footnote-ref-54)
55. ang. Change Advisory Board [↑](#footnote-ref-55)
56. ang. Emergency Advisory Board [↑](#footnote-ref-56)
57. ang. Post Implementation Review (PIR) [↑](#footnote-ref-57)
58. ang. Data Centre [↑](#footnote-ref-58)
59. ang. Reseller Hosting [↑](#footnote-ref-59)
60. ang. Operational Level Agreement – definiuje sposób współpracy między zespołami wewnątrz firmy [↑](#footnote-ref-60)
61. ang. Underpinning Contract – umowa z dostawcą np. Internetu. [↑](#footnote-ref-61)
62. Należy odczytać jedną wartość w zależności od priorytetu zmiany: niski/normalny/wysoki/bardzo wysoki. Jednostki: d – dzień, g – godzina, m – minuta. [↑](#footnote-ref-62)
63. ang. Create, Read, Update, Delete. Operacje pozwalające dodawać, odczytywać, uaktualniać i usuwać dane. [↑](#footnote-ref-63)
64. ang. web services [↑](#footnote-ref-64)
65. ang. Supply chain management — Zarządzanie łańcuchem dostaw [↑](#footnote-ref-65)
66. ang. Customer relationship management — Zarządzanie relacjami z klientami [↑](#footnote-ref-66)
67. ang. Enterprise resource planning — Zarządzanie zasobami przedsiębiorstwa [↑](#footnote-ref-67)
68. ang. Inversion Of Control (IOC) Container [↑](#footnote-ref-68)
69. ang Object-relational mapping (ORM) [↑](#footnote-ref-69)
70. ang. Java DataBase Connectivity – interfejs programistyczny umożliwiający komunikację z bazą danych [↑](#footnote-ref-70)
71. Wzorzec MVC stosowany w aplikacjach sieciowych trochę różni się od używanego w zwykłych aplikacjach okienkowych. [↑](#footnote-ref-71)
72. ang. Model-View-Controller [↑](#footnote-ref-72)
73. ang. Dependency Injection [↑](#footnote-ref-73)
74. ang. Spring Expression Language – rozszerzenie języka wyrażeń EL w bibliotece Spring [↑](#footnote-ref-74)
75. ang. Data tampering [↑](#footnote-ref-75)
76. ang. Data Manipulation Language – podzbiór zapytań języka SQL używanych do modyfikacji danych [↑](#footnote-ref-76)
77. w terminologii Domain Driven Development nazywana korzeniem agregatu (ang. aggregate root) [↑](#footnote-ref-77)
78. pod warunkiem, że blok jest wystarczająco duży, żeby pomieścić wszystkie dane klastra. [↑](#footnote-ref-78)
79. ang. Kilo Lines Of Code [↑](#footnote-ref-79)