

FRAMEWORK

INDIGO

DO INTEGROWANIA
MASZYN I SYSTEMÓW IT
ORAZ STEROWANIA NIMI ZA POMOCĄ
GRAFICZNEGO MODELOWANIA PROCESÓW

STABILIS

POWERED BY INDIGO

W jakim kierunku idziemy i dlaczego?

Poszukiwanie metody, która zapewni ludziom komfort porozumiewania się z maszynami w zrozumiały sposób, jest jednym z głównych kierunków rozwoju informatyki. Języki programowania stają się coraz bardziej zbliżone do języków naturalnych, co ułatwia i znacznie przyspiesza proces tworzenia niezawodnego oprogramowania.

Historia programowania rozpoczęła się od języka maszynowego, w którym komendy w postaci zer i jedynek (kod binarny) wydawane były wprost do procesora w urządzeniu. Oznaczało to, że dla każdego procesora trzeba było pisać osobne instrukcje. Następnie pojawiły się niskopoziomowe języki programowania, w których można było wykonać pewne operacje za pomocą oznaczeń symbolicznych. Ten trend umacniał się w kolejnych językach, takich jak C i C++, które coraz bardziej odchodziły od kodu maszynowego.

Były one dużo bardziej uniwersalne i umożliwiły rozwiązanie szerokiego zakresu problemów. Dodatkowo języki te – z których część jest używana i rozwijana do dziś – pozwoliły na znaczne skrócenie czasu tworzenia aplikacji i na zmniejszenie liczby błędów dzięki automatyzacji zadań wcześniej wykonywanych przez programistów. Przede wszystkim jednak pozwoliły na oddzielenie oprogramowania od warstwy sprzętowej, co umożliwiło jego przenośność, czyli możliwość wykorzystania tego samego kodu na różnych platformach sprzętowych lub w różnych systemach operacyjnych.

My zrobiliśmy kolejny krok naprzód i stworzyliśmy framework Indigo – technologię do integracji maszyn i systemów IT oraz do sterowania nimi za pomocą grafów. Indigo nie tylko może być przenoszone na różne platformy, ale co ważniejsze – pozwala łączyć ze sobą dowolne maszyny, systemy IT (np. ERP), sterowniki PLC czy różnego rodzaju automaty w taki sposób, aby poszczególne węzły mogły nie tylko swobodnie wymieniać między sobą informacje, ale także kontrolować wzajemnie swoje zachowanie.

Co więcej, programowanie w Indigo polega na modelowaniu poszczególnych procesów na grafach w notacji BPMN 2.0. Co z tego wynika? O ile w tradycyjnych językach programowania najmniejszy samodzielny fragment kodu jest w gruncie rzeczy listą prostych instrukcji, które tak naprawdę niewiele zmieniły się od czasów maszyny Turinga, o tyle w naszym przypadku jest to gotowy element jakiegoś procesu, który pełni konkretną funkcję, np. „wyświetl okno z komunikatem” czy też „wyślij komendę do urządzenia”.

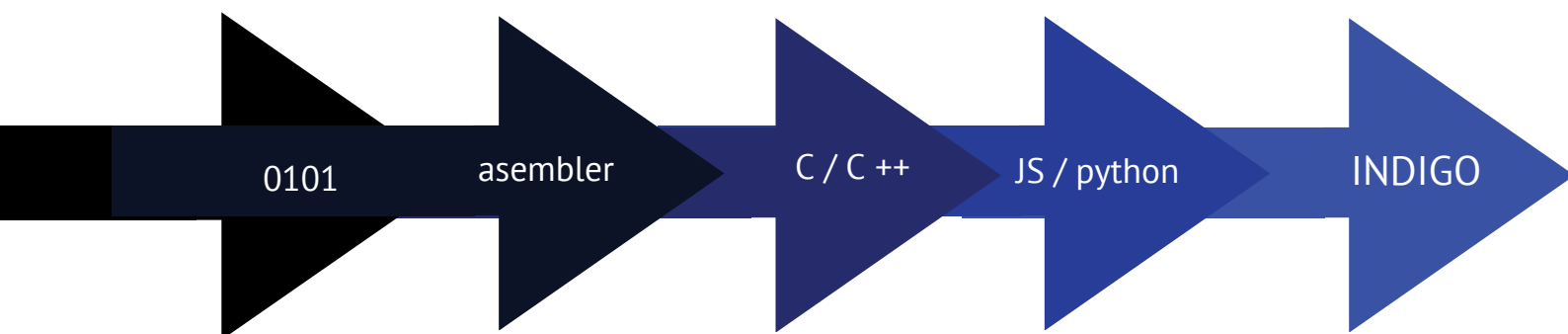
Aby zaprogramować cały proces w Indigo, wystarczy więc zamodelować jego przebieg na grafie. Poszczególne komponenty grafu mogą być poleceniami dla serwera do zapisu lub odczytu danych lub wykonania konkretnej funkcji, poleceniami dla urządzeń do wykonania zdefiniowanych wcześniej akcji, mogą też inicjować dwukierunkową komunikację z systemami IT.

Tworząc oprogramowanie w Indigo, nie myślimy więc o samym kodzie źródłowym jak w przypadku programowania w tradycyjnych językach, ale o jego funkcjach, które są wywoływane przez poszczególne elementy grafu.

Grafy są naszym językiem programowania – zrozumiałym nie tylko dla programistów.

Rys. 1

Ewolucja języków programowania



Dlaczego to tak dobrze działa?

Czyli architektura i technologie.

Architektura Indigo jest modułowa i przejrzysta. Została zaprojektowana w sposób, który przewiduje jej rozwijanie. Pozwala także na skalowanie poziome, czyli umożliwia zwiększenie wydajności oprogramowania przez dodawanie kolejnych serwerów (wirtualnych/fizycznych). Dodatkowo, system ma wbudowane mechanizmy agregowania informacji, więc jest przygotowany na przyjmowanie dużej ilości informacji (z wielu urządzeń) z bardzo dużą częstotliwością.

Na wyższym poziomie szczegółowości można opisać architekturę Indigo jako opartą o rozproszone węzły odpowiedzialne za realizację specjalistycznych zadań związanych ze sterowaniem, integracją oraz zbieraniem i przetwarzaniem danych od urządzeń i użytkowników. Każdy z węzłów jest samodzielny w działaniu i po odłączeniu od sieci, np. w przypadku awarii, nadal zbiera i przetwarza informacje, aby przestać je dalej natychmiast po wznowieniu komunikacji.

Szczególnym elementem systemu jest węzeł centralny, który posiada zdolność działania w wielu instancjach i umożliwia wysoką skalowalność rozwiązania oraz świadczenie usług w modelu SaaS.

Konstrukcja poszczególnych poziomów systemu oraz programowanie procesów za pomocą grafów umożliwiają zmianę w czasie rzeczywistym logiki działania każdego z elementów systemu oraz podmianę na inny posiadający taki sam zakres funkcjonalny.

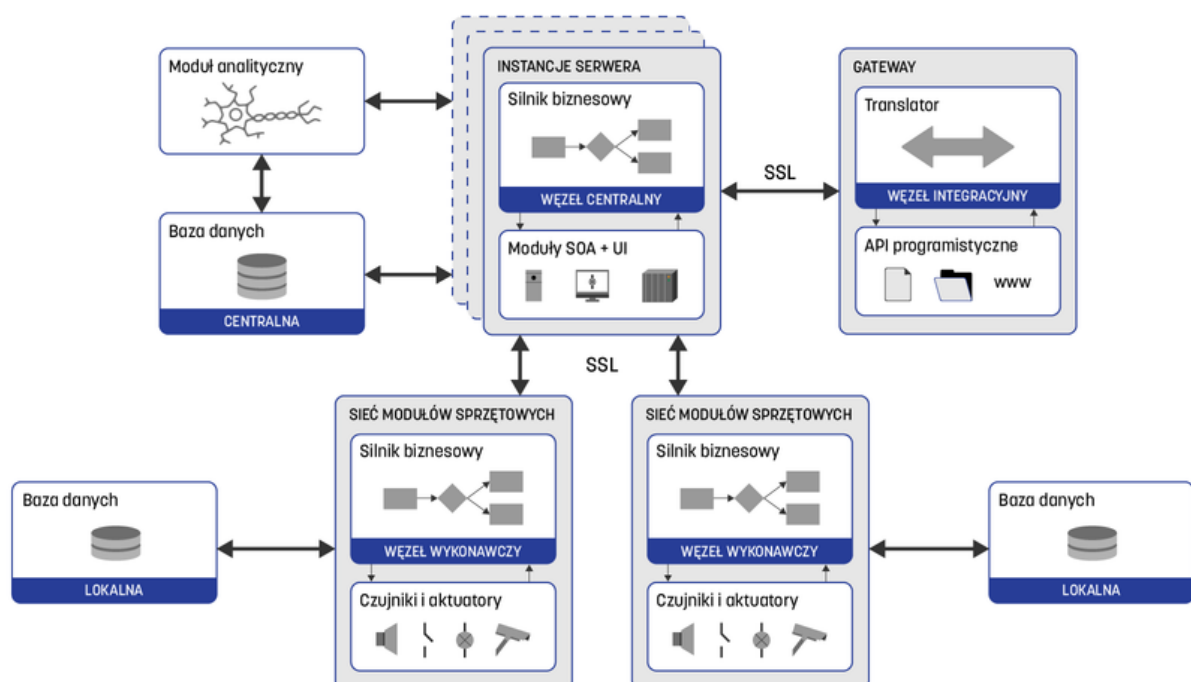
Z kolei izolacja poszczególnych poziomów systemu oraz programowanie procesów za pomocą grafów umożliwiają zmianę w czasie rzeczywistym logiki działania każdego z elementów systemu oraz podmianę na inny posiadający taki sam zakres funkcjonalny.

Tak zaprojektowana architektura spełnia wszystkie postulaty Przemysłu 4.0 umożliwiając przeobrażanie fabryk w czasie rzeczywistym, krótkoseryjną produkcję pod zamówienia jednostkowe przy jednoczesnym dbaniu o jakość oraz optymalizację na każdym poziomie działania produkcji, przedsiębiorstwa oraz całej organizacji.

Aby umożliwić użytkownikom stosowanie optymalnych rozwiązań sprzętowych i programistycznych pochodzących od różnych producentów, a jednocześnie zapobiec przywiązaniu tylko do jednego dostawcy (vendor lock in), Indigo w całości oparte jest o bezlicencyjne technologie OpenSource: framework działa na systemach operacyjnych GNU Linux i wykorzystuje sprawdzone, używane i rozwijane rozwiązania, takie jak serwer baz danych PostgreSQL, ORM SQLAlchemy, frameworki VueJS i Polymer, serwer HTTP nginx oraz języki programowania Python 3.7, JavaScript/TypeScript i C++.

Bezpieczeństwo danych przetwarzanych przez Indigo jest zapewnione przez asymetryczne szyfrowanie komunikacji za pomocą protokołów SSL/TLS.

Rys. 2
Architektura systemu



Silnik biznesowy

Najważniejszym elementem technologii Indigo jest lekki silnik procesowy programowany za pomocą grafów procesowych w notacji BPMN. Silnik może działać na urządzeniach klasy IoT sterując ich zachowaniem i komunikacją z innymi węzłami oraz serwerem.

Konstrukcja silnika nie ogranicza jego użycia do wybranych obszarów działania. Umożliwia wywoływanie dowolnych funkcjonalności, sterowanie interfejsami i realizację zadań synchronicznych oraz planowanych. Silnik realizuje swoje zadania na podstawie wgranych do niego grafów, które podlegają dalszej kompilacji. Wspiera obsługę uprawnień dzięki czemu umożliwia wywoływanie zadań bezpośrednio z interfejsu użytkownika.

Dzięki oparciu technologii Indigo o języki skryptowe oprogramowanie może być uruchamiane na dowolnych urządzeniach działających w architekturze x86, ARM oraz na mikrokontrolerach. Integracja z technologią Electron i niskie zapotrzebowanie na moc obliczeniową pozwalają uruchamiać Indigo na urządzeniach przenośnych: tabletach, komórkach, handheld'ach oraz jego aplikację desktopową na komputerach PC i Industry PC.

Komunikacja zewnętrzna i wewnętrzna systemu posiada własny lekki i znormalizowany protokół bazujący na JSON umożliwiający zbieranie i zapis danych w uspołnionej formie niezależnie od źródła ich pochodzenia (nazywamy to komunikacją funkcjonalną). Wszystkie dane mogą być zapisywane (każdy węzeł posiada własną bazę) dzięki czemu np. po restarcie proces może być kontynuowany. Aktualnie system oparty jest o bazę PostgreSQL, ale dzięki zastosowaniu warstwy pośredniej SQLAlchemy możliwe jest podłączenie każdej innej nowoczesnej relacyjnej bazy danych.

Silnik pozwala na wykorzystanie wielu procesów jednocześnie synchronicznie i asynchronicznie oraz ich planowanie w cron. Możliwe jest kolejgowanie komunikatów w przypadku offline i ich wysyłanie po przywróceniu łączności.

Grafy

Technologia Indigo oparta jest o programowanie graficzne. Grafami programuje się każdą warstwę logiczną – od sterowania modułami sprzętowymi, poprzez opis logiki działania wielu modułów towarzyszących urządzeniu, obiegu informacji w grupie urządzeń aż do przesyłania informacji pomiędzy różnymi grupami urządzeń i systemami IT również w odległych lokalizacjach.

Grafami sterowane jest także działanie interfejsu, tj. pokazywanie i chowanie okien, ich wewnętrzna logika działania oraz wyświetlane treści. W systemie definiuje się tylko szablony określające strukturę widoku i komponentów, z których składa się np. tabela, tekst czy przycisk.

Grafy definiują reakcję na komunikaty interfejsu, sprzętu, innych procesów określając kolejne kroki do wykonania, warunki ich wykonania oraz komunikaty do wysłania. Za ich pomocą definiuje się także obsługę błędów i zachowania w trybie offline.

Realizacja działań na grafie polega na wywołaniu funkcjonalności z generycznych bibliotek – nisko funkcjonalnych jak np. obsługa zmiennych, obsługa komunikacji przez porty, etc. i wysoko funkcjonalnych, jak np. moduł opisu przestrzennego (np. dla magazynu) czy moduł rejestracji przestojów.

Planowane jest stworzenie dedykowanego narzędzie WISIWYG do tworzenia widoków i malowania oraz debugowania grafów. Aktualnie używane jest opensource'owe oprogramowanie yED.

Integracja z zewnętrznymi systemami IT

Framework Indigo można integrować z zewnętrznymi systemami na kilka sposobów.

1. Za pomocą konektora, który umożliwia wywoływanie funkcjonalności obcego systemu za pomocą naszej komunikacji funkcjonalnej. Konektor działa w systemie na zasadzie kolejnego węzła.
2. Za pomocą modułu do wykorzystania na grafie.
3. Za pomocą szyny integracyjnej ESB.

Deployment

Indigo w pełni wspiera continuous deployment umożliwiając np. zdalne wgrywanie widoków i zasobów oraz podmianę grafów. System zbiera w sposób scentralizowany logi ze wszystkich węzłów w czasie rzeczywistym.

Framework jest w pełni zmodularyzowany tj. deployment zawiera tylko wymagane do działania moduły.

Interfejs, czyli jak komunikujemy się z użytkownikiem

Wysoka elastyczność i konfigurowalność Indigo dotyczy także interfejsu, który zaprojektowaliśmy zgodnie z najnowszymi trendami w tworzeniu aplikacji webowych.

Oparty jest o pulpity i widżety, z których konfigurujemy panele zarządzania. Budowa naszego interfejsu umożliwia prezentację różnego rodzaju informacji w różnych układach graficznych i funkcjonalnych, które wynikają z potrzeb danej implementacji. Tworzenie dedykowanych ekranów jest proste i szybkie, ponieważ komponuje się je z poziomu samego interfejsu bez konfiguracji w plikach i dodatkowego programowania (WYSIWYG). Dzięki temu interfejs może być zindywidualizowany dla każdego użytkownika. Możliwe jest także stworzenie jednolitego interfejsu dla niehomogenicznego parku maszynowego, działającego jako interfejs HMI.

Rys. 3

Przykładowy interfejs

The screenshot displays the STABILIS industrial interface. At the top, it shows the terminal ID (T50P_1056), user (Operator Maszyny), and status (PRACA [STOP]). The date and time are 9/14/2018 12:44:09 PM. The main section is titled 'DOSTĘPNE ZLECENIA PRODUKCYJNE' (Available Production Orders). Below this, there is a table 'ZLECENIA DO REALIZACJI' (Orders for Execution) with columns: NUMER ZLECENIA, KLIENT, METRAŻ [MB], DATA ZLECENIA, and STATUS. The table lists three orders: 2018/ZPS/SOK/561284 (LOREM, 200.12 MB, 2018-03-29), 2018/ZPS/SOK/556390 (IPSUM, 201.12 MB, 2018-03-29), and 2018/ZPS/SOK/556391 (DOLOR, 201.12 MB, 2018-03-29). The status for the last two is 'W TOKU'. To the right of the table are navigation buttons: up, down, search, and a 'START' button. Below the table, there is a note 'UWAGI: dostawa 26/05'. Another table 'LISTA PAKÓW W ZLECENIU - 2018/ZPS/SOK/561284' (List of packages in the order) shows three packages with their dimensions and weights. At the bottom, there are three main sections: 'OGÓLNE' (General) with a 'WYLOGUJ' button, 'MASZYNA' (Machine) with 'USTAWIENIA I KALIBRACJA', 'UTRZYMANIE RUCHU', and 'ALARMY' buttons, and 'NAWIGACJA' (Navigation) with 'RĘCZNE ZLECENIE' and 'PRODUKCJA WEWNĘTRZNA' buttons.

NUMER ZLECENIA	KLIENT	METRAŻ [MB]	DATA ZLECENIA	STATUS
2018/ZPS/SOK/561284	LOREM	200.12	2018-03-29	
2018/ZPS/SOK/556390	IPSUM	201.12	2018-03-29	
2018/ZPS/SOK/556391	DOLOR	201.12	2018-03-29	W TOKU

NUMER PAKA	DŁUGOŚĆ [MB]	CIĘŻAR [KG]	ARKUSZY	PACZKA [MB]
2018/PAK_PROD/SOK/070685	55.12	150.12	5	20.12
2018/PAK_PROD/SOK/070686	56.12	151.12	6	21.12
2018/PAK_PROD/SOK/070687	57.12	152.12	7	22.12

Za pomocą grafów sterujemy także interfejsem

Indigo w pełni wspiera mechanizmy uprawnień na łączonych rolach, internalizację, zmianę wielkości elementów interfejsu oraz rozdzielczości. Możliwe jest także dostosowanie wyglądu interfejsu (w tym w trakcie działania) w ramach schematu kolorów, wyglądu kontroltek, etc.

Kontrolki automatycznie rozkładają się w ramach zdefiniowanych sekcji (interfejs napisany jest w HTML5 z użyciem technologii flex).

Jak to robimy?

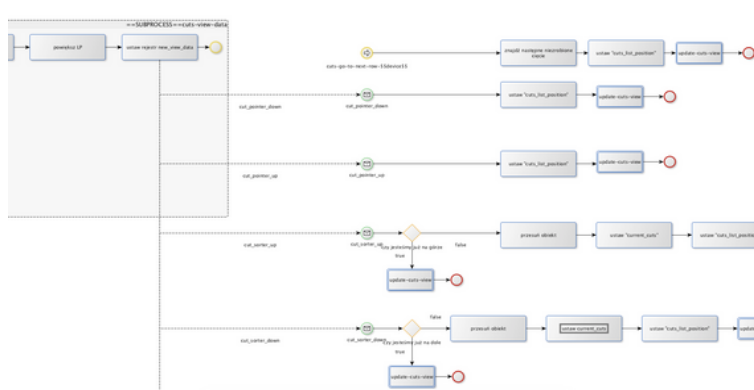
Rys. 4.

Szablon opisujący zachowanie widoku

```
<template-section id='body_section' type='vertical' scheme='fullscreen'>
  <template-section id='header_section' scheme='header'>
    <info-picture id='stabilis_logo' icon='images/stabilis-logo.png' position='start' left='--separation-big' right='--separation-big' />
    <info-line id='console_info' position='start' left='--separation-big' right='--separation-big' interspace='--separation-big' />
    <date-and-time id='clock' position='end' right='--separation-big' stick='end' />
  </template-section>
  <template-section id='main_section' type='vertical'>
    <template-section id='cuts_section' align='start'>
      <template-section id='table_section' type='vertical' size='stretch'>
        <info-table id='cut_table' grow='expand' />
        <info-line id='comments' size='--font-big' />
      </template-section>
      <template-section id='input_section' type='vertical' align='end' grow='none' size='stretch'>
        <input-number id='input_1kat' />
        <input-number id='input_2kat' stick='end' />
        <input-number id='input_waste' bottom='--separation-small' />
      </template-section>
      <template-section id='machine_section' size='stretch' grow='none'>
        <info-box id='machine_info' />
      </template-section>
      <template-section id='actions_section' scheme='commands'>
        <execute-command id='actions_navigate' />
        <execute-command id='actions_cuts' />
        <execute-command id='actions_machine' />
      </template-section>
    </template-section>
  </template-section>
```

Rys. 5.

Fragment programu w postaci grafu opisującego zachowanie widoku



Rys. 6.

Przykładowy ekran systemu odpowiedzialny za sterowanie maszyną i prezentację parametrów. Wygląd oraz sposób zachowania zostały w całości opisane za pomocą szablonu oraz grafu procesowego.

STABILIS

TERMINAL: 1245677

MASZYNA: 056

STAN: PRACA [AUTO]

1.02.2018

00:19:53

LISTA CIĘĆ

LP	PAK	DEGUOŚĆ (MB)	IŁOŚĆ (SZT)	I KAT. [SZT]	II KAT. [MB]	ODPAD (MB)	STATUS
1	P03	8500	15	15	0	0	OK
2	P03	7000	12	12	0	14	OK
3	P03	3500	43	43	0	0	OK
4	P03	8000	3	3	0	8	OK
5	P03	10000	21	8	0	0	W trakcie
6	P03	1000	70	0	0	0	
7	P03	1500	10	0	0	0	
8	P03	1250	1	0	0	0	
9	P03	1750	30	0	0	0	

↑

↓

I KAT. [SZT]

—

8

+

II KAT. [MB]

—

0

+

ODPAD [MB]

—

0

+

UWAGI: TYLKO PIERWSZY GATUNEK

BIEŻĄCE PARAMETRY PRODUKCJI

KRĄG:	6484/127/01/18	DEGUOŚĆ AKTUALNA:	-2 mm	PAUZA:	5 sek.
DOSTĘPNE:	350 m.b.	WYKORZYSTANE:	80 m.b.	POZOSTANIE:	120 m.b.

WYKONAJ DZIAŁANIE

ZLECENIA ◀◀	INSTRUKCJA UŻYCIA	DODAJ CIĘCIE	USUŃ CIĘCIE	SORTUJ WG DEGUOŚCI	USTAWIENIA I KALIBRACJA	ROZLICZ KRĄG
----------------	----------------------	-----------------	----------------	-----------------------	----------------------------	-----------------

Przewagi technologiczne

Największe korzyści z wykorzystania technologii Indigo:

- znaczące przyspieszenie tworzenia oprogramowania (od jednego do dwóch rzędów wielkości)
- zmniejszenie ilości błędów logiki, regresji i integracji
- znaczące obniżenie kosztów zmiany wymagań w trakcie projektu
- możliwość ewolucyjnego wprowadzania kolejnych funkcjonalności w zgodzie ze zwinnymi metodykami produkcji oprogramowania
- możliwość realizacji projektu przez personel o niższych kompetencjach (wdrożeńiowcy zamiast programistów)
- ułatwienie i przyspieszenie procesu analizy z klientem
- możliwość realizacji bardzo złożonych problemów
- pełne dostosowanie do potrzeb klienta na etapie wdrożenia oraz w trakcie użytkowania oprogramowania (ciągła poprawa)

Unikalne cechy i podsumowanie

To, co wyróżnia framework Indigo to dwukierunkowa komunikacja z maszynami lub systemami IT zaprogramowana za pomocą grafów.

Możemy nie tylko zbierać dane, ale także na nie reagować, wysyłając polecenia zwrotne dla maszyn czy informacje zwrotne dla systemów IT (np. ERP).

Dzięki temu, że każdy z procesów przebiegających w dowolnym systemie lub sieci maszyn jest zaprogramowany grafem możliwe jest modyfikowanie logiki działania systemu działającego na naszym oprogramowaniu niemalże w czasie rzeczywistym – wystarczy podmienić graf bez konieczności pisania kolejnych setek linijek kodu.

Dzięki grafom zwiększamy także dopasowanie i niezawodność naszego oprogramowania, ponieważ projektowanie funkcjonalności jest zrozumiałe dla zamawiającego i możliwe do modyfikacji nawet na etapie wdrożenia, kiedy to czasami okazuje się, że w praktyce pewne procesy należy zaprojektować inaczej, niż było to zakładane wcześniej.

Dzięki takiemu podejściu oprogramowanie Indigo wyróżnia wyjątkowa elastyczność rozumiana jako możliwość szybkiego wprowadzania zmian, otwartość na integrację z nowymi technologiami oraz możliwość dodawania nowych funkcjonalności poprzez konfigurację naszego frameworku bez konieczności tworzenia dodatkowego oprogramowania.

Słowem w pełni realizujemy postulaty metodologii rapid application development (RAD).

Jesteśmy przekonani, że tylko takie podejście pozwala w pełni mówić o Przemysle 4.0. Dlatego też w odróżnieniu od IIoT, rozumianego jedynie jako zbieranie i analizę danych z urządzeń, nazywamy naszą technologię Control over IIoT – ColloT.

Gdzie można zastosować technologię Indigo?

Gdzie można zastosować technologię Indigo?

Elastyczność Indigo daje możliwości zastosowania frameworku w przeróżnych obszarach. Nasza technologia sprawdzi się wszędzie tam, gdzie jest potrzeba zbierania dużej ilości danych z wielu urządzeń, ich analizowania oraz reagowania na zebrane dane. Niezależnie od tego, czy mowa o integrowaniu maszyn w fabryce, systemów automatyki budynkowej czy też różnych modułów sprzętowych w ramach jednego urządzenia (np. vendomat) zawsze dopasujemy się do indywidualnych potrzeb.

Obszary zastosowań:

- Zbieranie danych, analiza i sterowanie automatyką przemysłową.
- Inteligentne Systemy Transportowe (zbieranie danych, sterowanie, centrum zarządzania).
- Tworzenie zaawansowanych usługomatów i ich sieciowanie (maszyny vendingowe, car/bike sharing etc.).
- Systemy Zarządzania Budynkami (integracja systemów budynkowych, sterowanie automatyką).
- Obsługa procesów logistycznych (pobór opłat, sprzedaż, zliczanie, centra usługowe, optymalizacja procesów).
- Systemy lojalnościowe i dostępowe (skipassy, kontrola dostępu i przejazdu, karty lojalnościowe i zakupowe).
- Metering (energia elektryczna, gaz, woda, inne).

Przemysł 4.0

Naszym zdaniem jedną z najważniejszych inspiracji współczesności jest różnorodność. To różnorodność najbardziej stymuluje do tworzenia nowych technologii i do zmiany podejścia do sposobu wytwarzania oprogramowania.

W obszarze technologii dotyczy ona maszyn i urządzeń oraz systemów informatycznych. Integracja zróżnicowanej infrastruktury pozwala na kompleksową analizę i zarządzanie danymi, umożliwiając optymalizację oraz tworzenie nowych rozwiązań i źródeł dochodu. Unifikacja jest głównym trendem współczesności oraz podstawowym postulatem Przemysłu 4.0.

Globalna cyfrowa transformacja coraz śmielej wkracza do sektora przemysłowego. Rosnące zapotrzebowanie na narzędzia umożliwiające optymalizację produkcji i procesów biznesowych zainspirowało nas do stworzenia pierwszego produktu w oparciu o nasz framework – oprogramowania umożliwiającego szybkie tworzenie dedykowanych systemów pozwalających na integrację maszyn i systemów IT w celu wdrażania praktyk Lean Manufacturing, czyli opartego o informacje nowoczesnego zarządzania przedsiębiorstwem produkcyjnym. Nazwaliśmy je Stabilis.



Stabilis to produkty służące do optymalizacji produkcji, m.in. w zakresie Just in Time, TPM, TQM oraz wspierające Kaizen, które zwiększają produktywność i zmniejszają zapotrzebowanie na personel.