

POLITECHNIKA WROCŁAWSKA  
WYDZIAŁ ELEKTRONIKI

---

KIERUNEK: Automatyka i Robotyka (AIR)  
SPECJALNOŚĆ: Technologie informacyjne w systemach automatyki (ART)

PRACA DYPLOMOWA  
INŻYNIERSKA

System elektroniczny wspomagający  
kompletację zamówień w przedsiębiorstwie  
produkującym okna

An electronic system supporting completing  
parts in a window manufacturing company

AUTOR:  
Filip Kamiński

PROWADZĄCY PRACĘ:  
DR INŻ. JAROSŁAW PEMPERA, K-8

OCENA PRACY:

---

WROCŁAW 2017

# SPIS TREŚCI

<b>Wprowadzenie .....</b>	<b>3</b>
<b>Rozdział 1. Cel i zakres pracy .....</b>	<b>4</b>
<b>Rozdział 2. Opis środowiska produkcyjnego .....</b>	<b>5</b>
<b>Rozdział 3. Identyfikacja miejsc przechowywania .....</b>	<b>11</b>
<b>Rozdział 4. Baza danych systemu .....</b>	<b>14</b>
<b>Rozdział 5. Aplikacje dedykowane do systemu .....</b>	<b>18</b>
<b>5.1. Aplikacja na stanowisku okuwania .....</b>	<b>19</b>
<b>5.1.1. Opis działania aplikacji .....</b>	<b>20</b>
<b>5.1.2. Graficzny interfejs użytkownika .....</b>	<b>22</b>
<b>5.1.3. Korekcja wprowadzanych danych .....</b>	<b>22</b>
<b>5.1.4. Uwierzytelnianie dostępu do bazy danych.....</b>	<b>24</b>
<b>5.1.5. Buforowanie danych. Odporność aplikacji na awarię zasilania.....</b>	<b>25</b>
<b>5.1.6. Udogodnienia dla pracowników.....</b>	<b>26</b>
<b>5.2. Aplikacja na stanowisku montażu .....</b>	<b>27</b>
<b>5.2.1. Opis działania aplikacji .....</b>	<b>28</b>
<b>5.2.2. Graficzny interfejs użytkownika .....</b>	<b>28</b>
<b>5.2.3. Korekcja wprowadzanych danych .....</b>	<b>29</b>
<b>5.2.4. Uwierzytelnianie dostępu do bazy danych.....</b>	<b>29</b>
<b>5.2.5. Udogodnienia dla pracowników.....</b>	<b>30</b>
<b>Rozdział 6. Możliwości rozszerzania systemu .....</b>	<b>31</b>
<b>Rozdział 7. Podsumowanie.....</b>	<b>33</b>
<b>Bibliografia .....</b>	<b>35</b>

## Wprowadzenie

Wiele przedsiębiorstw produkcyjnych zмага się w dzisiejszych czasach z problemami udoskonalania swoich linii montażowych w celu zwiększenia możliwości przerobowych, względnie zminimalizowania kosztów produkcji. Proces racjonalizacji wiąże się często z próbami usuwania lub zmniejszania znaczenia najsłabszych ognisk łańcucha produkcyjnego.

Optymalizacja zadań wyszukiwania odpowiednich komponentów, czy kompletacji całych zamówień przy możliwie najmniejszej ingerencji w infrastrukturę linii produkcyjnej zakładu oraz pracę osób zatrudnionych na newralgicznych stanowiskach stanowi duże wyzwanie i wiąże się z potrzebą zastosowania nieszablonowych metod. Zniwelowanie wpływu tzw. „wąskiego gardła” determinującego sprawność procesu produkcyjnego przedsiębiorstwa wymaga często wdrożenia elektronicznych systemów wspomagających.

Bardzo ważnym elementem takich rozwiązań jest ich kompatybilność z funkcjonowaniem poszczególnych sektorów zakładu, uwzględnienie całego procesu produkcji, zharmonizowanie potrzeb pracowników z warunkami panującymi na ich stanowiskach.

Praca stanowi propozycję rozwiązania problemu optymalizacji kompletacji zamówień w przedsiębiorstwie zajmującym się produkcją okien i drzwi – BUDVAR Centrum Sp. z o.o. Postępowania w niej opisane zostały zaakceptowane w pełni przez kierownictwo zakładu, a gotowe rozwiązanie - w postaci elektronicznego systemu - wdrożono w cykl produkcyjny i otrzymano zadowalające efekty.

Do stworzenia wspomnianego systemu i dedykowanych do niego aplikacji wykorzystano dostępne w zakładzie materiały i infrastrukturę w taki sposób, aby w możliwie największym stopniu wykorzystać ich funkcjonalności do zmaksymalizowania potencjału produkcyjnego. Takie postępowanie nie spowodowało żadnego obciążenia dla budżetu przedsiębiorstwa, a wydatnie polepszyło jego możliwości przerobowe.

Zadbano również o to, by w przypadku dalszego rozwoju zakładu, zwiększenia powierzchni produkcyjnej i dostawieniu kolejnych stacji roboczych, system mógł być z powodzeniem na nich implementowany i by mógł pokryć niezbędną infrastrukturę produkcyjną, a jednocześnie nadal spełniał wymagania personelu korzystającego z systemu każdego dnia pracy.

## Rozdział 1. Cel i zakres pracy

Celem pracy było zmniejszenie wpływu „wąskiego gardła” na możliwości produkcyjne przedsiębiorstwa poprzez zaprojektowanie i wdrożenie elektronicznego systemu wspomagającego kompletację komponentów zamówień. System ten miał usprawnić pracę pewnego sektora hali produkcyjnej – przyspieszyć wyszukiwanie skrzydeł pasujących do ram.

Zrealizowanie celu pracy wymagało:

- zapoznania się z:
  - warunkami panującymi na hali produkcyjnej – umiejscowieniem stanowisk i sprzętu dla pracowników,
  - procesem produkcyjnym przedsiębiorstwa – cykl pracy, linia montażowa,
  - Zakładowym systemem kodowania elementów zamówienia – znaczenie i umiejscowienie kodów kreskowych nadawanych ramom i odpowiadającym im skrzydłom.
- stworzenia systemu identyfikacji slotów w stojakach na skrzydła okienne i wprowadzenie go w odpowiednie miejsca hali produkcyjnej,
- stworzenia systemowej bazy danych, w której tabelach przechowywane byłyby dane o miejscach, w których znajdują się skrzydła dedykowane dla danej ramy,
- zaimplementowanie aplikacji działających na wszystkich stanowiskach sektora stanowiącego wąskie gardło.

W fazie planowania projektu należało uwzględnić:

- uodpornienie systemu na utratę danych w przypadku awarii zasilania,
- dostosowanie się do panujących w firmie standardów oznaczeń produktów i miejsc ich składowania ,
- stworzenie przyjaznego dla pracowników interfejsu graficznego aplikacji,
- wykorzystanie narzędzi, jakimi dysponują pracownicy na swoich stanowiskach takich jak: skanery kodów kreskowych, komputery z systemem Windows 7 Embedded, monitory dotykowe.

## Rozdział 2. Opis środowiska produkcyjnego

W dzisiejszych czasach wiele przedsiębiorstw produkcyjnych zmagają się z problemami możliwości wytwórczych swoich linii montażowych. Jednym z powszechniej występujących problemów jest tak zwany „problem wąskiego gardła”.

*„Wąskim gardłem nazywamy takie ogniwo, które w danym ciągu technologicznym ma najmniejszą zdolność produkcyjną i limituje wielkość produkcji, jaką może wytwarzać cały łańcuch powiązanych ze sobą stanowisk.”<sup>1</sup>*

Oznacza to, że pewne miejsce – ogniwo łańcucha produkcyjnego determinuje możliwości przerobowe fabryki. Gdyby zostało ono odpowiednio zagospodarowane i zmodyfikowane wydajność linii montażowej znacząco by się podniosła.

Powstawanie takiego elementu stymulowane jest zazwyczaj brakiem pewnych zasobów, odpowiednich technologii lub niedoborem pracowników.

Bardzo istotne dla prawidłowego funkcjonowania zakładu jest zlokalizowanie takiego ogniwa oraz podjęcie działań, by możliwie ograniczyć, a najlepiej zupełnie wyeliminować, jego niepożądany wpływ na cykl produkcji. W związku z przyczynami powstawania „wąskiego gardła” działania, jakie można podjąć w celu jego eliminacji skupiają się wokół:

- zwiększenia liczby pracowników na newralgicznych stanowiskach,
- zreorganizowania, zmiany infrastruktury technicznej, urządzeń hali produkcyjnej (samego miejsca lub całego łańcucha produkcyjnego),
- zainwestowania w nowe technologie usprawniające dane miejsce systemu produkcji.

Pierwsze dwie kategorie działań należą do trudnych do zrealizowania w wielu zakładach produkcyjnych. Zwiększenie liczby zatrudnionych wiąże się z zainicjalizowaniem procesu rekrutacyjnego nowych pracowników przeszkoleniem i wdrożeniem w procedury firmowe pozytywnie zweryfikowanych kandydatów – co zazwyczaj jest nie tylko kosztowne dla przedsiębiorstwa, znacząco obciąża budżet firmy, ale jest również czasochłonne i nie zawsze kończy się pomyślnym rezultatem. Nawet najlepsze przeszkolenie pracowników nie daje gwarancji poprawnego wykonywania przez nich zadań.

---

<sup>1</sup> „Podstawowe zagadnienia zarządzania produkcją”, Liwowski B., Kozłowski R., Oficyna Ekonomiczna, Kraków 2006

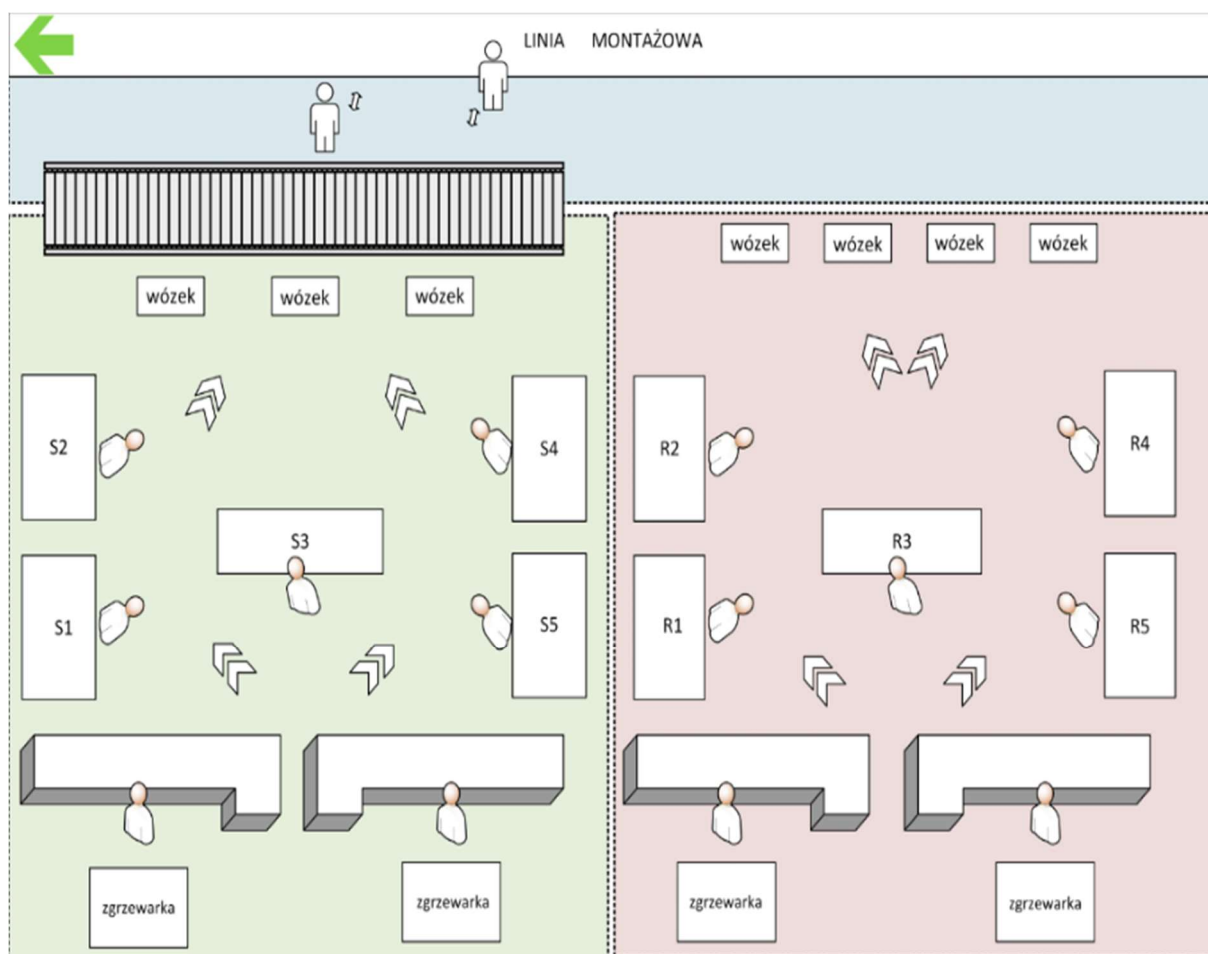
Reorganizacja hali produkcyjnej wymaga natomiast wstrzymania całego łańcucha zadań na stanowiskach poprzedzających i następujących, co spowoduje przestoje produkcyjne, uniemożliwi zrealizowanie zamówień i spowoduje duże straty finansowe dla przedsiębiorstwa ze względu na niewykonanie zadania czy nawet obowiązek zapłaty odszkodowań. W dodatku każda taka reorganizacja może wiązać się niezamierzonym przesunięciem „wąskiego gardła” w inne miejsce łańcucha produkcyjnego.

Skupienie się na zwiększeniu potencjału przerobowego stanowiska, które stanowi „wąskie gardło”, inwestowanie w technologie wspomagające je przynosi zazwyczaj bardzo dobre i stosunkowo szybkie efekty. Szczególnie, jeśli wdrażane rozwiązania opracowane zostały wewnętrznie, z uwzględnieniem preferencji pracowników, ich nawyków i przyzwyczajeń, z dbałością o wykorzystanie dostępnych urządzeń i jak najmniejszą ingerencję w dotychczasowe działania na hali produkcyjnej. Takie rozwiązania wcale nie muszą być kosztowne, a gwarantują ściśle dopasowanie do potrzeb i warunków zakładu produkcyjnego bez dezorganizacji czy zakłóceń pracy przedsiębiorstwa.

Dodatkowym atutem takich rozwiązań jest fakt, że przed ich wdrożeniem można przeanalizować wszelkie możliwe konfiguracje sprzętowe wraz z ewentualnym oprogramowaniem, ich wpływ na pracę poszczególnych działów, sektorów hali oraz, co niezwykle ważne, przeprowadzić symulację wyników działań tych rozwiązań na każdym etapie wdrażania – również w ujęciu całościowym, dla całego zakładu, by później skonfrontować je z rzeczywistością i dokonać ewentualnych korekt.

Problem wąskiego gardła w przedsiębiorstwie Budvar Centrum Sp. z o.o. występował praktycznie w połowie łańcucha produkcyjnego.

Na Rys.1 przedstawiono schemat sektora hali produkcyjnej, który determinował potencjał produkcyjny zakładu.



Rysunek 1. Schemat sektora hali produkcyjnej

**Legenda:** „S<sub>x</sub>” symbolizują stanowiska pracowników okuwających skrzydła,

„R<sub>x</sub>” symbolizują stanowiska pracowników okuwających ramy,

**Podwójne groty** wskazują kierunek przechodzenia materiału przez sektor,

**Obustronne strzałki** oznaczają ruch pracowników montażu, charakterystyczny dla ich stanowiska,

**Zielona strzałka** wskazuje kierunek przemieszczania się wyrobów w cyklu produkcyjnym.

Do strefy zgrzewarek dostarcza się odpowiednio przycięte wcześniej elementy, które pod wpływem wysokiej temperatury oraz właściwego ułożenia utworzą kolejne egzemplarze skrzydeł lub ram okiennych. Gdy proces zgrzewania oraz wstępnej obróbki wytworzonych obiektów – czyszczenia i wygładzania niedoskonałości – zostanie zakończony, wówczas pracownicy okuwający zabierają gotowy obiekt i przenoszą go na swoje stanowisko.

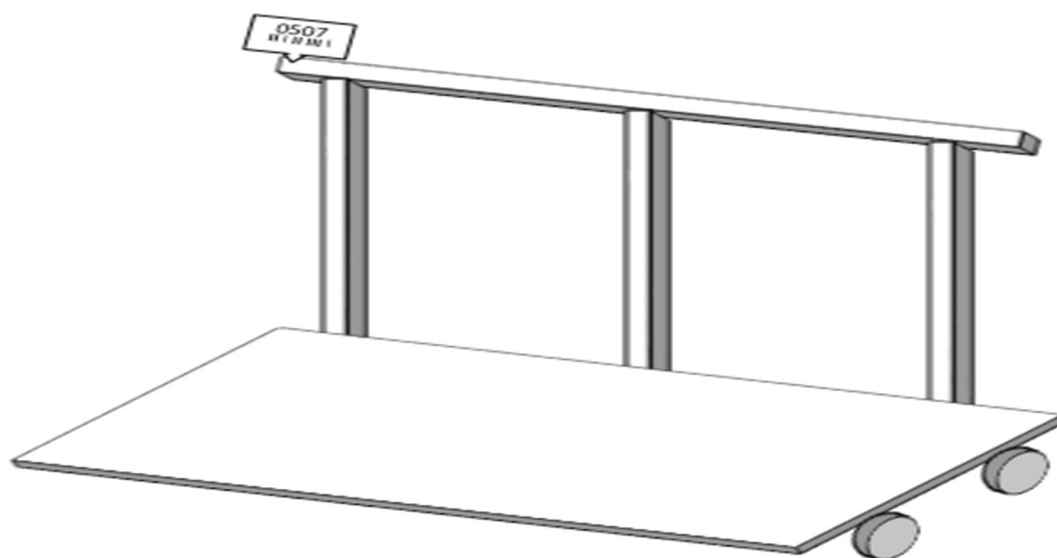
Następnie dane skrzydło bądź ramę poddaje się zabiegom rozmieszczania zaczepów, elementów ryglujących, mocowania zawiasów, nawiercania otworów na klamki oraz docinania zasuwnic i ramion rozwórek.

Tak przygotowane ramy przenoszone są na linię montażową, a skrzydła odstawia się do specjalnie do tego celu przeznaczonych miejsc przechowywania – stojaków lub wózków, znajdujących się nieopodal stanowisk obróbki.

Po drugiej stronie, pracownicy zatrudnieni na stanowisku montażu wyszukują odpowiadające ramom skrzydła i montują je razem tworząc kolejny wyrób. To właśnie w tym miejscu zanotowano największe przestoje produkcyjne. Nieklasyfikowane i nieusystematyzowane odkładanie i przechowywanie skrzydeł w miejscach przeznaczenia powodowało, że pracownicy działu montażu poświęcali bardzo dużo czasu na odnalezienie skrzydeł przydzielonych do danej ramy zwyczajnie chodząc wzdłuż stojaków o 80 slotach i wózków – których liczba wzrastała, gdy stojaki się zapelniały (każdy wózek mieścił do 25 skrzydeł) – i wyszukując żadanego wyrobu po kodzie kreskowym. Wiązało się to z porównywaniem ze sobą często podobnych produktów, wyjmowaniem ich z miejsca przechowywania, przypasowaniem do ramy i ponownym odkładaniem w razie pomyłki.

Należało zatem znaleźć rozwiązanie, które spowodowałoby uporządkowanie odkładania i przechowywania skrzydeł oraz przyspieszenie ich wyszukiwania i dopasowania do odpowiadających ram. Projektując takie rozwiązanie należało również wprowadzić system identyfikowania miejsc przechowywania.

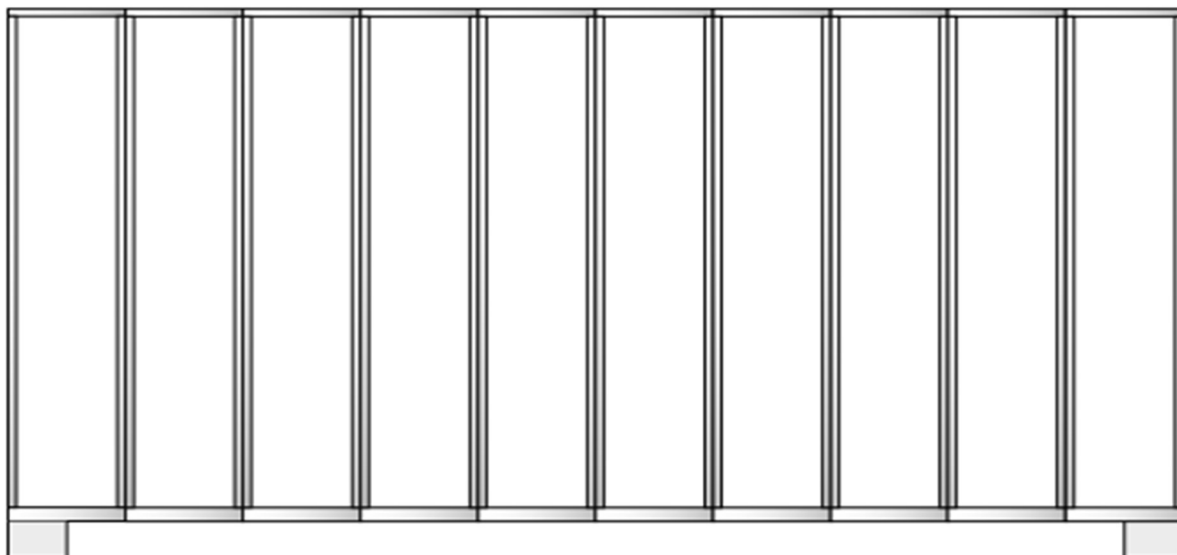
Do tej pory w przedsiębiorstwie uwzględniano charakterystyczne oznakowanie wózków.



*Rysunek 2. Szkic wózka z etykietą identyfikacyjną*

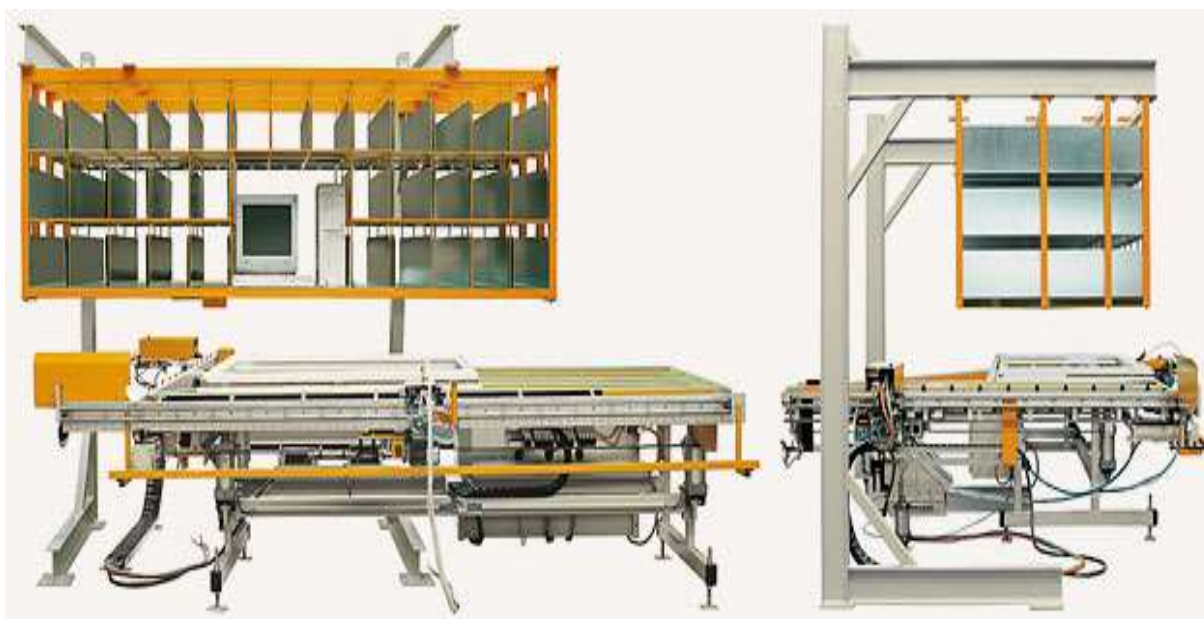
Natomiast przegrody stojaków pozostawały bez jakiegokolwiek oznakowania. Taka sytuacja panowała na całym terenie zakładu, na wszystkich sektorach hali ze stanowiskami okuwania.





*Rysunek 3. Szkic stojaka na okute skrzydła*

W związku z ograniczoną przestrzenią, jaką dysponowali pracownicy na swoich stanowiskach, należało również skupić się na możliwie maksymalnym wykorzystaniu urządzeń, jakich już używali, by nie zabierać dodatkowo miejsca przy stołach.



*Rysunek 4. Przykładowe stanowisko okuwania skrzydeł<sup>2</sup>*

<sup>2</sup> Źródło: [http://afs-federhenn.de/inhalte/produkt/fluegel/4654\\_FAA%20S2/faa%20s2.html](http://afs-federhenn.de/inhalte/produkt/fluegel/4654_FAA%20S2/faa%20s2.html)



*Rysunek 5. Przykładowe stanowisko do okuwania skrzydeł i skaner kodów kreskowych<sup>3</sup>*

Wszystkie elementy projektowanego systemu musiały dodatkowo być zrozumiałe, intuicyjne dla pracowników, a także ergonomiczne, nie narażające ich na utratę zdrowia, nie pogarszające ich warunków pracy oraz nie narażające ich na dyskomfort.

Wobec powyższych zdecydowano się na wprowadzenie elektronicznego systemu wspomagającego kompletację zamówień, obejmującego:

- dwie dedykowane dla stanowisk okuwania i montażu aplikacje,
- bazę danych przechowywanych skrzydeł, ich miejsc, daty i godziny wprowadzenia ich do danego miejsca,
- system identyfikacji miejsc przechowywania
- zabezpieczenie systemu przed utratą danych spowodowaną awarią zasilania w danym sektorze fabryki.

Wszystkie elementy projektowanego systemu opisane zostały w kolejnych rozdziałach.

---

<sup>3</sup> Źródło: <http://fobra.pl/category/bez-kategorii/page/9/>

### Rozdział 3. Identyfikacja miejsc przechowywania

Niewątpliwie najbardziej efektywnym i atrakcyjnym dla oka rozwiązaniem problemu identyfikacji miejsc byłoby zastosowanie dodatkowego mikrokontrolera przy stojakach, który sterowałby elektroniką oświetleniową. Wówczas możliwa byłaby pewna wizualizacja danych. Nad wyszukiwanym przez pracownika montażu skrzydłem, przetrzymywanym w danym stojaku, mogłaby się zaświecić kontrolka, która ułatwiałaby odnalezienie skrzydła pracownikowi. Największą zaletą takiego rozwiązania byłaby oczywiście wygoda. Taka kontrolka widoczna by była z wielu miejsc hali produkcyjnej.

Jednak ze względu na specyficzne warunki panujące na hali produkcyjnej zrezygnowano z tego pomysłu. Niemożność zainstalowania dodatkowego okablowania ani – ze względów bezpieczeństwa – zastosowania standardu Wi – Fi, bardzo prawdopodobne duże zanieczyszczenie drobnymi odpadami produkcyjnymi, możliwe wymiany stojaków lub wózków między sektorami spowodowały odrzucenie tego rozwiązania.

W celu jak najmniejszej ingerencji w zakładowe zwyczaje zdecydowano, że najlepszym rozwiązaniem do oznaczenia miejsc przechowywania skrzydeł będzie zastosowanie obecnego systemu identyfikacji wózków i przeniesienie go również, z drobną modyfikacją, na stojaki.

Schemat oznaczeń na wózkach prezentował się następująco:

@X<sub>1</sub>X<sub>2</sub>X<sub>3</sub>X<sub>4</sub>X<sub>5</sub>X<sub>6</sub>X<sub>7</sub>, co po zakodowaniu przekładało się na graficzną:



*Rysunek 6. Kod kreskowy wózka oznaczonego jako: @1000965*

gdzie najważniejszą rolę odgrywał charakterystyczny pierwszy znak oraz ostatnie 4 znaki. Te one umieszczane były w widocznych miejscach na wózkach – malowane na uchwytych i metalowych szkieletach, doklejane na plastikowych plakietch.

Modyfikacja polegała na zwiększeniu ilości zakodowanych w postaci kreskowej znaków o jedną cyfrę. Dzięki temu zabiegowi, uniknięto ewentualnych pomyłek przy wyszukiwaniu skrzydeł w miejscach oznaczonych identycznym numerem. Choć wartość liczbową pozostaje taka sama, to przez tę drobną zmianę, system będzie w stanie rozróżnić np. stojak oznaczony numerem 55 od wózka o takim samym numerze.

Zatem nowe oznaczenia dla stojaków przybrały postać:

@X<sub>1</sub>X<sub>2</sub>X<sub>3</sub>X<sub>4</sub>X<sub>5</sub>X<sub>6</sub>X<sub>7</sub>X<sub>8</sub>

Aby ułatwić pracownikom identyfikację przegrody i przechowywanego w niej skrzydła postanowiono wydrukować plakietki na specjalnym, posrebrzonym materiale i umieścić je w miejscu dostępnym wygodnie dla wszystkich pracowników. Materiał ten, odporny na przetarcia, to świetne zabezpieczenie systemu identyfikacji przed uszkodzeniami mechanicznymi na długi czas.

Ze względu na konstrukcję stojaków w grę wchodziło zamontowanie plakietek na samej górze – na ożebrowaniu, lub na dolnej, poziomej belce. Aby nie narażać pracowników na dyskomfort ciągłego unoszenia głowy w celu odnalezienia szukanego numeru wybrano drugą opcję i plakietki zamieszczono jak na schemacie przedstawionym na Rys. 7.



*Rysunek 7. Umieszczenie etykiet slotów na stojaku*

Plakietki umieszczane na stojakach po stronie stanowiska montażu zawierały jedynie numer identyfikujący dany slot. Natomiast ich odpowiedniki od strony stanowiska okuwania oprócz numeru – wygodnego dla oka pracownika, ale nie dla aplikacji systemowej – posiadały

jeszcze kody kreskowe, dzięki którym wprowadzanie danych do aplikacji i bazy danych trwało znacznie krócej i przyczyniało się do znacznie mniejszej ilości błędów.

Po konsultacjach z personelem zatrudnionym w sektorze okuwania zdecydowano się na zastosowanie układu jak na Rys. 8.



*Rysunek 8. Przykładowe oznaczenie slotu stojaka*

Taki układ – u góry kod kreskowy, na dole numer slotu – gwarantował wygodę odczytywania i kontrolowania miejsc, w które zostały odkładane skrzydła i jednocześnie nie powodował żadnych problemów przy skanowaniu kodu kreskowego do aplikacji przy dowolnie obranej pozycji pracownika odkładającego okuty produkt. Pracownicy nie musieli się pochylać ani wyciągać nadto rąk, aby skaner poprawnie odczytał zakodowane informacje. Nie byłoby to możliwe, a na pewno wygodne, gdyby zastosowano układ odwrotny – kod kreskowy na dole, a u góry miejsce – czy gdyby zamontowano plakietki na górnej żerdzi stojaków.

## Rozdział 4. Baza danych systemu

Aby pracownicy działu montażu mogli skutecznie odczytywać położenie skrzydeł odstawionych w odpowiednie miejsca przez pracowników działu okuwania należało przetrzymywać gdzieś informacje zarówno o samym kodzie kreskowym skrzydła, jak również o miejscu jego przechowywania. W tym celu stworzona została systemowa baza danych – jako jedna z instancji firmowej bazy.

Do jej opracowania wykorzystano narzędzie, jakim był system do administrowania bazami danych firmy Microsoft – MS SQL w wersji 2014 odznaczający się dużą wydajnością i niezawodnością. Zdecydowano się na taki krok, aby nie zwiększać niepotrzebnie wydatków przedsiębiorstwa. System ten był na wyposażeniu zakładu, do dyspozycji dla upoważnionych pracowników, z zapewnionym wsparciem technicznym. Za jego pomocą zbierano wszelkie informacje niezbędne dla funkcjonowania firmy, a później poddawano te informacje analizie, obróbce, przekazywano je dalej.

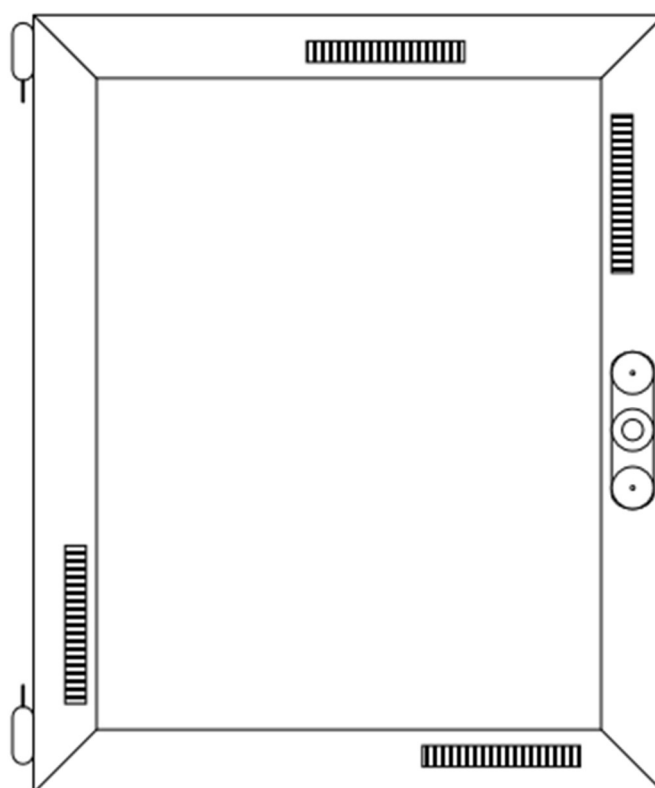
Instancją o szczególnym znaczeniu była ta, w której przechowywano wytyczne zamówień – ilość wystąpień danego produktu, numer partii, kody kreskowe zamawianych produktów wraz z ich znaczeniem itp.

Dla tej instancji bazy danych stworzona została procedura. Jako argument przyjmowała ona kod kreskowy skrzydła lub ramy i zwracała pełne informacje o danej pozycji zamówienia, w której ten kod występuje. Oznacza to, że po wprowadzeniu kodu kreskowego jednego z wymienionych produktów następowało przeszukiwanie instancji przechowującej zamówienia pod kątem wystąpienia tego kodu. Jeśli został on znaleziony, procedura zwracała informacje o numerze zamówienia, w którym występuje kod, numerze partii produkcyjnej, w której produkt został zaplanowany, ilości wystąpień pozycji – ilości sztuk do wyprodukowania oraz o parametrze nazwanym w firmie „barcodem”. Parametr ten odgrywał ogromną rolę w przyporządkowywaniu, łączeniu ze sobą kilku półproduktów w jeden większy obiekt.

W przedsiębiorstwie Budvar Centrum Sp. z o.o. wszystkie półprodukty stolarki okiennej – drzwiowej przechodzące przez łańcuch produkcyjny tworzone były z przygotowanych, pociętych wcześniej płacht materiału – aluminium lub PCV. Wszystkim elementom tworzącym skrzydło lub ramę, jeszcze przed ułożeniem ich na zgrzewarkach, nadawano charakterystyczny, niepowtarzalny kod kreskowy. Oznacza to, że gotowy produkt stolarki – skrzydło albo rama okienna – mógł posiadać przynajmniej cztery takie kody. Każde urozmaicenie w postaci dodatkowej poprzeczki, belki oznaczało dodatkowy kod.



Rysunek 9. Przykładowy kod kreskowy skrzydła



Rysunek 10. Możliwe konfiguracje kodów kreskowych skrzydła

„Barcode” stanowił pewnego rodzaju łącznik wszystkich tych elementów. Każdy z nich, oprócz kodu kreskowego, posiadał w odpowiedniej instancji bazy danych identyczny atrybut jak ten, który został przypisany do każdej pozycji z zamówienia. W skrócie – elementy tworzące ramę i skrzydło mają taki sam „barcode” – przydzielony do danego skrzydła lub ramy w zamówieniu – choć ich kody kreskowe są różne. Teoretycznie, skoro każda rama ma przypisane do siebie skrzydło o określonych kodach kreskowych, nie powinna mieć miejsca sytuacja, w której montowane są ze sobą półprodukty z dwóch różnych zamówień.

Rzeczywistość jednak prezentowała się inaczej. Pracownicy zatrudnieni na stanowisku montażu, aby nie tracić jeszcze więcej czasu, gdy tylko znaleźli skrzydło zgrywające się z daną ramą, po prostu mocowali ze sobą te dwa komponenty, nie zważając na to, czy należą one do tego samego zamówienia, czy nie. Taką sytuację należało oczywiście zmienić i przywrócić porządek w systemie fakturowym.

Do realizacji tego oraz wcześniej wymienionych celów zdecydowano stworzyć prostą instancję bazy danych, czytelną dla administratorów i prostą w obsłudze, o postaci:

- dwóch tabel, po jednej dla każdego typu miejsca służącego do przechowywania okutych skrzydeł, o nazwach zgodnych z nazwami tych miejsc,
- każdej tabeli zawierającej:
  - kod kreskowy skrzydła,
  - numer miejsca,
  - barcode,
  - datę i godzinę wprowadzenia rekordu do bazy – w celach porządkowych.

Ponadto w bazę danych wbudowane zostały wyzwalacze odpowiadające za:

- przydzielenie każdemu wpisowi – rekordowi – daty, godziny, minuty i sekundy wprowadzenia go do bazy,
- usuwanie przestarzałych wpisów w bazie danych, jeśli są one w bazie dłużej niż miesiąc.

Przydzielanie dat i godzin wpisom w momencie ich wprowadzania do bazy danych było zdecydowanie wygodniejsze do implementacji. Poza tym eliminowało dodatkowe obciążenie pamięci komputera znajdującego się na stanowisku roboczym oraz usunęło możliwość pojawienia się „chaosu czasowego” – nie można było założyć, ani sprawdzić, że żaden z używanych na hali produkcyjnej komputerów nie miał zmienionego czasu startowego procesora. Zastosowane rozwiązanie – dodawanie daty i czasu za pomocą wyzwalacza – okazało się rozwiązaniem estetycznym i praktycznym, przyniosło zamierzone efekty, wobec czego zdecydowano się je pozostawić.

Aby nie zaśmiecać bazy danych przestarzałymi wpisami, a tych, biorąc pod uwagę możliwość produkcyjne zakładu, mogło się nagromadzić po pewnym czasie sporo, zdecydowano się ustawić kolejny wyzwalacz w bazie danych, który byłby odpowiedzialny za usuwanie tego nadmiaru. Maksymalną długość życia pojedynczego rekordu ustalono razem z kierownictwem zakładu – sprawa wymagała konsultacji z osobami decyzyjnymi i znającymi



problematykę długości realizacji zamówień – biorąc pod uwagę przerwy weekendowe, świąteczne i najmniej korzystną ich konfigurację ze sporym zapasem, na 30 dni.

Oba wyzwalacze zaprogramowano tak, aby uruchamiały się z każdym nowododawanym do bazy danych wpisem. Przy każdym wprowadzaniu pakietu danych zostawała do niego dodawana data i godzina wprowadzenia oraz skanowane były obie tabele bazy pod kątem wartości w kolumnie daty. Jeśli ta wartość przekracza długość 30 dni licząc od momentu świeżo wprowadzonego wpisu, to cały rekord, którego częścią jest ta wartość, zostawał usuwany.

Organizacja operacji przeprowadzanych na bazie danych zostanie opisana w następnym rozdziale z powodu zaszczytu w wewnętrznej strukturę programów kodu źródłowego odpowiedzialnego za realizację działań pracowników – dodawanie informacji do bazy, wyszukiwanie danych, usuwanie wpisów z bazy.

## Rozdział 5. Aplikacje dedykowane do systemu

Elementami spajającym działania na hali produkcyjnej z serwerem, na którym umieszczona została systemowa baza danych, zostały aplikacje działające na każdym stanowisku do okuwania skrzydeł oraz montażu skrzydeł i ram. Ze względu na zupełnie inne zadania, które należało wykonywać po obu stronach stojaków do przechowywania skrzydeł zdecydowano, że powstaną dwie aplikacje, ściśle dedykowane do problemu, zamiast jednej, uniwersalnej, ale której walorów nie sposób będzie w pełni wykorzystać na żadnym ze stanowisk.

Do wykonania obu aplikacji wykorzystano środowisko Qt Creator dla framework'a Qt 5.6.2 dedykowanego do języka C++. Na wybór tego zestawu narzędzi zdecydowano się już na samym początku pracy nad systemem ze względu wiele jego zalet.

Dużą rolę odgrywała w tym wypadku możliwość skorzystania z wbudowanego w Qt Creator modułu umożliwiającego łatwe i wygodne tworzenie interfejsu graficznego dla programu. Konstruowanie takiego komponentu aplikacji za pomocą przeciągania odpowiednich bloków, umieszczania ich w żądanym miejscu, modyfikowania ich korzystając z zakładki dołączonego formularza jest zdecydowanie lepszym rozwiązaniem od tworzenia go pod postacią kodu, kompilacji i jego ciągłej edycji i ponownej kompilacji, by osiągnąć wymagany efekt.

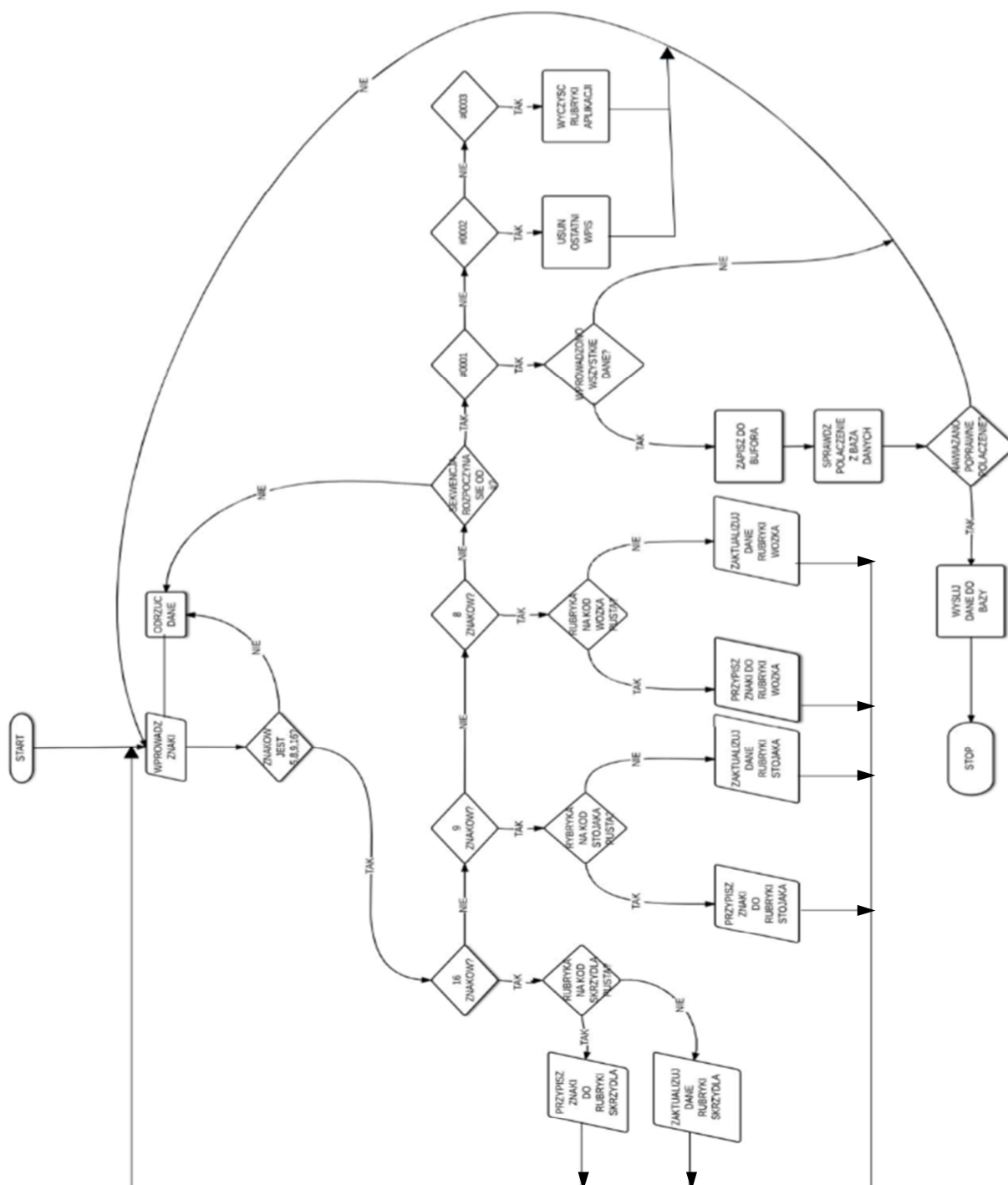
Rozbudowana dokumentacja i łatwość znalezienia w niej niezbędnych informacji stanowiła kolejny atut. Wyszukiwanie informacji mogło odbywać się lokalnie – za pomocą zbioru dokumentów dołączonych podczas instalacji środowiska lub za pomocą strony internetowej Qt.

Konieczność stworzenia bazy danych w systemie MS SQL Server uwarunkowywała korzystanie z takiego środowiska, które umożliwia komunikację z tym narzędziem. Świetny w tym wypadku okazał się właśnie Qt ze swoimi rozwiązaniami, wspierającymi obsługę sterowników do wielu systemów bazodanowych (m. in.: MySQL, SQLite, Oracle i wielu innych).

Środowisko to wybrano również ze względu na rozbudowane możliwości mechanizmu posługiwania się sygnałami i slotami, wsparciem wielowątkowości, które nie powodują zwiększonego zużycia zasobów procesora i pozwalają w pełni korzystać z dobrodziejstw szybkości wykonywanych obliczeń, przeprowadzanych za pomocą języka C++.

## 5.1. Aplikacja na stanowisku okuwania

Na Rys. 11 zaprezentowany jest schemat blokowy działania aplikacji.



Rysunek 11. Schemat blokowy działania aplikacji dla stanowiska okuwania

### 5.1.1. Opis działania aplikacji

Ten program miał zostać wdrożony na stacjach roboczych obejmujących fragment sektora pomiędzy zgrzewarkami, a stojakami i wózkami. Głównym zadaniem aplikacji miało być zbieranie danych wprowadzonych przez użytkowników (ręcznie lub przy pomocy skanera), przechowywanie ich w buforze danych i wysyłanie do bazy danych, jeśli istniała taka możliwość – jeśli poprawnie nawiązano połączenie.

Wprowadzane przez użytkowników dane kierowane są w wyznaczone w interfejsie graficznym miejsce. Program rozpoznaje długość wprowadzanego ciągu znaków. Jeśli sekwencja nie pasuje do jednego z przyjętych standardów – nie liczy 5, 8, 9 lub 16 znaków – jest odrzucana, a interfejs graficzny przywracany jest do pierwotnego stanu.

Łańcuch 16 znaków, jeśli te są liczbami – rozpoznawany jest przez program jako kod skrzydła. Gdy po tej sekwencji program napotka na ślad przypisania wartości, przejścia do nowej linii – naciśnięcie przycisku ENTER na klawiaturze (w przypadku użycia skanera kodów kreskowych nie ma potrzeby zatwierdzania w taki sposób – skaner sam dodaje tak znak), nastąpi przypisanie wprowadzonej wartości do odpowiedniej zmiennej oraz wyświetlenie jej na ekranie w przewidzianym do tego miejscu. Dzięki temu użytkownicy aplikacji są w stanie stwierdzić, czy wprowadzono kod poprawnego skrzydła i w razie pomyłki mogą dokonać odpowiednich korekt.

Łańcuch 8 lub 9 znaków, o ile rozpoczyna się od znaku specjalnego „@”, a pozostałe znaki są liczbami, zostanie przez aplikację rozpoznany jako kod miejsca (odpowiednio: wózka lub slotu w stojaku). Zatwierdzenie wprowadzonej sekwencji odbywa się identycznie jak w przypadku wprowadzania kodu okutego skrzydła. Po napotkaniu znaku nowej linii program przypisuje rozpoznaną wartość – oczywiście bez znaku specjalnego – do zmiennej w pamięci oraz wyświetla ją w drugiej, przygotowanej do tego celu, komórce interfejsu graficznego, aby użytkownik miał pełną kontrolę nad wprowadzanymi danymi. Jeśli zauważy, że dane wyświetlane przez program są inne niż te, które chciał zatwierdzić, może dokonać korekty.

Sekwencja 5 znaków zarezerwowana jest dla akcji, które można wywołać w programie. Te akcje to: „Zatwierdź”, „Cofnij”, „Reset. Łańcuch znaków musi rozpoczynać się od znaku specjalnego „#”, a następne 4 znaki muszą być liczbami.

Ciąg znaków	Przypisana akcja programu
#0001	Zatwierdź
#0002	Cofnij
#0003	Reset

Tabela 1. Ciągi znaków dla akcji programów



# AKCEPTUJ

Rysunek 12. Kod kreskowy przycisku „Akceptuj”



# COFNIJ

Rysunek 13. Kod kreskowy przycisku „Cofnij”



# RESET

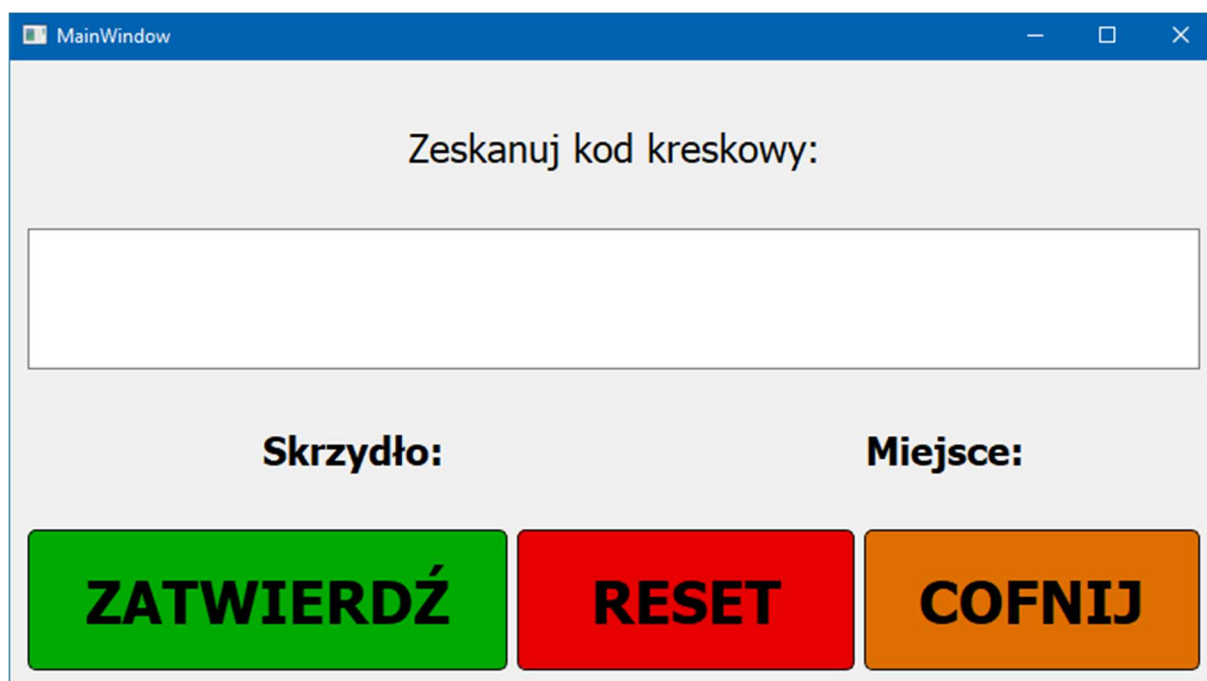
Rysunek 14. Kod kreskowy przycisku „Reset”

Jeśli użytkownik aplikacji stwierdzi pełną zgodność informacji – tzn. że wszystkie dane wyświetlane przez program są takie, jakie chciał, aby zostały wyświetlone może je zatwierdzić, za pomocą przewidzianego przycisku interfejsu graficznego. Wówczas dane zapisywane są według procedury postępowania opisaną szerzej w punkcie 5.1.5. Jeśli pola informacji są puste

– nie wprowadzono żadnych danych, lub dane zostały usunięte – naciśnięcie przycisku nie wywoła żadnej akcji, wygląd interfejsu pozostanie niezmieniony.

### 5.1.2. Graficzny interfejs użytkownika

Na Rys. 15 widnieje okno aplikacji, zaraz po jej wystartowaniu.



*Rysunek 15. Podstawowy widok aplikacji na stanowisku okuwania*

### 5.1.3. Korekcja wprowadzanych danych

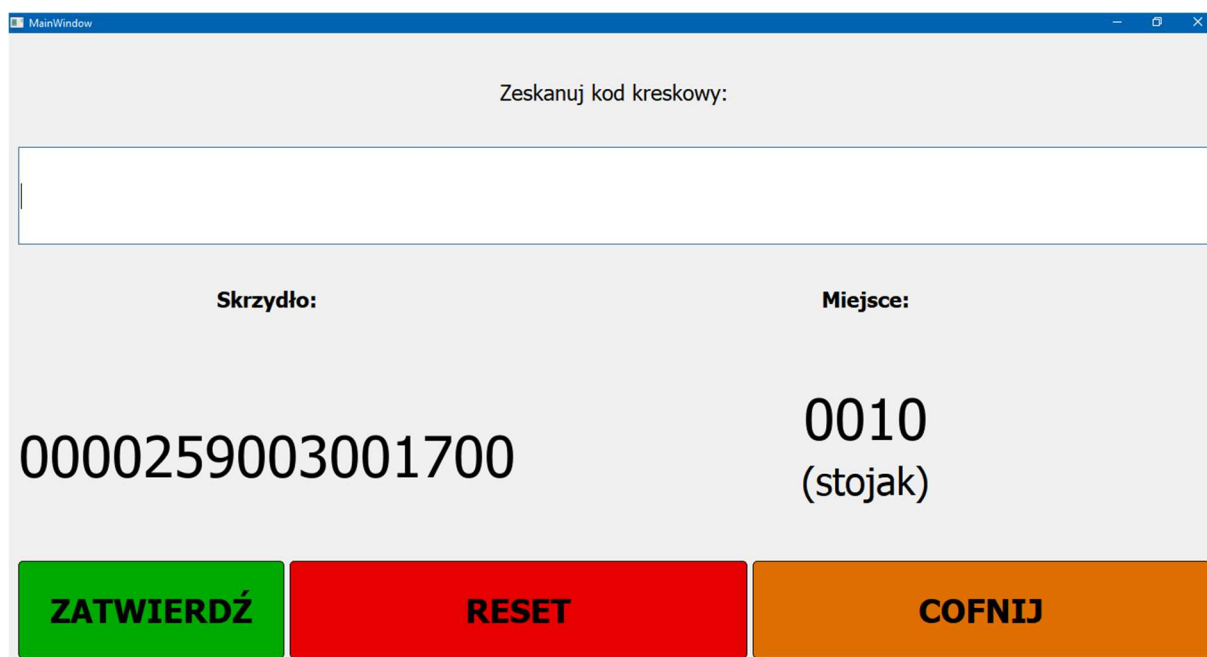
Korekty wprowadzanych danych można dokonać na kilka sposobów. Jednym z nich – najbardziej manualnym – jest wykorzystanie stworzonych do tego celu przycisków. Do korekt przewidziano przyciski o etykietach „Cofnij” i „Reset”.

Naciśnięcie pierwszego z nich uruchamia funkcję, która sprawdza, jaka informacja została wprowadzona przed chwilą. Jeśli wypełnione jest tylko jedno miejsce – „Skrzydło” lub „Miejsce” – wówczas zostaje ono opróżnione i aplikacja wraca do stanu początkowego. Jeśli wprowadzono obie informacje – tzn. o skrzydle i miejscu, w jakim jest przechowywane – to wówczas opróżniana jest rubryka najświeższa, która została wypełniona jako ostatnia – ta starsza pozostaje nietknięta. Dwukrotne naciśnięcie przycisku „Cofnij” wyczyści wszystkie rubryki i przywróci interfejs aplikacji do stanu początkowego.

Działanie przycisku „Reset” jest również bardzo intuicyjne. Naciśnięcie tego przycisku powoduje natychmiastowe wyczyszczenie rubryk z wprowadzanymi danymi, przywraca interfejs do stanu zaraz po uruchomieniu i pozwala użytkownikowi zacząć ponownie proces wprowadzania sekwencji znaków z „czystą kartą”.

Funkcje ukryte pod przyciskami „Cofnij” i „Reset” zostały również odpowiednio zakodowane dla wygody pracowników. Specjalne kody kreskowe zawierające 5 znaków, zeskanowane i wprowadzone do programu powodują uruchomienie tych samych funkcji, co po naciśnięciu przycisku. Etykiety z tymi kodami kreskowymi zostały zamontowane na każdym stanowisku pracowniczym.

Dodatkowo, w programie została przewidziana możliwość korekty danych przez ich aktualizacje – bez potrzeby używania przycisków. Jeśli pracownik zauważy, że którakolwiek informacja wyświetlana na ekranie monitora jest niezgodna z jego zamierzeniami, może po prostu wprowadzić ponownie sekwencję znaków. To znaczy, że może ponownie wpisać symbole z klawiatury lub zeskanować kody kreskowe skrzydła lub miejsca, a program sam podmieni wartości po rozpoznaniu długości łańcucha znaków.



Rysunek 16. Okno aplikacji po wprowadzeniu do niej wszystkich danych

Rysunek 17. Widok aplikacji po użyciu „Cofnij” i wprowadzonym ostatnio miejscu

#### 5.1.4. Uwierzytelnianie dostępu do bazy danych

Autoryzowanie przesyłania informacji ze stanowisk pracowniczych do bazy danych rozwiązano za pomocą umieszczenia danych do logowania w pliku tekstowym i stworzenia odpowiedniej funkcji czytającej z tego pliku w aplikacji. Dane w pliku tekstowym zawsze zapisywane były przez osoby upoważnione, w specjalnym, chronionym różnymi zabezpieczeniami administratorskimi, miejscu na dysku stacji roboczej, w postaci:

login; hasło\_do\_logowania

Rozwiązanie takie było atrakcyjne z tego względu, że uniemożliwiło jednoczesne wysyłanie i pobieranie informacji do i z bazy przez wielu użytkowników z tymi samymi danymi i powstawanie związanych z tym ewentualnych błędów – gdyby dane do połączenia z bazą umieszczone zostały w kodzie źródłowym aplikacji.

Dodatkowo, dzięki takiemu rozwiązaniu problemu, w przypadku chęci rozszerzenia systemu o kolejne stanowiska robocze, nie ma potrzeby ingerowania w kod aplikacji, rekompilacji i zainstalowania programu na nowej stacji. Wystarczy, że administrator uwzględni nowego użytkownika w systemie bazodanowym – nazwę i hasło może ustalić według własnego uznania – i na nowej stacji roboczej utworzy nowy plik z danymi do logowania lub skopiuje plik ze starymi danymi i je zmieni.



### 5.1.5. Buforowanie danych. Odporność aplikacji na awarię zasilania

Aplikacja instalowana na stanowisku do okuwania skrzydeł, w związku z tym, że ma na celu zgromadzenie informacji i przesłanie ich do bazy danych, została wyposażona w system buforowania danych. Użytkownik – operator stanowiska – nie łączy się z bazą od razu po zatwierdzeniu danych wprowadzonych do programu. Po naciśnięciu przycisku „Zatwierdź” lub po zeskanowaniu kodu, który mu odpowiada, wszystkie wprowadzone przez użytkownika dane zapisywane są w pliku tekstowym, tworzone automatycznie przez aplikację, o nazwie *bufor.txt*. Postać, pod jaką zapisywane są informacje jest następująca:

```
Numer Skrzydla: 0000259003001700    Numer Miejsca: 0005    śr. wrz 13 11:12:54 2017
```

Rysunek 18. Jedna z linii pliku *bufor.txt*

Równocześnie z zapisywaniem danych uruchamiany jest drugi wątek aplikacji, który sprawdza, czy możliwe jest nawiązanie połączenia z bazą danych. Jeśli odpowiadająca za to funkcja zwróci błąd, to dane pozostaną zapisane w pliku i nie zostaną wysłane. Wątek ten uruchamiany jest przy każdym uruchomieniu akcji „Zatwierdź”. Tak długo, jak połączenie nie zostanie nawiązane, dane będą dopisywane do pliku tekstowego w nowej linii, zgodnie ze schematem pokazanym wyżej.

```
4. Numer Skrzydla: 0004850003014500    Numer Miejsca: 0007    śr. wrz 13 11:11:30 2017
5. Numer Skrzydla: 0000279016028300    Numer Miejsca: 0009    śr. wrz 13 11:12:10 2017
6. Numer Skrzydla: 0000259003001700    Numer Miejsca: 23      śr. wrz 13 11:12:30 2017
7. Numer Skrzydla: 0000136900111000    Numer Miejsca: 0005    śr. wrz 13 11:12:54 2017
```

Rysunek 19. Buforowanie danych w pliku *bufor.txt*

Przy udanej próbie połączenia z bazą danych następuje opróżnianie bufora. Program otwiera plik *bufor.txt*, wczytuje z niego pierwszą linię i odseparowuje zawarte w niej dane do odpowiednich zmiennych w pamięci; pozostałe dane zapisuje w tablicy i przepisuje je do innego pliku tekstowego *tmp.txt*. Te zmienne, opatrzone odpowiednimi etykietami, wędrują do bazy danych. Następnie zmieniony już plik *bufor.txt* – pozbawiony pierwszej linii jest usuwany, a nowoutworzonemu plikowi tymczasowemu (*tmp.txt*), wypełnionemu resztą danych, przydzielana jest nazwa starego, dla porządku. Cała procedura jest powtarzana tak długo, jak w pliku buforowym istnieją jeszcze jakieś linie. Jeśli w czasie trwania tej pętli programu,

użytkownik postanowi wprowadzić kolejne dane do programu, nie spowoduje to zakłócenia działania aplikacji. Dane zostaną dopisane w nowej linii, na końcu pliku buforowego.

### 5.1.6. Udogodnienia dla pracowników

Dla wygody pracowników, biorąc pod uwagę różne charaktery, cały wachlarz preferencji, przygotowano szereg udogodnień. Aby program był postrzegany za przyjazny w obsłudze zdecydowano się na implementację różnych konwencji korekty wprowadzonych danych. Pracownicy mają w tym celu do dyspozycji:

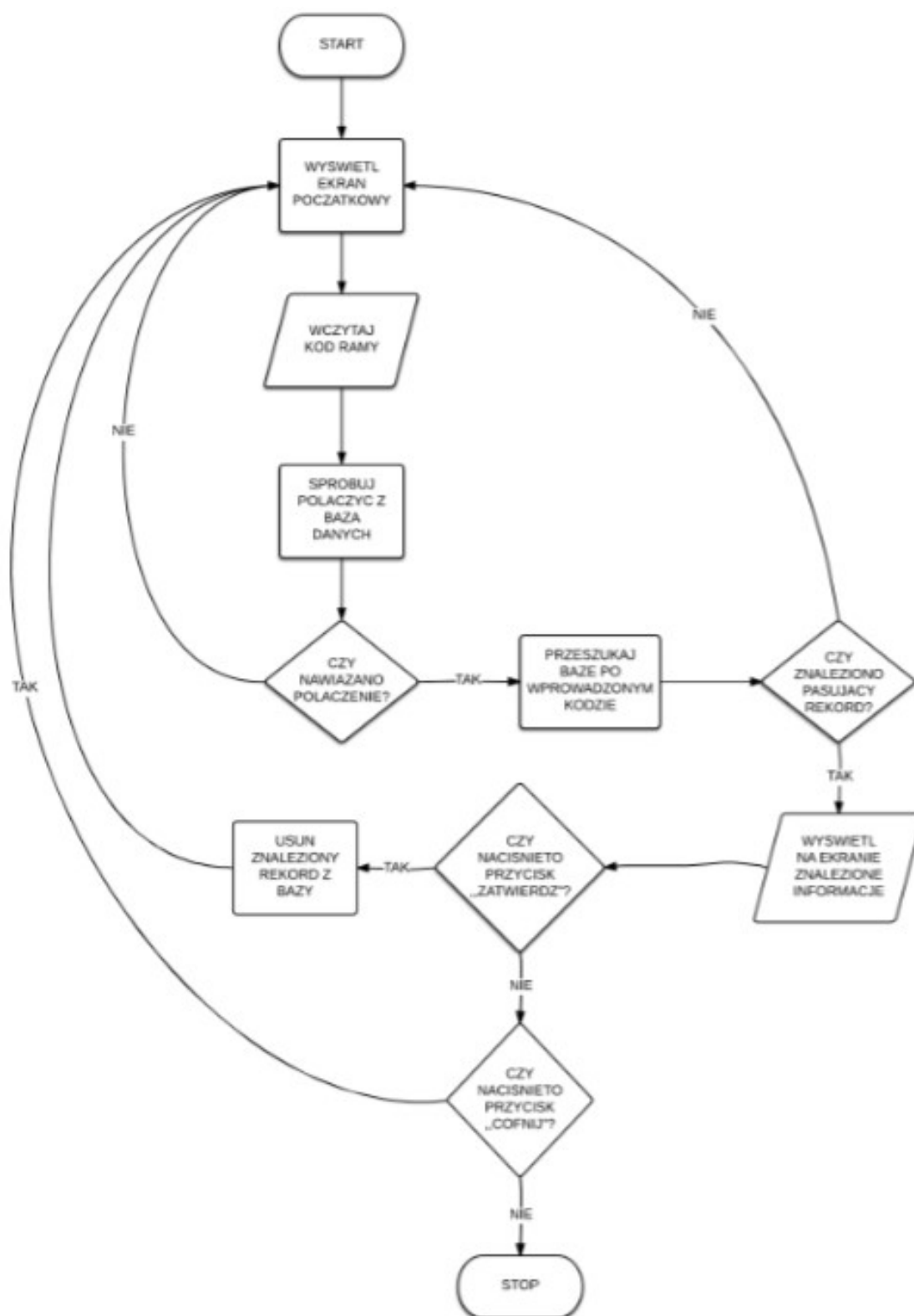
- skorzystanie z przycisków interfejsu graficznego (za pomocą ekranu dotykowego lub myszy),
- skorzystanie z kodów kreskowych akcji – wystarczy, że zeskanuje przygotowany na plakietce umieszczonej przy jego miejscu pracy kod, co uruchomi odpowiednią funkcję programu,
- ponowne zeskanowanie danych (kodu skrzydła i kodu miejsca), w celu aktualizacji danych.

W kodzie źródłowym przewidziana została również auto – maksymalizacja okna programu zaraz po jego uruchomieniu. Dodatkowo zastosowano w aplikacji layout typu siatka, dzięki czemu cała struktura interfejsu pozostaje nienaruszona, w stałych proporcjach, niezależnie od ekranu, na którym program jest wyświetlany.

Aby zapewnić pracownikom komfort pracy, zdecydowano się nie narzucać im kolejności wprowadzania danych. Ponieważ program rozpoznaje długość łańcucha znaków, to nie ma znaczenia, która informacja zostanie mu przekazana najpierw. Pracownicy zatem mogą zeskanować kod skrzydła jeszcze w trakcie okuwania, a następnie podać numer slotu lub wózka, do którego skrzydło odłożył, lub wprowadzić najpierw miejsce, a potem zeskanować jeden z kodów skrzydła.

## 5.2. Aplikacja na stanowisku montażu

Na Rys. 20 zaprezentowany jest schemat blokowy działania aplikacji.



Rysunek 20. Schemat blokowy działania aplikacji na stanowisku montażu

### 5.2.1. Opis działania aplikacji

Ten program został wdrożony na stacjach roboczych obejmujących linię montażu ram i skrzydeł w zestawy. Głównym zadaniem aplikacji jest wysyłanie zapytania do bazy danych w celu przeszukania jej pod kątem wystąpienia skrzydeł pasujących do ramy i miejsc, w których są przechowywane. Rozpoznawanie, które skrzydła są sparowane z ramą odbywa się za pomocą „barcode’u” – proces został dokładnie opisany w rozdziale dedykowanym bazie danych. Pracownicy skanując kod kreskowy ramy, lub wprowadzając go za pomocą klawiatury, podają argument, po którym baza ma zostać przeszukana. W odpowiedzi otrzymują wykaz miejsc, w których składowane są skrzydła dedykowane do ramy, która aktualnie znajduje się na taśmie. Wykaz ten jest prezentowany w odpowiednim miejscu interfejsu graficznego. Aplikacja umożliwia zrzucenie z bazy danych rekordu, który zawiera informację o skrzydłach, które zostały już zamontowane do ramy.

### 5.2.2. Graficzny interfejs użytkownika

Na Rys.21 widnieje okno aplikacji, zaraz po jej wystartowaniu.



MainWindow

Zeskanuj kod ramy:

NUMER ZAMOWIENIA:      WYSTĄPIENIE:      NUMER PARTII:

SKRZYDŁO ZNAJDUJE SIĘ W

**ZATWIERDŹ**      **COFNIJ**

Rysunek 21. Podstawowy widok aplikacji na stanowisku montażu

Po wprowadzeniu i odnalezieniu danych w bazie aplikacja przybiera postać widoczną na Rys.22.

The screenshot shows a software window titled 'MainWindow'. It contains the following elements:

- A label 'Zeskanuj kod ramy:' followed by a text input field containing the value '0000259003001700'.
- Three data fields:
  - 'NUMER ZAMOWIENIA:' with the value '27-09/1411-17'
  - 'WYSTĄPIENIE:' with the value '4'
  - 'NUMER PARTII:' with the value '41/09/51-231'
- A green status message: 'SKRZYDŁO ZNAJDUJE SIĘ W'.
- A label 'STOJAKU NUMER:' followed by two large numbers, '10' in red and '15' in pink.
- A label 'WÓZKU NUMER:' followed by two large numbers, '58' in blue and '375' in green.
- At the bottom, there are two large buttons: a green one labeled 'ZATWIERDŹ' and an orange one labeled 'COFNIJ'.

Rysunek 22. Widok aplikacji na stanowisku montażu po wprowadzeniu danych

### 5.2.3. Korekcja wprowadzanych danych

Podobnie jak w przypadku aplikacji stworzonej dla stanowiska okuwania, tak i w tym przypadku, użytkownik programu może dokonywać korekt – aktualizacji na dwa sposoby. Przewidziano tutaj akcję czyszczącą formularz, do którego wprowadza się dane. Uruchomienie jej odbywa się po:

- naciśnięciu przycisku cofnij na ekranie dotykowym,
- naciśnięciu przycisku za pomocą myszy,
- zeskanowaniu kodu kreskowego, oznaczonego etykietą „Cofnij”, znajdującego się na stanowisku pracowniczym,
- zeskanowaniu nowego kodu kreskowego – automatyczna aktualizacja danych.

### 5.2.4. Uwierzytelnianie dostępu do bazy danych

Proces uwierzytelniania połączenia z bazą danych wygląda niemalże jak w przypadku aplikacji do obsługi stanowiska okuwania. Z tą różnicą, że próba połączenia jest nawiązywana na samym początku – w momencie uruchamiania programu. Jeśli połączenie nie zostało nawiązane, użytkownik nie ma możliwości wyszukania informacji o miejscu przechowywania

skrzydła. Jeśli połączenie zostanie nawiązane poprawnie, istnieje możliwość przeszukiwania bazy danych pod kątem miejsc przechowywania skrzydeł do danej ramy. Po zeskanowaniu kodu kreskowego ramy, aplikacja wysyła żądanie do bazy, otrzymuje informację zwrotną w postaci: numeru zamówienia, ilości wystąpień numeru partii i miejsc składowania wszystkich skrzydeł dla danej ramy.

Pracownicy po zamontowaniu zestawu półproduktów mogą zrzucić wyświetlany na ekranie rekord z bazy danych, jeśli naciśną przycisk „Akceptuj” lub zeskanują odpowiedni kod kreskowy z plakietki przymocowanej przy swoich stanowiskach. Po zrzuceniu rekordu z bazy danych interfejs wraca do początkowego widoku.

Jeśli jednak rekord nie zostanie zrzuty z bazy danych przez użytkowników programu, zostanie w niej przechowany nie dłużej niż miesiąc – ze względu na działanie opisanych we wcześniejszym rozdziale wyzwalaczy zaimplementowanych w bazie.

#### **5.2.5. Udogodnienia dla pracowników**

Oprócz zaimplementowania tych samych udogodnień dotyczących korekt, co w przypadku aplikacji dla stanowiska okuwania – korekta automatyczna lub manualna, za pomocą przycisków – przewidziano dodatkowe udogodnienie dla pracowników montażu. W interfejs graficzny wbudowano specjalną tabelkę. W jej komórkach wyświetlane są wszystkie miejsca przechowywania skrzydeł dla danej ramy, z separacją na skrzydła w slotach stojaków i na wózkach. Każde kolejne skrzydło oznaczone jest innym kolorem w celu łatwiejszej orientacji i eliminacji skrzydeł wyszukanych i zamontowanych.

## Rozdział 6. Możliwości rozszerzania systemu

Dynamiczny rozwój przedsiębiorstwa, stale wzrastająca liczba zamówień i pojawianie się w ofercie nowych produktów wymusiły uwzględnienie w projektowanym systemie możliwości jego dalszego powielania, rozszerzania na kolejne stacje robocze świeżo tworzonych nowych sektorów hali produkcyjnej, czy w ogóle przeniesienie systemu również do innego budynku, jeśli w przyszłości zajdzie taka potrzeba.

Oczywiście nie sposób przygotować jakiegokolwiek systemu tak, aby był on sprawny w każdych warunkach, niezależnie od sprzętu, na jakim ma działać, środowiska uruchomieniowego i przede wszystkim takiego systemu, który sprosta wszystkim przyszłym zadaniom, jakie się przed nim postawi.

W fazie projektowania systemu uwzględniono dość dużą elastyczność możliwości jego powielania. Wprawdzie każda zmiana sposobu oznaczania gotowych wyrobów lub miejsc przechowywania wymagać będzie zmiany w kodzie źródłowym i ponownej kompilacji programów, jednak takie postępowanie wydaje się bezpodstawne. System identyfikacji wystąpień w zamówieniach w firmie z pewnością nie zostanie zmieniony, byłoby to niezgodne z polityką firmy, a zmiany w systemie identyfikacji miejsc przechowywania są bezzasadne – obecne rozwiązania, wdrożone w wielu punktach hali produkcyjnej, sprawdzają się znakomicie i żaden z pracowników zakładu nie zgłaszał do nich żadnych zastrzeżeń. Jeśli zatem wymienione wyżej standardy nie zostaną zmienione, rozszerzenie używanego systemu o kolejne stacje robocze, czy wdrożenie go w innych sektorach hali produkcyjnej ograniczy się tylko do wprowadzenia do nowego komputera specjalnie przygotowanego wcześniej archiwum. Archiwum takie zawiera:

- aplikacje dedykowane do systemu skompilowane i wydane w postaci pliku wykonywalnego,
- biblioteki .dll środowiska Qt niezbędne do prawidłowego działania programu,
- plik tekstowy z przygotowanymi przykładowymi danymi do logowania do bazy danych, które wystarczy zmienić na takie, które uwzględnił administrator serwerowni („uzytkownik.txt),
- instrukcje dotyczące ścieżek docelowych wszystkich plików archiwum, uwzględniające charakterystyczne firmowe zabezpieczenia.

Forma pliku wykonywalnego umożliwia uruchomienie programu również poza środowiskiem programistycznym – bez konieczności kompilacji, budowania i wystartowania programu ręcznie. Jest to bardzo wygodne rozwiązanie, ponieważ uwalnia ono od żmudnego instalowania środowiska programistycznego na każdej stacji roboczej, na której program ma być uruchomiony. Jednak do poprawnego działania takiej formy programu na danym systemie operacyjnym niezbędne jest dołączenie pakietu bibliotek. W archiwum znajdują się biblioteki odpowiadające między innymi za:

- uruchamianie programu na systemach Windows,
- poprawne działanie konektora i wykorzystanie sterowników do bazy danych,
- obsługę wielowątkowości programu,
- prawidłowe wyświetlanie interfejsu graficznego aplikacji.

Pełny wykaz bibliotek koniecznych do uruchomienia programów na systemie Windows 7 Embedded i Windows XP – tymi dysponowała firma na hali produkcyjnej:

- libgcc\_s\_dw2-1.dll,
- libstdc++ - 6.dll,
- libwinpthread-1.dll,
- Qt5Core.dll,
- Qt5Gui.dll,
- Qt5Sql.dll,
- Qt5Widgets.dll,
- Zawartość folderu platforms, znajdującego się w lokalizacji Qt,
- Zawartość folderu sqldrivers, znajdującego się w lokalizacji Qt,

Dzięki przygotowanemu szablónowi pliku z danymi do logowania do bazy danych zniwelowano problem uwierzytelniania połączenia. Wystarczy, że osoba administrująca bazą danych doda kolejnego użytkownika, a przy instalowaniu zestawu na nowej stacji roboczej zostaną zmienione dane w pliku na login i hasło nowoutworzonego użytkownika.

Wszystkie powyższe zabiegi, łącznie z właściwościami charakterystycznymi stworzonego systemu sprawiają, że rozbudowa architektury jest bardzo prosta i wygodna do przeprowadzenia, a w dodatku nie wymaga instalowania na każdej dodanej stacji roboczej środowiska programistycznego do poprawnego działania programu. Dzięki takim działaniom skalowalność systemu jest niemalże nieograniczona.



## Rozdział 7. Podsumowanie

Opracowany system spełniał wszystkie założenia – omówione na początku pracy – postawione przez firmę, dla której został zaprojektowany. Dzięki niemu udało się znacząco zwiększyć przepustowość sektora hali, co przełożyło się w ogólnym rozrachunku na większą ilość wyrobów przesyłanych na dalsze ogniwa łańcucha produkcyjnego.

Wykorzystanie w pełni narzędzi na stanowiskach pracowniczych i odpowiedniej infrastruktury hali produkcyjnej pozwoliło ograniczyć wydatki firmy na ten projekt do absolutnego minimum – opłaty wiązały się kosztami eksploatacji drukarki etykiet, będącej na wyposażeniu biurowym przedsiębiorstwa – co nie nadszarpnęło w żaden sposób budżetu firmowego, a pozwoliło rozwiązać naprawdę istotny dla funkcjonowania zakładu problem.

Metoda identyfikacji slotów stojaków okazała się nie tylko efektywna, ale również ergonomiczna i razem z aplikacją na stanowisku okuwania znacząco wpłynęły na uporządkowanie składowania gotowych półproduktów, rozsądniejsze gospodarowanie miejscem przechowywania – w pewnych momentach produkcji wózki – jako strefy buforowe okazywały się zbędne.

Wyposażenie aplikacji w system buforów stanowiących barierę dla danych przepływających między pracownikiem, aplikacją, a bazą danych, przetrzymujących te dane na wypadek utraty zasilania, nieprawidłowego uwierzytelnienia, czy jakichkolwiek operacji konserwacyjnych wykonywanych na bazie przez administratorów okazał się znakomitym pomysłem i całkowicie spełnił swoje zadanie.

Dzięki zaimplementowanym w aplikacji kilku wątkom żaden z pracowników zatrudnionych na stanowisku okuwania nie musiał przejmować się ewentualnym zakorkowaniem bazy danych. Niezależnie od tego, czy połączenie z bazą istniało, czy nie, wszystkie dane były przechowywane lokalnie, co stanowiło – w wypadku niepomyślnej pracy bazy danych – potwierdzenie działań pracownika. Zapisywanie danych, odbywające się w jednym wątku, było błyskawiczne, a wysyłanie ich do bazy, zaimplementowane w innym wątku, działało się po wywołaniu odpowiedniej akcji („Zatwierdź”), co zapewniało płynność i bezawaryjny charakter całego systemu i nie narażało pracowników na przestoje podczas zmian.

Interfejs użytkowników aplikacji został przyjęty przez pracowników z entuzjazmem. Głównie ze względu na jego prostotę i uniwersalność – dzięki czemu pracownicy nie musieli

przechodzić czasochłonnego szkolenia, ani nie trzeba było tworzyć obszernej instrukcji obsługi programu.

Przygotowanie kodów kreskowych „akcji” programu („Zatwierdź”, „Cofnij”, „Reset”), umożliwiło pracownikom obsługę programu zgodnie z preferencjami. Ci, którzy woleli posługiwać się skanerem, mogli po prostu wczytywać do aplikacji odpowiednie kody. Jeśli jednak ktoś cenił sobie bardziej korzystanie z dotykowego ekranu – jeśli w takie zaopatrzone było jego stanowisko, mógł pracować w ten sposób. Pozostawienie tej dowolności spodobało się pracownikom i sprawiło, że każdy z nich rozpoczął pracę w nowym systemie z zapałem i bez obiekcji.

Po przeprowadzeniu obserwacji użytkowania systemu w normalnym cyklu produkcyjnym przedsiębiorstwa stwierdzono, że dzięki wszystkim elementom tego systemu udało się w pełni wykorzystać potencjał pracowników, którzy od momentu wdrożenia systemu nie marnowali już swojego czasu na wielokrotne marsze wzdłuż wszystkich stojaków i wózków w poszukiwaniu odpowiedniego dla ramy skrzydła. Mogli wykazać się od tego momentu swoimi umiejętnościami obróbki dostarczonych na ich stanowiska wyrobów, nabytymi podczas zakładowych szkoleń oraz szybciej przekazywać owoce swojej pracy – oraz większą ich ilość – dalej, do kolejnego ogniw łańcucha produkcyjnego.

Zastosowanie systemu wspomagającego kompletację elementów zamówień w przedsiębiorstwie Budvar Centrum Sp. z o.o. zaowocowało nie tylko na zwiększeniu potencjału produkcyjnego sektora hali o najmniejszej przepustowości – czyli zintensyfikowaniem produkcji, a co za tym idzie przyniesieniem znacznie wyższych dochodów firmie – ale przełożyło się w ostateczności na ogólne zadowolenie pracowników, na zwiększenie poczucia dobrze wykonywanej przez nich pracy, a nie marnotrawieniu czasu na bezczynności i polepszenia atmosfery panującej na każdej zmianie.

## Bibliografia

1. Summerfield M., *Biblioteki Qt. Zaawansowane programowanie przy użyciu C++*, Helion, 2014.
2. [blog.matthew.org.pl/kategoria/programowanie/qt/kurs-qt/](http://blog.matthew.org.pl/kategoria/programowanie/qt/kurs-qt/)
3. Blanchette J., Summerfield M., *C++ GUI Programming with Qt 4. Second Edition*, Trolltech ASA, 2008.
4. Ezust A., Ezust P., *C++ i Qt. Wprowadzenie do wzorców projektowych. Wydanie II*, Helion, 2014.
5. <http://doc.qt.io/>
6. Thelin J., *Foundations of Qt Development*, Apress, 2007.
7. Nevarez B., *Microsoft SQL Server 2014 Optymalizacja zapytań*, Helion, 2015.
8. Pelikant A., *MS SQL Server. Zaawansowane metody programowania*, Helion, 2014.
9. Liwowski B., Kozłowski R., *Podstawowe zagadnienia zarządzania produkcją*, Oficyna Ekonomiczna, Kraków 2006.
10. Strychalski R., *Programowanie w C++ z użyciem biblioteki Qt4*, Nakom, 2014.

# SPIS RYSUNKÓW

<b>Rysunek 1.</b> Schemat sektora hali produkcyjnej .....	7
<b>Rysunek 2.</b> Szkic wózka z etykietą identyfikacyjną .....	8
<b>Rysunek 3.</b> Szkic stojaka na okute skrzydła.....	9
<b>Rysunek 4.</b> Przykładowe stanowisko okuwania skrzydeł .....	9
<b>Rysunek 5.</b> Przykładowe stanowisko do okuwania skrzydeł i skaner kodów kreskowych ....	10
<b>Rysunek 6.</b> Kod kreskowy wózka oznaczonego jako: @1000965.....	11
<b>Rysunek 7.</b> Umieszczenie etykiet slotów na stojaku.....	12
<b>Rysunek 8.</b> Przykładowe oznaczenie slotu stojaka .....	13
<b>Rysunek 9.</b> Przykładowy kod kreskowy skrzydła.....	15
<b>Rysunek 10.</b> Możliwe konfiguracje kodów kreskowych skrzydła.....	15
<b>Rysunek 11.</b> Schemat blokowy działania aplikacji dla stanowiska okuwania.....	19
<b>Rysunek 12.</b> Kod kreskowy przycisku „Akceptuj” .....	21
<b>Rysunek 13.</b> Kod kreskowy przycisku „Cofnij” .....	21
<b>Rysunek 14.</b> Kod kreskowy przycisku „Reset”.....	21
<b>Rysunek 15.</b> Podstawowy widok aplikacji na stanowisku okuwania .....	22
<b>Rysunek 16.</b> Okno aplikacji po wprowadzeniu do niej wszystkich danych .....	23
<b>Rysunek 17.</b> Widok aplikacji po użyciu „Cofnij” i wprowadzonym ostatnio miejscu.....	24
<b>Rysunek 18.</b> Jedna z linii pliku bufor.txt.....	25
<b>Rysunek 19.</b> Buforowanie danych w pliku bufor.txt.....	25
<b>Rysunek 20.</b> Schemat blokowy działania aplikacji na stanowisku montażu .....	27
<b>Rysunek 21.</b> Podstawowy widok aplikacji na stanowisku montażu .....	28
<b>Rysunek 22.</b> Widok aplikacji na stanowisku montażu po wprowadzeniu danych.....	29