Kraków, 23.01.2017
AGH, WIMiIP, IS02
BARTOSZ GOSŁAWSKI

INŻYNIERIA OPROGRAMOWANIA

Temat 3: **PROJEKT PROGRAMU UMOŻLIWIAJĄCEGO AUTOMATYCZNE WYGENEROWANIE OPTYMALNEGO LUB SUB-OPTYMALNEGO PLANU ZAJĘĆ**

Funkcja w zespole: **Analityk**
Niniejszy dokument jest jedynie częścią całego projektu i opisuje zadania skierowane do mojej osoby, która pełni pierwszą z pięciu funkcji w zespole, którego pozostałymi członkami są:
Patryk Jakubiec, Paweł Głowacz, Grzesiek Haczyk, Magdalena Chmura, Piotr Hanusiak

Sprawozdanie podzielone jest na następujące segmenty:

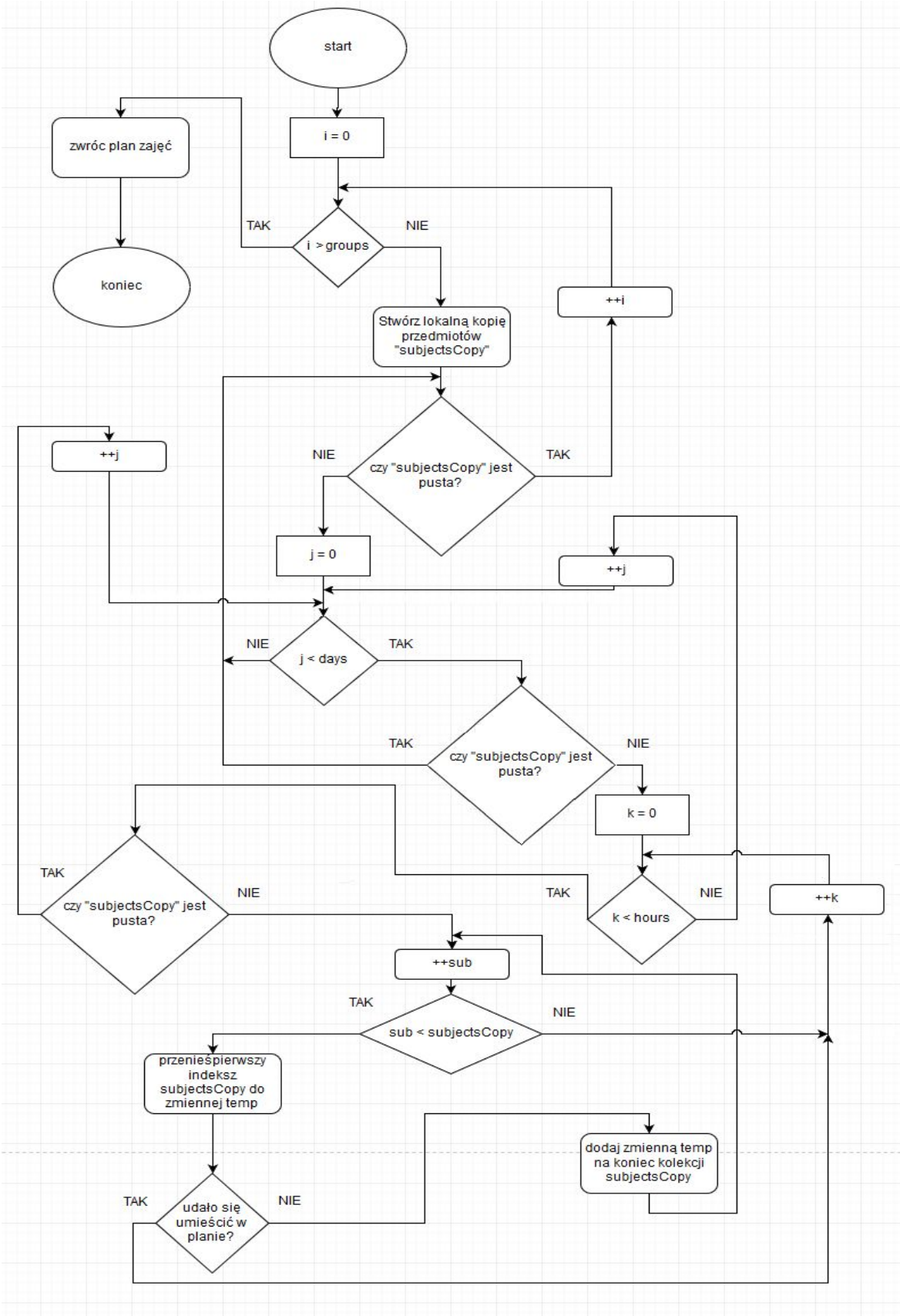
1. a.Algorytmy, b.Analizy.
2. a.Lista wymagań, b.przypadki użycia, c.historie użytkownika.
3. Diagram wymagań.
4. a.Diagram stanów, b.diagram aktywności
5. Zakończenie (użyte narzędzia, bibliografia)

Wprowadzenie:

Zadanie które zostało powierzone mojej grupie: *3. Program umożliwia automatyczne wygenerowanie optymalnego lub sub-optymalnego planu zajęć.* polegało na szeregu problemów, które musieliśmy rozwiązać w zespole. Z racji sprawowania funkcji analityka, byłem odpowiedzialny za: wykonanie algorytmu ustalającego plan zajęć dla uczniów i nauczycieli, analiza możliwych do użycia gotowych rozwiązań oraz wykonanie szeregu diagramów, pozwalających jasno sprecyzować funkcjonalność, budowę i możliwości nieniejszego programu.

1a. Algorytm:

Schemat blokowy:



W programie wykorzystano algorytm przypisujący zajęcia wg powyższego schematu blokowego. Przed przystąpieniem do algorytmu, program wysuwa zapytanie, czy ma się sugerować preferencjami. Lista przedmiotów zostaje umieszczona na temp. liście. Następnie, algorytm hierarchicznie przechodzi przez każdą grupę, przez każdy dzień tygodnia, przez każdą godzinę lekcyjną i próbuje przypisać odpowiedni przedmiot. W tym kroku (odpowiedzialnym za przypisanie przedmiotu) sprawdzanych jest kilka warunków odpowiedzialnych za możliwość przypisania. Jeżeli wszystkie są spełnione przedmiot zostaje przypisany. Jeżeli nie, przedmiot zostaje ustawiony na końcu listy. W efekcie program próbuje przyporządkować plan dla każdej grupy odpowiednio na początek tygodnia, w kolejnych godzinach.

1b. Analiza

Zastosowany algorytm jest sporządzony oraz opracowany przez naszą grupę. Został napisany i przetestowany przez nasz zespół i nie zostały w nim użyte gotowe wzorce lub algorytmy. Powody całkowicie własnej implementacji są następujące:

-brak konieczności zaznajamiania się z gotowymi algorytmami i czytania oraz opracowywania nie swojego kodem (wyklucza to możliwość błędów popełnionych przez kogoś w gotowym rozwiązaniu)

-satysfakcja z autorskiego rozwiązania problemu

-brak konieczności zaznajamiania się np. z teorią algorytmów genetycznych

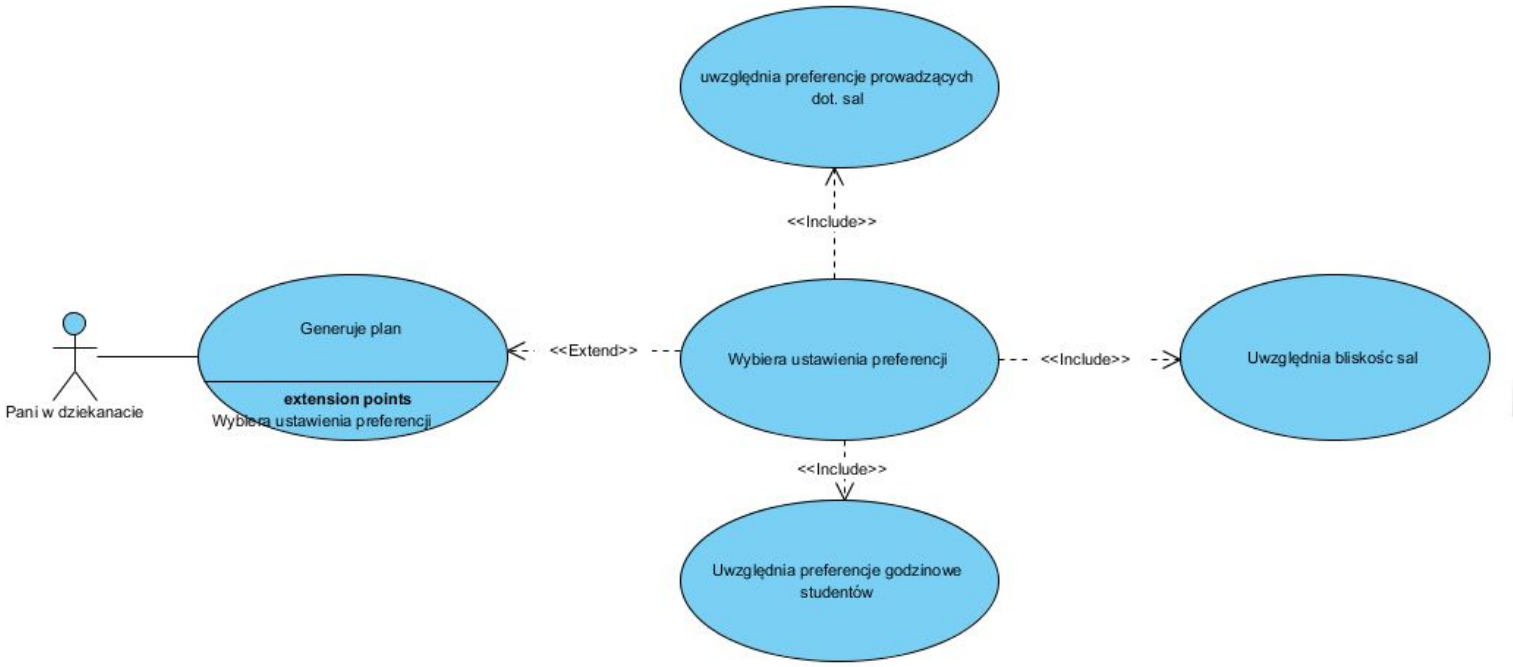
-ambitne podejście do prezentowanego problemu

Jednakże warto wiedzieć jakie narzędzia były do dyspozycji z gotowych rozwiązań. Otóż można tutaj zaprezentować metodę algorytmów genetycznych. W problemie związanym z układaniem planu zajęć bardzo dobrze wszelkie zagadnienia opisuje Marek Jaszuk z PWSZ w Chełmie:

"*Przystępując do układania planu musimy przyporządkować określonych nauczycieli określonym przedmiotom oraz określonym grupom studentów. Jeżeli połączymy ze sobą te trzy elementy, to uzyskamy pewien element podstawowy, który nazwiemy zajęciami. Jednostka zajęciowa będzie stanowiła podstawowy element, z którego zbudowane są chromosomy, a więc gen. (...). W tym momencie możemy przystąpić do rozwiązywania głównego problemu, czyli rozmieszczenia jednostek zajęciowych w interwałach czasowych oraz w salach. (...) W pierwszym etapie poszczególnym zajęciom przyporządkowywane są interwały czasowe. Po zakończeniu alokacji czasowej przystępujemy do rozmieszczania zajęć w salach zgodnie z ustalonymi wcześniej godzinami. (...) Mamy w tym przypadku do czynienia z genami wielo wartościowymi. W fazie układania godzin zajęciowych gen zawiera numer godziny. Zakładając, że mamy w ciągu tygodnia Ng godzin zajęciowych przy liczbie zajęć (długości chromosomu) równej N otrzymujemy N^N g możliwych kombinacji. Z kolei w fazie rozmieszczania zajęć w salach każdy gen zawiera numer sali przyporządkowanej danym zajęciom. Przy założeniu, że mamy Ns dostępnych sal, dostajemy liczbę możliwych kombinacji równą N Ns . Oczywiście większość z tych kombinacji daje w wyniku plany, których nie możemy zaakceptować z uwagi na omówione wcześniej ograniczenia. Nie należy*

się jednak tym przejmować, gdyż zadanie utworzenia planu nie łamiącego ograniczeń spoczywa na algorytmie genetycznym.(...)"[3]

2a. Przypadki użycia:



2b: Historie użytkownika:

A. Pani w dziekanacie, generuje plan, żeby udostępnić studentom i prowadzącym ich plany zajęć

B. Pani w dziekanacie, uwzględnia preferencje prowadzących, by spełnić ich oczekiwania

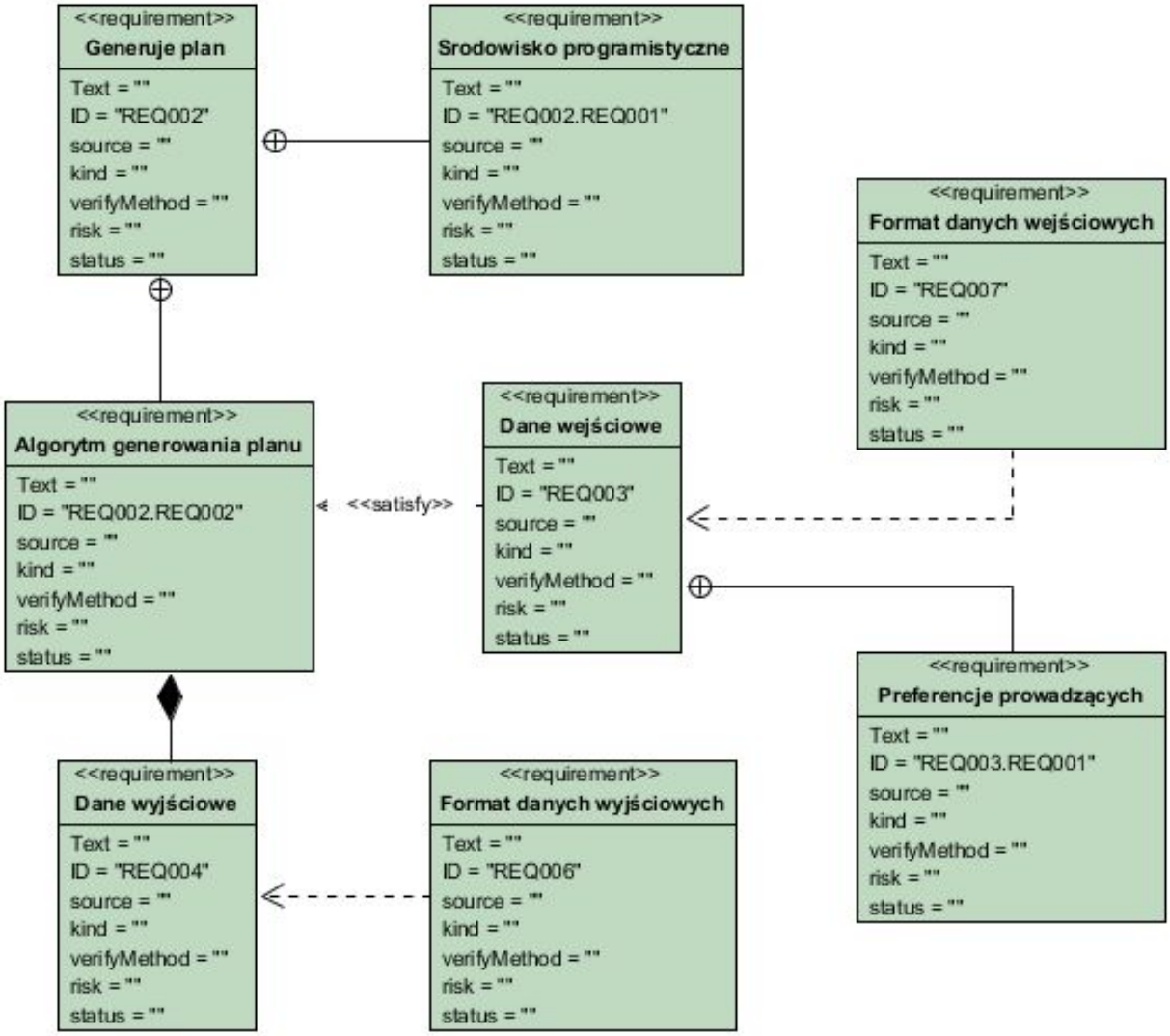
D. Pani w dziekanacie, uwzględnia preferencje studentów, by spełnić ich oczekiwania*

2c: Lista wymagań:

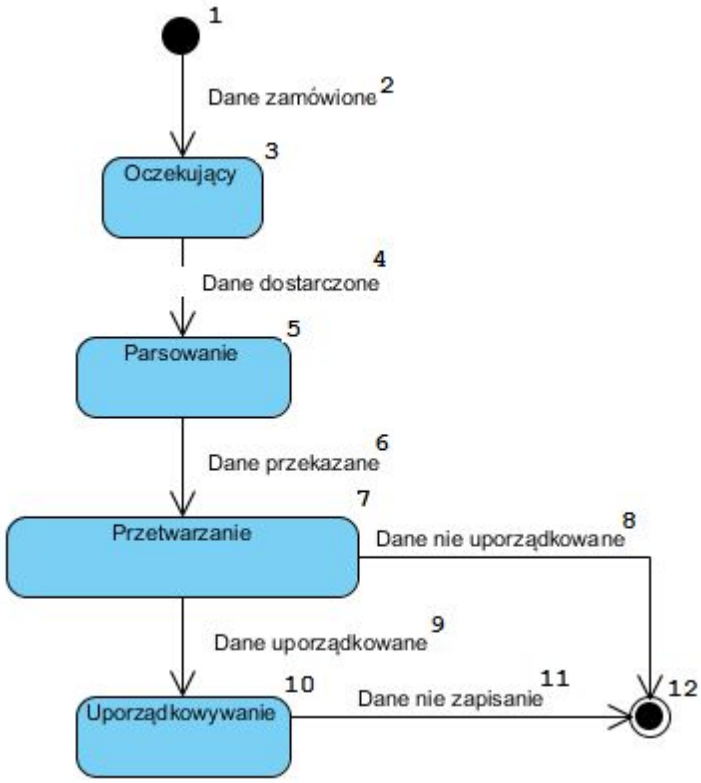
-rola pani w dziekanacie
-generowanie spójnego planu dla studentów, prowadzących
-rozkład sal (piętro, sala)
-zapis planu do odpowiedniego formatu
-sprecyzowane preferencje prowadzących
-sprecyzowane preferencje studentów*

NIestety nie udało się stworzyć wersji algorytmu uwzględniającego preferencje studentów.

3. Diagram wymagań



4a. Diagram stanów:



przyjęto, że uporządkowane dane to plan zajęć

-Klasa dane - posiada stan początkowy (1) i stan końcowy (12).

-W trakcie swojego istnienia dane mogą przyjąć jeden z czterech stanów:
Oczekujący(3), Parsowanie(5), Przetwarzanie(7), Uporządkowywanie(10)

-Zdarzenie <Dane zamówione> prowadzi do sytuacji, gdy znikąd (1) w systemie pojawiają się
dane(narodziny). Natychmiast po narodzinach dane przechodzą w stan Oczekujący(3).

-Jeśli wystąpi zdarzenie <Dane dostarczone>(4), a dane pozostają w stanie Oczekującym(3),
zdarzenie to powoduje zmianę tego stanu na w Parsowanie(5).

-W wyniku zdarzeń <Dane przekazane>(6) dane zmienią swój stan z Parsowanie(5) na Przetwarzanie(7)

-Następnie Dane w wyniku zdarzenia <Dane uporządkowane>(9) zmieniają stan z Przetwarzanie(7) na Uporządkowywanie(10)

-Zdarzenie <Dane nie uporządkowane>(8) jak i <Dane nie zapisane>(11) prowadzą do końca
życia obiektu kolejno ze stanów Przetwarzanie(7) i Uporządkowywanie(10). Koniec życia oznacza, iż znikają w nicości (12) (śmierć).

4b. Diagram aktywności:

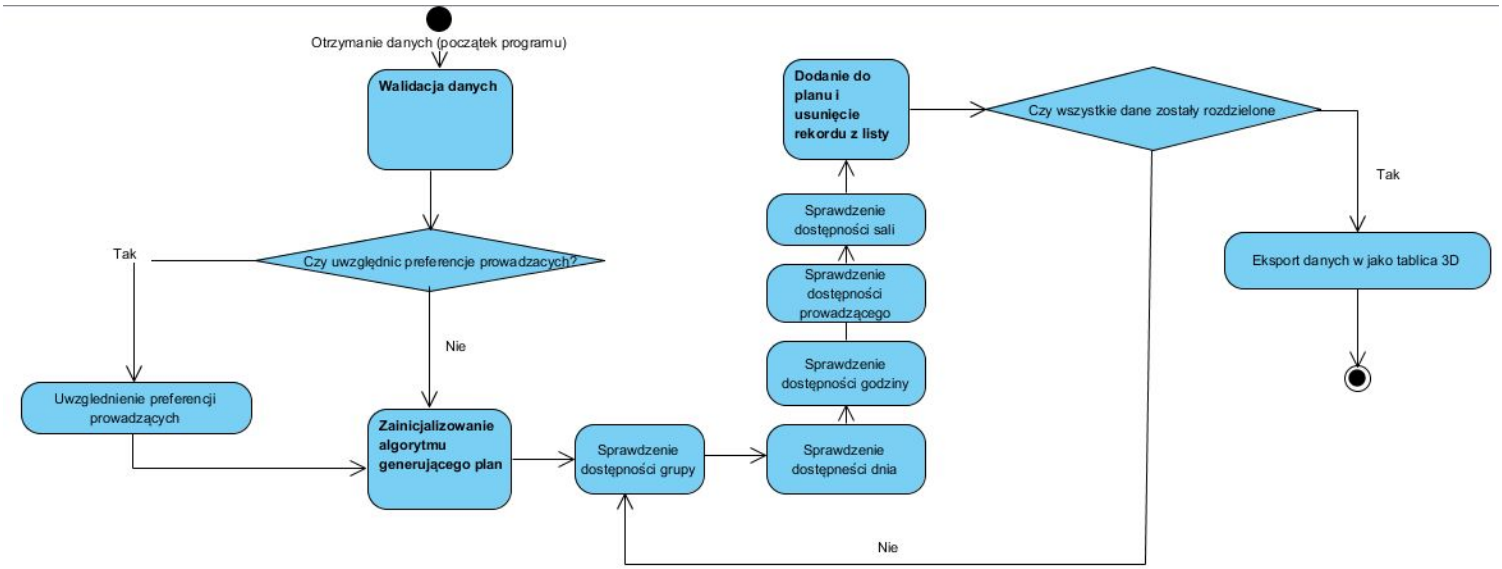


Diagram aktywności w UML służy do projektowania czynności i odpowiedzialności elementów oraz użytkowników systemu. Opisuje działania związane z wieloma obiektem, pomiędzy którymi może występować komunikacja

5. Zakończenie

Wykorzystany język półformalny: Język UML

A. Za twórców UML uważa się:
-G. Booch
-I. Jacobson
-J. Rumbaugh [1]

B. Unified Modeling Language (UML)-służy do opisywania systemów z użyciem słów i diagramów.
Może być używany do modelowania różnorodnych systemów: informatycznych, biznesowych lub dowolnych innych.[2]

C. UML w skrócie: czym jest?
-Językiem pozwalającym tworzyć modele systemów (np. informatycznych)
-Pozwala obrazować, specyfikować, tworzyć i dokumentować elementów systemu
-Ułatwia wymianę informacji pomiędzy przyszłymi użytkownikami systemu, menadżerami, analitykami, projektantami, programistami i testerami

-Ułatwia wykorzystanie zalet programowania obiektowego
-UML jest jedynie językiem modelowania używanym w procesie analizy i projektowania systemów komputerowych

D. Czym nie jest?
-Językiem programowania - choć generowanie kodu jest możliwe na podstawie niektórych diagramów, rzadko jest to stosowane
-Narzędziem - choć zawiera specyfikacje dla narzędzi
-Metodyką - nie definiuje procesu tworzenia oprogramowania
-Unified Modeling LanguageUML nie jest sposobem na analizę i projektowanie systemów komputerowych[1]

Istnieje kilka argumentów przemawiających za wykorzystaniem UML-a w charakterze języka modelowania. Oto one:

§ UML pozwala na unifikację terminologii i standaryzację notacji, co prowadzi do znacznego uproszczenia komunikacji pomiędzy wszystkimi zaangażowanymi w projekt grupami ludzi. Dzięki niemu możliwa jest bezproblemowa wymiana modeli systemu między departamentami lub firmami. Co więcej, projekty mogą być z łatwością przekazywane między zespołami projektowymi lub ich członkami.
§ UML rozrasta się wraz ze wzrostem oczekiwań w stosunku do technik modelowania systemów. Mimo że jest on językiem modelowania o dużych możliwościach, znajomość z nim można spokojnie zacząć od prostych modeli; można w nim też tworzyć rozbudowane i szczegółowe projekty systemów. Jeśli standardowe możliwości UML-a nie wystarczają, można go rozbudować, używając do tego celu stereotypów.
§ UML wykorzystuje szereg znanych i akceptowanych w branży metodologii. Oznacza to, że nie został on zaprojektowany od zera jako projekt akademicki — wręcz przeciwnie, miał stanowić odpowiedź na z gruntu praktyczne problemy i został oparty o istniejące języki modelowania. Taka geneza gwarantuje jego użyteczność i nadaje mu bardzo praktyczną wartość.
§ UML jest językiem obsługiwanym przez wiele narzędzi.[2]

Bibliografia:

[1] - źródło: *http://zasoby.open.agh.edu.pl*
[2] - źródło: "*Uml 2.0 in Action: A Project-based Tutorial*", 2005 by Galileo Press. ISBN: 978-83-246-4732-3
[3] - źródło http://kis.pwszchelm.pl/publikacje/III/Jaszuk.pdf Marek Jaszuk Państwowa Wyższa Szkoła Zawodowa w Chełmie

Pozostałe źródła:

-Sztuczna inteligencja, wykład 12: Przeszukiwanie Dr hab. inż. Grzegorz Dudek Wydział Elektryczny Politechnika Częstochowska, Częstochowa 2004
-http://www.abaco.com.pl/produkty/c_plany.html
-Even S., Itai A., Shamir A., On the complexity of timetable and multicommodity flow problems, SIAM Journal of Computing, 5(4), 1976

-Tripathy A., School timetabling – a case in large binary integer linear programming, Management Science 30(12), 1984