Factory Network

projekt koncepcyjny

Piotr Czechowski, Konrad Wajda, Bartłomiej Fryz

Etap 1: Analiza obiektów

Przedstawiamy wstępnie obiekty występujące w programie oraz zadania im odpowiadające.

1. LoadingRamp
   1.1. Dostarczanie półproduktów z zadaną **częstotliwością**.
   1.2. Przekazuje półprodukty pracownikom z **własnym rozkładem prawdopodobieństwa.**
   1.3. Przechowuje **możliwe** połączenia z zadanym prawdopodobieństwem.
   1.4. Posiada własne id

2. Worker
   2.1. Przetwarza półprodukty otrzymane z rampy z odpowiadającym mu **czasem przetwarzania**
   2.2. Przechowuje i przetwarza półprodukty **FIFO** lub **FILO**
   2.3. Przekazuje produkty do magazynu z **własnym rozkładem prawdopodobieństwa.**
   2.4. Przechowuje **możliwe** połączenia z zadanym prawdopodobieństwem.
   2.5. Posiada własne id

3. StoreHouse
    3.1. Przechowuje gotowe produkty.
    3.2. Posiada własne id

4. Link
    4.1. Posiada wskaźnik połączenia.
    4.2. Posiada prawdopodobieństwo.

5. Product
    5.1. Przechowuje swoje **id** oraz informację o tym czy jest **przetworzony**
    5.2. Pozwala rozróżniać poszczególne produkty.

6. Simulation
    6.1. Inicjuje ConfigurationLoader.
    6.2. Inicjuje generację raportu końcowego.
    6.3. Przechowuje instancje klas LoadingRamp, Worker, StoreHouse,
    6.4. Zapewnia użytkownikowi kontrolę nad procesem.
        6.4.1. Przechodzenie do kolejnych kroków.
        6.4.2. Raporty chwilowe.
        6.4.3. Wybranie pliku konfiguracyjnego (startowego).
    6.5. Symuluje kolejne tury.
    6.6. Na podstawie listy obiektów generuje o nich raport.

7. ConfigurationLoader
    7.1. Otwiera i analizuje plik konfiguracyjny.
    7.2. Sprawdza poprawność konfiguracji.
    7.3. Na jego podstawie tworzy obiekty.

Etap 2: Diagram klas

Na podstawie analizy obiektów tworzymy początkowy diagram klas. Jest on na razie nie zgodny z zasadami SOLID. (załącznik JPG)

Etap 3: Refaktoryzacja

Analizujemy diagram klasowy pod względem abstrakcji, wielokrotnego użycia, czytelności oraz spełniania zasad SOLID. Dokonujemy potrzebnych zmian.

Błędy w dotychczasowym diagramie:

- zmienna stanu przetworzenia “state” w klasie “Product” nie jest potrzebna,
- getRaport powinien zostać wyciągnięty do osobnej klasy i nie powinien być wykonywany wewnątrz Nodów (bardziej prawdopodobna jest zmiana wyglądu raportu niż struktury sieci),
- wymagana reorganizacja abstrakcji ponieważ w obecnej sytuacji Link nie może wskazywać na Magazyn,
- potrzebny dodatkowy adapter dla Stosu oraz Kolejki,

Etap 4: Implementacja

W metodologii TestDrivenDevelopment. Najpierw piszemy testy jednostkowe dla każdej klasy.